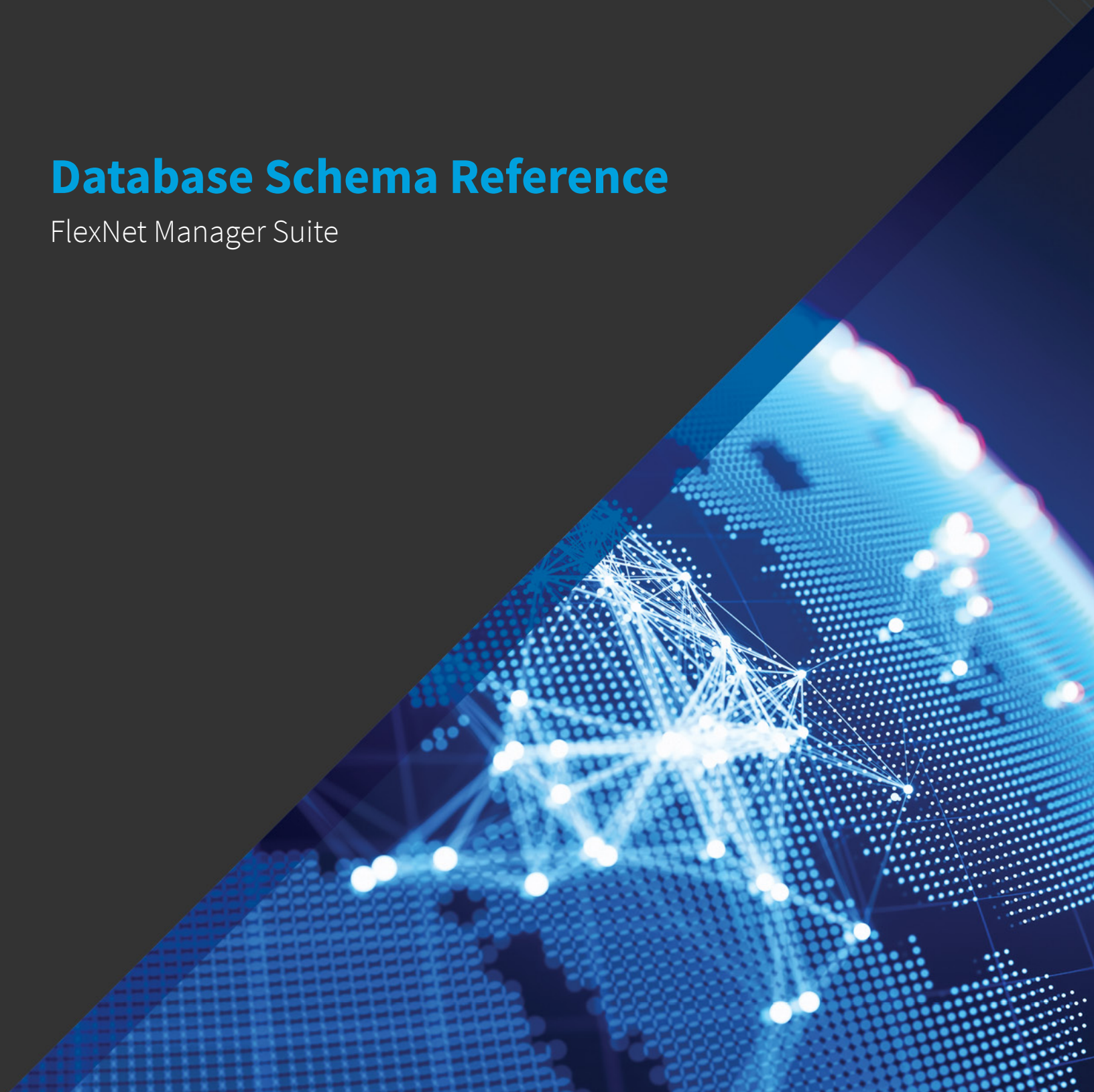
Database Schema Reference

FlexNet Manager Suite



Legal Information

Document Name: FlexNet Manager Suite Schema Reference version 2019 R2.2 (for on-premises and cloud implementations)

Part Number: FMS-14.2.0-DR01

Product Release Date: February 27, 2020

Copyright Notice

Copyright © 2020 Flexera.

This publication contains proprietary and confidential technology, information and creative works owned by Flexera and its licensors, if any. Any use, copying, publication, distribution, display, modification, or transmission of such publication in whole or in part in any form or by any means without the prior express written permission of Flexera is strictly prohibited. Except where expressly provided by Flexera in writing, possession of this publication shall not be construed to confer any license or rights under any Flexera intellectual property rights, whether by estoppel, implication, or otherwise.

All copies of the technology and related information, if allowed by Flexera, must display this notice of copyright and ownership in full.

FlexNet Manager Suite incorporates software developed by others and redistributed according to license agreements. Copyright notices and licenses for this externally-developed software are provided in the link below.

Intellectual Property

For a list of trademarks and patents that are owned by Flexera, see http://www.flexera.com/intellectual-property. All other brand and product names mentioned in Flexera products, product documentation, and marketing materials are the trademarks and registered trademarks of their respective owners.

Restricted Rights Legend

The Software is commercial computer software. If the user or licensee of the Software is an agency, department, or other entity of the United States Government, the use, duplication, reproduction, release, modification, disclosure, or transfer of the Software, or any related documentation of any kind, including technical data and manuals, is restricted by a license agreement or by the terms of this Agreement in accordance with Federal Acquisition Regulation 12.212 for civilian purposes and Defense Federal Acquisition Regulation Supplement 227.7202 for military purposes. The Software was developed fully at private expense. All other use is prohibited.

Overview

The data underlying FlexNet Manager Suite is arranged in a number of distinct databases. Most frequently these databases live within a single database server, although in very large scale implementations, it is possible to implement them across multiple servers.

The most fundamental distinction is between:

- Data imported from various instances of the FlexNet inventory agent as software and hardware inventory from individual computers within the enterprise (see Inventory Database Schema)
- Data used to calculate license positions, combining the software applications recognized from the imported inventory, the license entitlements collated from purchase records and other sources, structural information about the enterprise itself, and so on (see Compliance Database Schema).

A small set of tables is common to both these databases. These shared tables are documented within each of the above chapters.

In support of this basic structure, there are also the following major aspects:

- Staging tables used to rationalize data being imported into the main compliance database by `ComplIanceReader.exe` (see Compliance Reader Database Schema)
- A separate schema for presenting summarized license information on a once-separate web portal (see License Portal Database Schema).

Each of the chapters covering these schemata has a common structure:

- The chapter header includes a list of different *aspects* of the data described in the chapter. (These aspects are also the lowest level included in the summary table of contents for the entire volume.)
- The chapter header is followed by a reminder of the information structure in each of the database table descriptions.
- Each aspect then has a section header page listing all the individual database tables contained within that aspect.
- Finally, the detailed topics, one for each database table, listing all the properties (columns) in the table and various attributes of each one.

This structure makes it easy to drill down from a high-level understanding of the data structure to an individual table. Conversely, if you know a table name, use the PDF search mechanism in your reader software to locate its description. Similarly, you can also search for individual properties within tables, even when you don't know their provenance.

One final chapter takes a slightly different approach. Rather than documenting an internal schema, it covers the schema used for spreadsheets importing inventory information, and the mapping of those columns to the relevant database tables and column.

This document is not an exhaustive description of the entire database structure. For example, the system makes widespread use of views extracted from these underlying tables for (amongst other reasons) performance improvements. These views are not documented here. Nor are the mechanisms used in a multi-tenant implementation for partitioning each tenant's data made explicit in this document. However, this is a complete description of all the basic data tables from which all else is derived.

Furthermore, the descriptions of each database table are compiled automatically using the same mechanism that generates the database schemata themselves. This process guarantees complete coverage of all tables at each release.

Contents

1

Logical Data Models

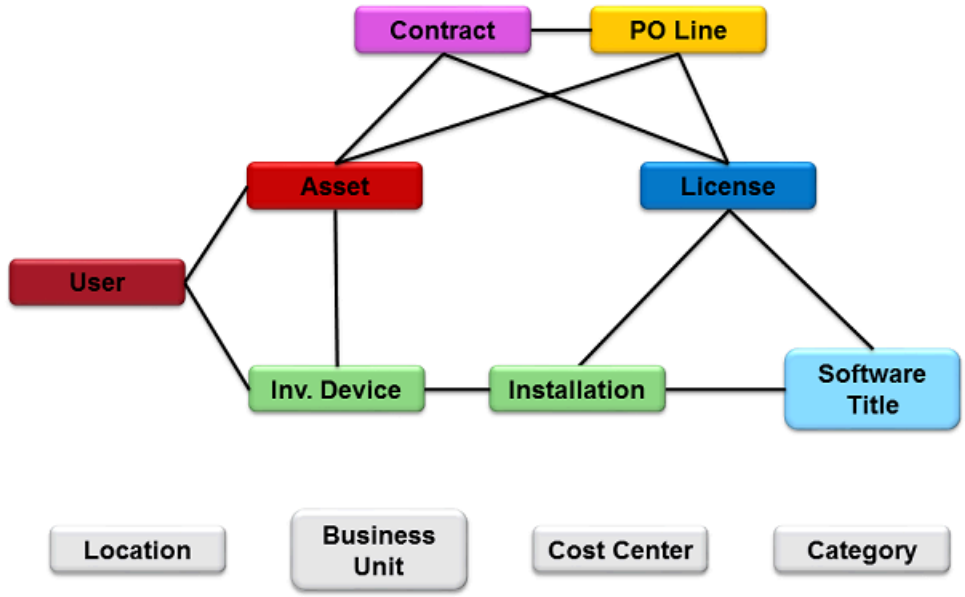
In a database schema of this size, it can be hard to get your bearings. To help you understand the territory, this topic contains some logical data models, generally centered around key database objects.

Note: *These illustrations are not detailed schema diagrams (such as you could generate using Microsoft SQL Server). Instead, they provide high-level "mud maps" of key objects in the FlexNet Manager Suite system, with some indications of how they relate to one another. These are logical or conceptual models. For details about how individual database tables link to each other, see the detailed descriptions in the following pages.*

Overview

The first diagram gives an overview of the major components (database objects) in the system. Because the four kinds of enterprise groups shown across the bottom of the diagram have so many possible links to the other objects, no links for these are included in the overview (more links are visible in the following more specialized diagrams).

Figure 1: Overview model



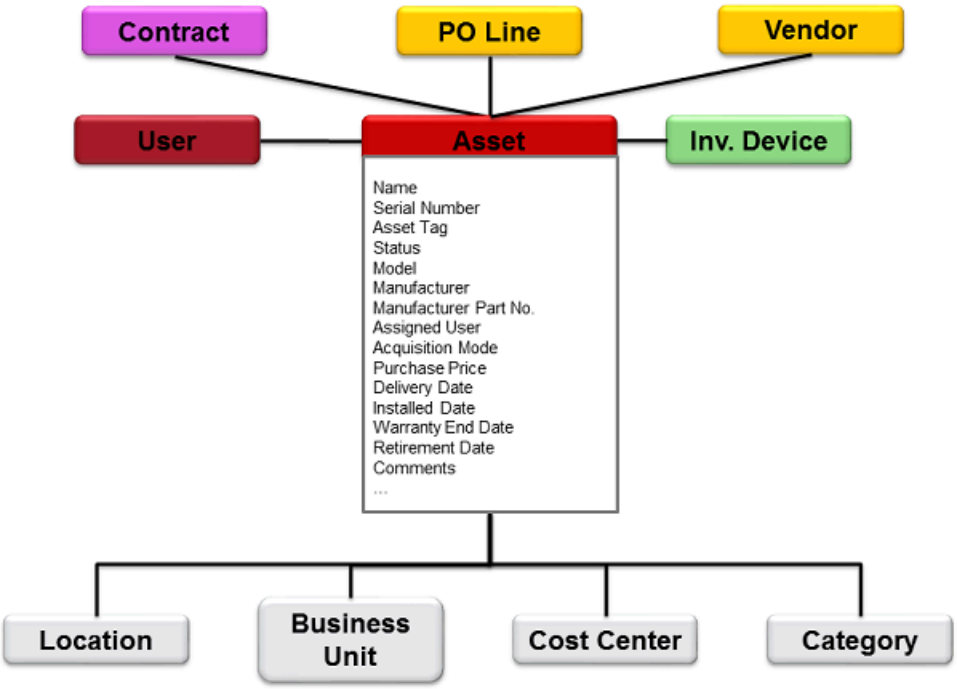
The following logical models focus on one of these objects at a time, providing a few of the more important attributes or

properties of those key objects in the database, and fleshing out more details of their relationships to other objects.

Asset model

In FlexNet Manager Suite, an asset is an item of hardware (including, but not limited to, computer hardware). Like a physical asset register, these records are kept separate from the inventory records that may contribute to the details about computer hardware. For this reason, you see the close link between the asset object and the inventory device object. Also notice that an asset may be linked to one of each kind of enterprise group (shown in gray across the bottom of the diagram).

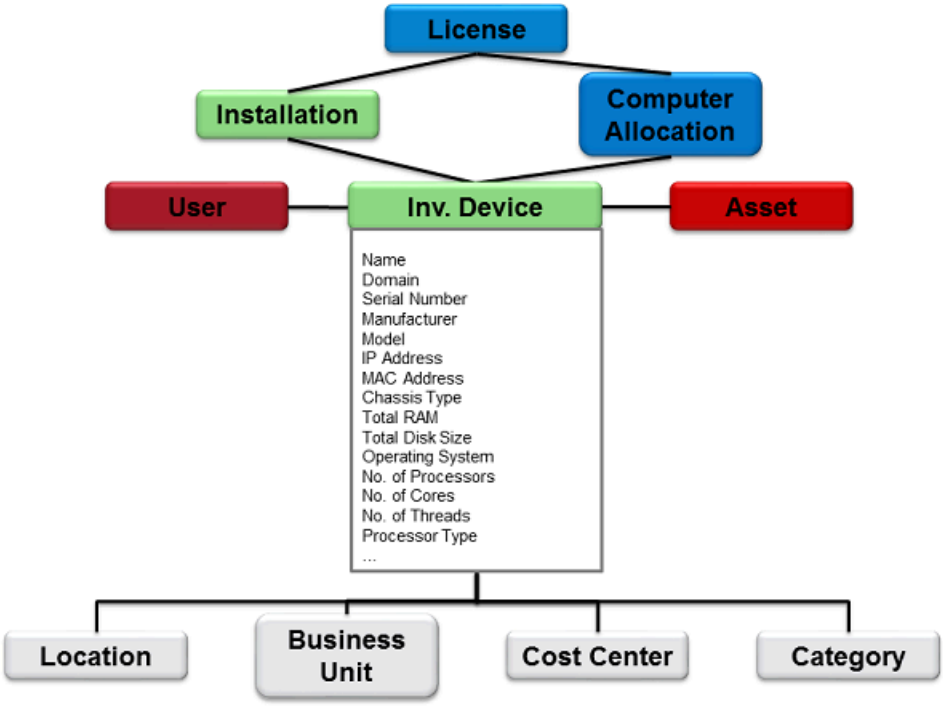
Figure 2: Asset model



Inventory device model

Inventory devices are records of hardware objects from which hardware and (most often) software inventory has been collected. Even though inventory devices are closely related to assets, they have their own potential links to one of each kind of enterprise group. To avoid double handling, there are settings in the web interface for FlexNet Manager Suite to have the ownership of one track the other. However, it is possible to assign these records separately, so that (for example) you may link an asset to the Illinois state head office for its asset register, but have the inventory device linked to a location in the Itasca local office.

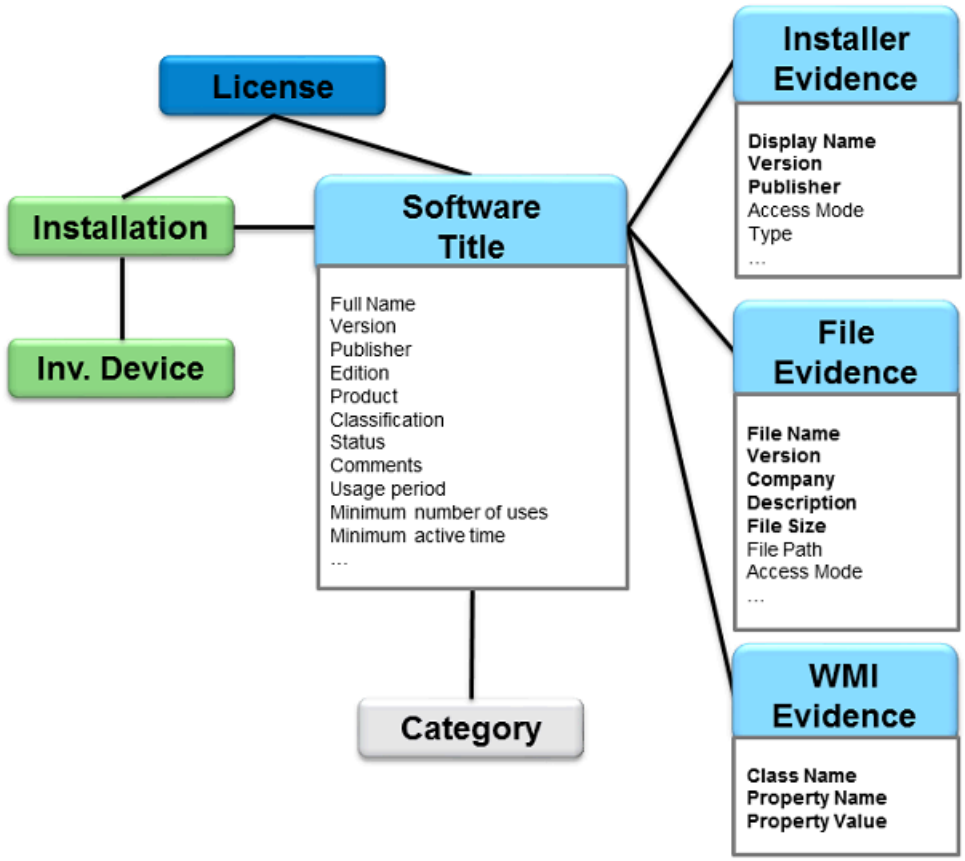
Figure 3: Inventory device model



Software title model

A software title database object models what is called an *application* in the web interface of FlexNet Manager Suite. *Evidence* of various types is whatever may be found on a computer that identifies the application, with the mapping between evidence and application normally supplied through the Application Recognition Library. Applications do not link directly with inventory devices: there is an intermediate installation object that provides this link. Note also that some server-based software has additional evidence types (such as access and usage evidence) that helps to track requirements for CALs.

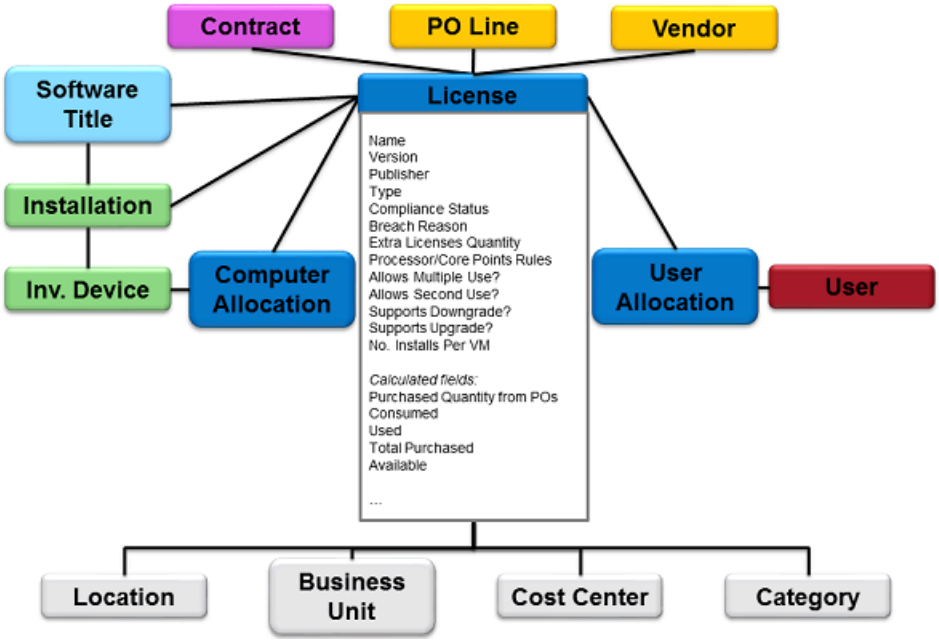
Figure 4: Software title model



License model

The license is perhaps the most central object in the data model, since ultimately everything else exists to allow correct calculation of incoming entitlements and consumption of those entitlements within your enterprise. Notice that individual allocations, controlled through the license properties in the web interface, are kept as separate records linking the license record either to an inventory device or a user.

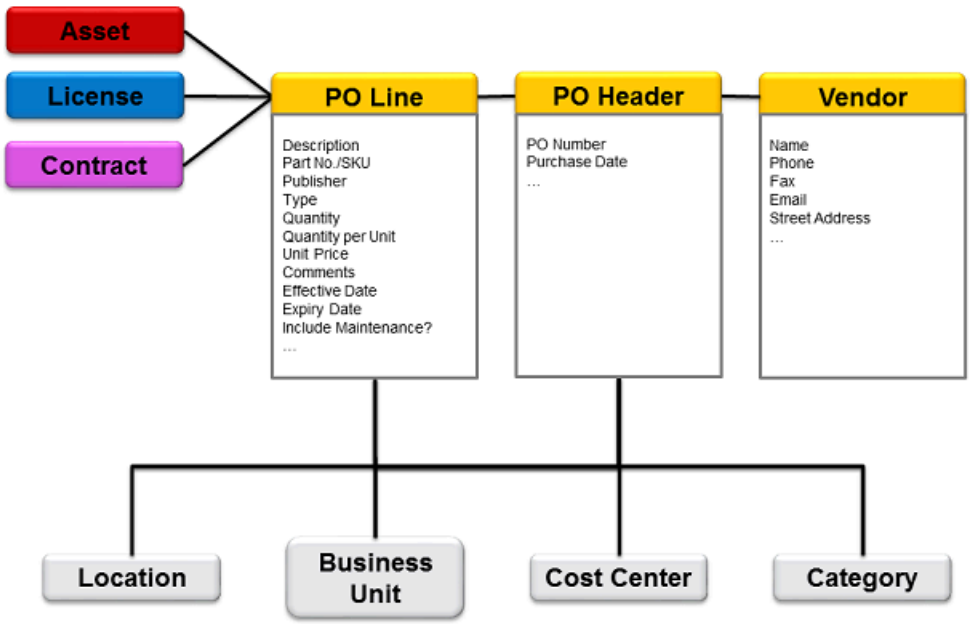
Figure 5: License model



Purchase order model

For historical reasons, the database models a purchase order as a separate header record and one or more line items from that purchase order. In the web interface for FlexNet Manager Suite, purchases are now represented as separate objects (each purchase maps to one PO line in the database), with purchase order headers represented only by a few common values appended to the top of the purchase properties. The common structure for purchases may be used for a variety of objects: software and hardware purchases, as well as renewals of maintenance contracts and the like.

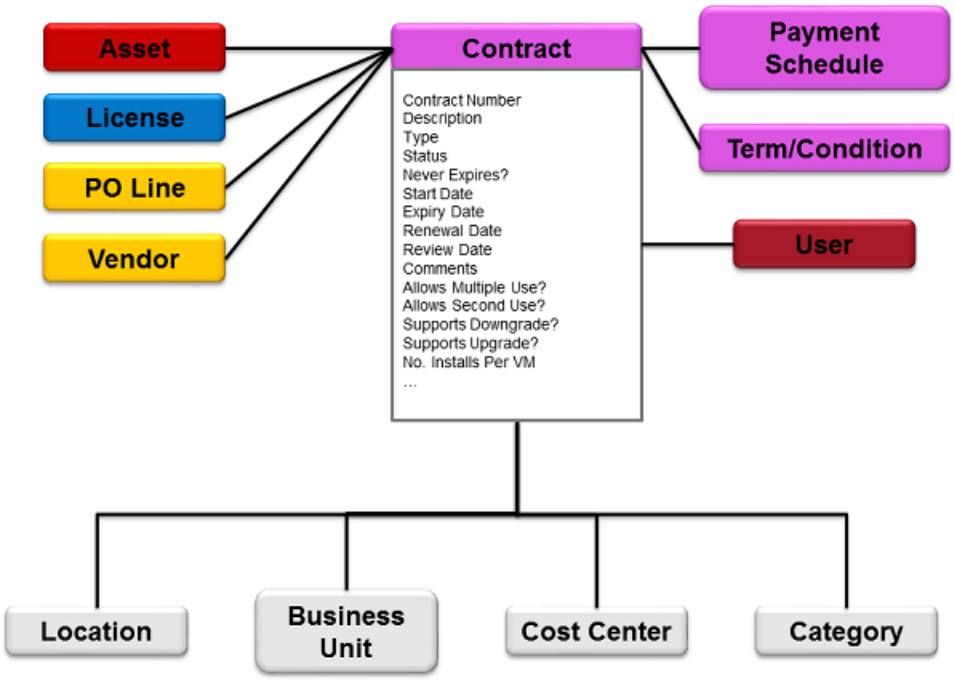
Figure 6: Purchase model



Contract model

Contracts may be used to track any kind of real-world contract, and they are particularly useful for modeling support contracts or maintenance (or in Microsoft terms, Software Assurance). These are also the mechanism for tracking regular payments. Since a contract may include many terms and conditions, these are modeled as separate objects in the database.

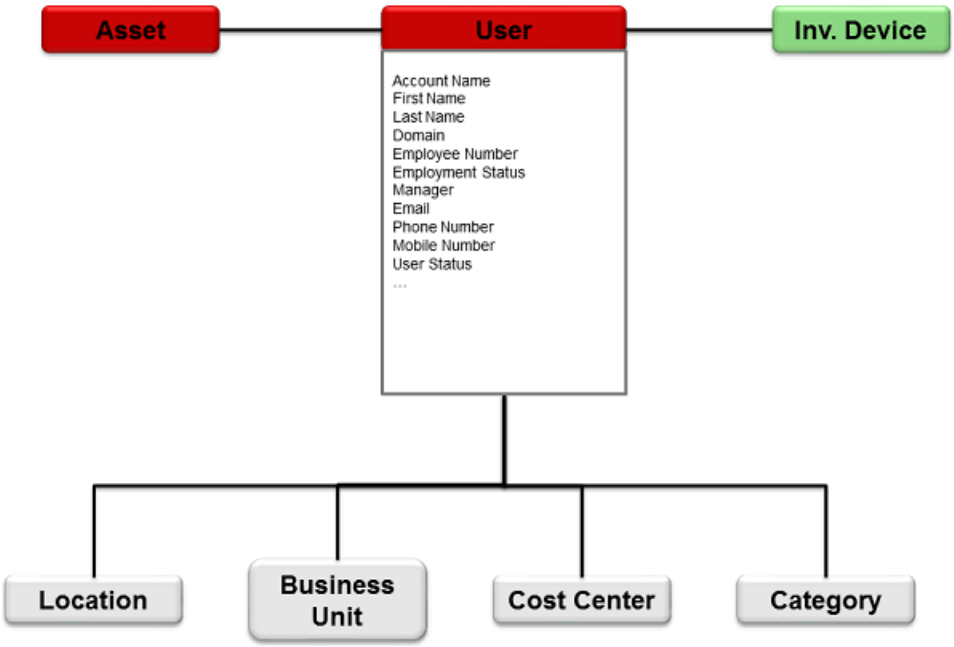
Figure 7: Contract model



User model

A user is not a person operating the FlexNet Manager Suite system itself (these people are called operators, and are managed separately). A user is a person allowed to use an inventory device, or may be also be linked as the owner of an asset. In earlier incarnations, these were called "end users", if that helps to clarify the distinction from operators.

Figure 8: User model



2

Compliance Database Schema

This chapter describes the schema for the main database underlying FlexNet Manager Suite.

Separately documented is the schema for the inventory tables for inventory gathered by the FlexNet inventory agent, either when installed on 'adopted' devices, or when executing a remote, zero-touch inventory (see Inventory Database Schema).

Some tables from that inventory database are (correctly) duplicated in this compliance database, and these shared tables are also listed toward the end of this chapter.

Information Structure

The following information is provided about database tables. Items appear only when relevant to the database column, and are suppressed where they do not apply. Two of these items (shown bold) are columns in the following pages, and the remainder are displayed within the **Details**.

Item	Comment
Database Column	The name of the column in the SQL table.
Type	The data type of the contents of the database column.
Size	For types that have a maximum capacity, the upper limit is provided in parentheses.
Key	The word “Key” appears when a column is a unique key field within the table. It is possible for several database columns to be part of the key, so that this indicator may appear for several columns in a table.
Generated ID	This indicates that a numeric ID is assigned by the database.
Nullable	If this indicator is present, the database column permits nulls.
Computed	This indicator appears for columns that are automatically computed by the database.
Default	If a column has a default value declared in the schema, this is specified at the end of the first set of details for the column.

Item	Comment
Details	Describes the data stored in the database column, including many of the indicators described above.

BatchProcessing.Common Tables

The complete set of database tables documented here includes:

- BatchProcessExecution table (see BatchProcessExecution Table)
- BatchProcessExecutionData table (see BatchProcessExecutionData Table)
- BatchProcessExecutionDataName table (see BatchProcessExecutionDataName Table)
- BatchProcessSchedule table (see BatchProcessSchedule Table)
- BatchProcessStatus table (see BatchProcessStatus Table)
- BatchProcessType table (see BatchProcessType Table)
- BatchProcessTypeLimit table (see BatchProcessTypeLimit Table)

BatchProcessExecution Table

`BatchProcessExecution` is a table storing the details of batch processes requested and executed.

Table 1: Database columns for `BatchProcessExecution` table

Database Column	Details
`BatchProcessExecutionID`	*Type:* integer. Key. Generated ID A unique identifier for a batch processor.
`GUID`	*Type:* unique identifier. Key The GUID identifying a batch process execution.
`BatchProcessTypeID`	*Type:* integer. Key The type of this batch process execution. Foreign key to the `BatchProcessType` table.
`Submitted`	*Type:* datetime. Key The date and time at which this batch process execution was submitted.
`OperatorLogin`	*Type:* text (max 512 characters). Nullable The login name of the operator requesting the batch process, NULL indicates a system request.

Database Column	Details
`BeaconID`	*Type:* integer. Key. Nullable The ID of the beacon which requested a batch process execution. Foreign key to the `Beacon` table.
`BatchProcessorHostname`	*Type:* text (max 128 characters). Key. Nullable The batch processor responsible for the execution of this batch process. A processor by this name may be in the `BatchProcessor` table, but this is not required.
`BatchProcessStatusID`	*Type:* integer Status of the batch process execution. Foreign key to the `BatchProcessStatus` table.
`StartTime`	*Type:* datetime. Nullable The date and time the batch process execution was started.
`FinishTime`	*Type:* datetime. Key. Nullable The date and time the batch process execution finished.
`Progress`	*Type:* integer Percentage indicator of how far through the batch process execution is.
`ReturnCode`	*Type:* integer. Nullable The return code of the batch process execution.
`Output`	*Type:* text. Nullable Contains any output reported by a batch process execution.
`GroupName`	*Type:* text (max 50 characters). Nullable The group name used to partition this batch process. Only relevant for types that require separation by group.
`TenantUID`	*Type:* text (max 40 characters). Key. Nullable The tenant UID for this batch process. Only relevant for types that require separation by tenant.
`RawMessage`	*Type:* text. Nullable The raw, serialized message. Used for pending messages to reconstruct the queue when the batch processor restarts.

BatchProcessExecutionData Table

This table stores any extra data needed for a `BatchProcessExecution` record.

Table 2: Database columns for `BatchProcessExecutionData` table

Database Column	Details
`BatchProcessExecutionDataID`	*Type:* integer. Key. Generated ID A unique identifier for this table.
`BatchProcessExecutionID`	*Type:* integer. Key The ID of the BatchProcessExecution record this data is asociated with. Foreign key to the `BatchProcessExecution` table.
`BatchProcessExecutionDataNameID`	*Type:* integer. Key An identifier for the data being stored in this row
`DataValue`	*Type:* text The value being stored in this row

BatchProcessExecutionDataName Table

This table holds a list of the different types of data that can be stored in `BatchProcessExecutionData`.

Table 3: Database columns for `BatchProcessExecutionDataName` table

Database Column	Details
`BatchProcessExecutionDataNameID`	*Type:* integer. Key. Generated ID A unique identifier for this table.
`Name`	*Type:* text (max 128 characters). Key Name of the setting.

BatchProcessSchedule Table

`BatchProcessSchedule` stores the schedule of a batch process.

Table 4: Database columns for `BatchProcessSchedule` table

Database Column	Details
`BatchProcessScheduleID`	*Type:* integer. Key. Generated ID A unique identifier for this table.
`BatchProcessTypeID`	*Type:* integer. Key The process type ID this schedule belongs to. Foreign key to the `BatchProcessType` table.

Database Column	Details
`TenantUID`	*Type:* text (max 40 characters). Key. Nullable The tenant UID for this batch schedule.
`BatchProcessScheduleData`	*Type:* text The Quartz scheduler data
`UpdatedBy`	*Type:* text (max 200 characters). Nullable The last operator to update the event.
`UpdatedDate`	*Type:* datetime. Nullable The date the event was last updated.
`GUID`	*Type:* unique identifier. Key Unique identifier for schedule.
`LastRun`	*Type:* datetime. Nullable The datetime this schedule was last executed.
`Enabled`	*Type:* boolean The datetime this schedule was last executed.

BatchProcessStatus Table

`BatchProcessStatus` is a static table listing status values for batch process execution.

Table 5: Database columns for `BatchProcessStatus` table

Database Column	Details
`BatchProcessStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `BatchProcessStatus`. Possible values and the corresponding default strings are: • 1 = Submitted • 2 = Queued • 3 = Processing • 4 = Success • 5 = Error • 6 = Duplicate

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an status of batch process execution. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

BatchProcessType Table

`BatchProcessType` is a static table storing the types of batch processes

Table 6: Database columns for `BatchProcessType` table

Database Column	Details
`BatchProcessTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `BatchProcessType`. Possible values and the corresponding default strings are: • 1 = License reconcile • 2 = PO line import • 3 = Enterprise group import • 4 = User assignment import • 5 = Inventory import • 6 = Active directory import • 7 = Entitlement recommendations recalculation • 8 = SAP user recommendations export • 9 = Business adapter import • 10 = Generate business adapter config • 15 = ServiceNow export • 16 = FNMEA enterprise groups export • 17 = IBM Passport Advantage import • 18 = Data Warehouse access rights update • 19 = Update license consumption of IBM PVU licenses • 20 = Data Warehouse export • 21 = Import SAP inventories • 22 = Import SAP package license • 23 = Inventory import and license reconcile • 24 = Recognition data import • 25 = Inventory manager compliance import • 26 = Compliance import readers only • 27 = Compliance import writers only • 28 = Recognition data download • 29 = Recognition data cleanup • 30 = IM Data maintenance

Database Column	Details
	• 31 = SAP user and activity information import • 32 = Inventory import spreadsheet and license reconcile • 33 = FNMP Data maintenance • 34 = FNMP software usage history update • 35 = Delete activity log history • 36 = Baseline import processing • 37 = Sync FNMS tenants with Cognos • 38 = IM Tenant Data maintenance • 30 = Data Warehouse partial export • 31 = SAP user and activity information import • 32 = Inventory import spreadsheet • 33 = FNMP Data maintenance • 34 = FlexNet Manager Platform software usage history update • 35 = Activity log history delete • 36 = Baseline import processing • 37 = Sync FNMS tenants with Cognos • 38 = IM Tenant Data maintenance • 39 = FlexNet Manager Suite dashboard update • 40 = FlexNet Manager Suite Precalc update • 41 = Generate scoped Oracle LMS archive for logged in operator • 42 = Tenant deletion • 43 = SAP license position calculation • 44 = CSA Billing Export
`TypeName`	*Type:* text (max 256 characters). Key The unique name of the batch process type.
`ResourceName`	*Type:* text (max 256 characters) The unique name of the localizable resource string representing a batch process type. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
StarvedAt	*Type:* integer. Nullable The age, in minutes, after which a task of this type will be given priority over other tasks to avoid starvation.
Timeout	*Type:* integer. Nullable The age, in minutes, after which a task of this type will be regarded as failed if its processor becomes unresponsive.
BatchProcessTypeLimitID	*Type:* integer. Nullable An optional reference to a limit that will restrict the number of items of this type that can execute at the same time.

BatchProcessTypeLimit Table

BatchProcessTypeLimit is a table storing the limits placed on the parallel execution of tasks within the Batch Processor. A limit is associated with one or more BatchProcessTypes. The limit value is the number of tasks of the associated types that may be executed at any one time.

Note that these limits are applied after the standard parallel execution restrictions are applied. This means that these limits will generally affect a single tenant system. They will take effect only if the limit is applied to types that are allowed to run in parallel for a tenant. For example, if a limit is applied to a types that run the ComplianceReader executable, the Business importer and the ARL import, it may be possible to reach the limit.

In a multi-tenant system, the limits allow the system administrator to define reasonable limits to try to ensure that the Batch scheduler does not overload the hardware it is allotted.

Table 7: Database columns for BatchProcessTypeLimit table

Database Column	Details
BatchProcessTypeLimitID	*Type:* integer. Key. Generated ID A unique identifier for a BatchProcessTypeLimit.
Name	*Type:* text (max 128 characters). Key The name of this BatchProcessTypeLimit. This name will be used internally to reference the limit, and will be shown in the tracing output.
MaxTasks	*Type:* integer The number of tasks associated with this limit that may be executed in parallel by the Batch scheduler. A zero or negative value in this column will cause the limit to be ignored.

BatchProcessing Tables

The complete set of database tables documented here includes:

- BatchProcessor table (see BatchProcessor Table)
- BatchProcessorProcessType table (see BatchProcessorProcessType Table)

BatchProcessor Table

`BatchProcessor` is a table storing the machines responsible for executing batch processes.

Table 8: Database columns for `BatchProcessor` table

Database Column	Details
`BatchProcessorID`	*Type:* integer. Key. Generated ID A unique identifier for a batch processor.
`Hostname`	*Type:* text (max 128 characters). Key The host name of this batch processor.
`LastHeartbeat`	*Type:* datetime. Nullable The UTC date and time this batch processor configured.
`LastExecution`	*Type:* datetime. Nullable The UTC date and time this batch processor last executed a batch process.

BatchProcessorProcessType Table

This table records the mapping of process types to batch processors.

Table 9: Database columns for `BatchProcessorProcessType` table

Database Column	Details
`BatchProcessorID`	*Type:* integer. Key The ID of the BatchProcessor record this data is asociated with. Foreign key to the `BatchProcessor` table.
`BatchProcessTypeID`	*Type:* integer. Key The ID of the BatchProcessType record this data is asociated with. Foreign key to the `BatchProcessorType` table.

Compliance.Logic.Administration Tables

The complete set of database tables documented here includes:

- APIServiceAccount table (see APIServiceAccount Table)
- ComplianceConnection table (see ComplianceConnection Table)
- ComplianceConnectionParameter table (see ComplianceConnectionParameter Table)
- ComplianceCultureType table (see ComplianceCultureType Table)
- ComplianceOperator table (see ComplianceOperator Table)
- ComplianceOperatorAudit table (see ComplianceOperatorAudit Table)
- ComplianceOperatorTenant table (see ComplianceOperatorTenant Table)
- ComplianceResourceString table (see ComplianceResourceString Table)
- ComplianceSetting table (see ComplianceSetting Table)
- ComplianceTenantSetting table (see ComplianceTenantSetting Table)
- ConfigurationFile table (see ConfigurationFile Table)
- ConfigurationFileType table (see ConfigurationFileType Table)
- ConnectionType table (see ConnectionType Table)
- Currency table (see Currency Table)
- MasterConfigurationFile table (see MasterConfigurationFile Table)
- MasterConfigurationFileDescription table (see MasterConfigurationFileDescription Table)
- MasterConfigurationFileOptionState table (see MasterConfigurationFileOptionState Table)
- MasterConfigurationFileSelection table (see MasterConfigurationFileSelection Table)
- MasterConfigurationFileVersions table (see MasterConfigurationFileVersions Table)
- OperatorTenantSetting table (see OperatorTenantSetting Table)
- ResourceStringCultureType table (see ResourceStringCultureType Table)
- RightDefinition table (see RightDefinition Table)
- SettingName table (see SettingName Table)
- TimezoneType table (see TimezoneType Table)

APIServiceAccount Table

Stores a collection of external API service accounts.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 10: Database columns for `APIServiceAccount` table

Database Column	Details
`APIServiceAccountID`	*Type:* integer. Key. Generated ID Unique identifier for a API service account.
`ComplianceOperatorID`	*Type:* integer. Key. Nullable Reference to a compliance operator.
`AccessThreshold`	*Type:* integer. Nullable API access alert threshold
`AccessCount`	*Type:* integer API access count.
`LastSync`	*Type:* datetime. Nullable Indicates the last datetime this account is synced with FNOOD or validateToken API is called.
`Description`	*Type:* text (max 256 characters). Nullable Description for this service account.
`CreationUser`	*Type:* text (max 256 characters). Nullable Created by.
`CreationDate`	*Type:* datetime. Nullable Creation date.
`UpdatedUser`	*Type:* text (max 256 characters). Nullable Operator who made the latest change to the currency record.
`UpdatedDate`	*Type:* datetime. Nullable Updated date

ComplianceConnection Table

The `ComplianceConnection` table stores details about databases configured for use in compliance imports, such as Microsoft SMS.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 11: Database columns for `ComplianceConnection` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Generated ID A unique identifier for a compliance connection.
`ConnectionTypeID`	*Type:* integer. Key The compliance connection type. Foreign key to the `ConnectionType` table.
`ConnectionName`	*Type:* text (max 128 characters). Key The internal, unique name of the connection.
`ConnectionNameDisplayName`	*Type:* text (max 64 characters) The name of the connection for display purposes.
`UseFnmpDbServerAsSource`	*Type:* boolean Use the FNMP database server as the source.
`Server`	*Type:* text (max 128 characters). Nullable The name of the SQL Server.
`UseWindowsAuth`	*Type:* boolean. Nullable If this field is set to `True`, the connection will use Windows authentication when connecting to the database. If `False`, SQL authentication will be used.
`Username`	*Type:* text (max 128 characters). Nullable The username to use when connecting with SQL authentication.
`Password`	*Type:* text. Nullable The password to use when connecting with SQL authentication.
`DatabaseName`	*Type:* text (max 128 characters). Nullable The name of the database to connect to.
`ConnectionString`	*Type:* text. Nullable The connection string used to connect to a datasource.
`LastImportDate`	*Type:* datetime. Nullable Date and time when data from this data source was successfully imported into the staging area (reader execution). The imported data may not have been applied to the core tables.
`LastImportStarted`	*Type:* datetime. Nullable Date and time when the import from this data source started.
`LastImportEnded`	*Type:* datetime. Nullable Date and time when the import from this data source ended.

Database Column	Details
`LastImportSuccessful`	*Type:* boolean Whether or not the last import attempted for this datasource succeeded or failed.
`SourceType`	*Type:* text (max 256 characters) The source database type (one of several predefined values, such as `ManageSoft` or `SMS`).
`SourceTypeDisplayName`	*Type:* text (max 128 characters) A version of the SourceType field, that has been scoped to be specific to this connection.
`Signature`	*Type:* text (max 128 characters) A connection signature optionally given by the source database. This allows the source database to identify its connection.
`PrimaryConnection`	*Type:* boolean. Key Set this to `True` if this is the primary data source to import from. If computers or users exist in multiple connections, data from the primary connection is always given precedence.
`TestConnection`	*Type:* boolean Indicate if this connection is a test connection. If this is set to `True` writer will not populate target FNMP tables with data in the imported tables from this connection. If this is set to `False` writer will populate data from this connection as is. Compliance Reader Editor UI sets connection as test so that test data would not accidentally be written to target FNMP tables.
`Enabled`	*Type:* boolean Indicate if this connection is enabled. If this is set to `False` reader will not import data from this connection.
`GroupName`	*Type:* text (max 256 characters). Nullable The GroupName represents subgroups of data from the source. For example, for a citrix connection, this stores a farm name. If this is Null, then there is no sub-grouping (import all).
`ExpiryPeriod`	*Type:* integer. Nullable The number of days before considering records in `ImportedComputer` to be out of date and should be considered stale. NULL means use the Compliance Setting value StaleInventoryThreshold. 0 means always include device data regardless of age.
`PerformStaleInventory Check`	*Type:* boolean Indicates if this connection needs to have the inventory checked to see if data is considered stale. It is reset to 1 after completing the reader's step of an import.

Database Column	Details
`IsRemote`	*Type:* boolean Is this a remote connection, where the source side of the readers are running on a remote location (an Inventory Beacon)?
`ConnectionExID`	*Type:* unique identifier. Key The externally unique identifier for this connection, that can be used by both an Inventory Beacon and the server to track a connection.
`BeaconUID`	*Type:* unique identifier. Key. Nullable The unique ID of the beacon where this connection is running.

ComplianceConnectionParameter Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 12: Database columns for `ComplianceConnectionParameter` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key A unique identifier for a compliance connection.
`Name`	*Type:* text (max 256 characters). Key The name of the compliance connection parameter
`Value`	*Type:* text The value of the compliance connection parameter
`Type`	*Type:* text (max 64 characters) The type of compliance connection parameter

ComplianceCultureType Table

The `ComplianceCultureType` table holds all the different languages that FlexNet Manager Suite supports.

Table 13: Database columns for `ComplianceCultureType` table

Database Column	Details
`CultureType`	*Type:* text (max 12 characters). Key A unique identifier for a culture type.
`DefaultCulture`	*Type:* boolean Indicates whether this language is a default language on the system.
`Installed`	*Type:* boolean Indicates whether string for this language are installed.
`DisplayName`	*Type:* text (max 80 characters) The display name for this culture.

ComplianceOperator Table

`ComplianceOperator` stores the list of people (operators) authorized to use FlexNet Manager Suite. Operators need not be end-users of the enterprise.

Table 14: Database columns for `ComplianceOperator` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key. Generated ID A unique identifier for the operator.
`OperatorLogin`	*Type:* text (max 256 characters). Key The login (account name) of the operator. Usually of the form [domain\account].
`OperatorName`	*Type:* text (max 512 characters). Nullable The name of the operator.
`IsEnabled`	*Type:* boolean When `False`, this operator may not use FlexNet Manager Suite, even if he or she is assigned to roles granting them access.
`Email`	*Type:* text (max 200 characters). Nullable The operator's email address.
`JobTitle`	*Type:* text (max 128 characters). Nullable The job title of the end-user.
`ComplianceUserID`	*Type:* integer. Key. Nullable An optional link to an end-user in the system. Foreign key to the `ComplianceUser` table.

Database Column	Details
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`BusinessReportingToken`	*Type:* text (max 256 characters). Nullable A token that is issued to an operator to allow them to authenticate with the business reporting framework.
`TenantID`	*Type:* small integer. Nullable The default tenant that this operator works on. Note that there is no tenant-filtered view on this table.
`GlobalOperator`	*Type:* boolean Allows an operator to access all tenants.
`Interactive`	*Type:* boolean Non-interactive accounts are service accounts.
`LastLogin`	*Type:* datetime. Nullable Last login datetime.
`LastLogout`	*Type:* datetime. Nullable Last logout datetime.

ComplianceOperatorAudit Table

`ComplianceOperatorAudit` is a multi-tenant table that stores the last login and log out date and time for each operator per tenant

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 15: Database columns for `ComplianceOperatorAudit` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key The operator of the setting. Foreign key to the `ComplianceOperator` table.
`LastLogin`	*Type:* datetime. Nullable Last login datetime.
`LastLogout`	*Type:* datetime. Nullable Last logout datetime.
`LastActive`	*Type:* datetime. Key. Nullable Last active datetime.
`IsActive`	*Type:* boolean. Nullable Indicates whether the operator has been active.
`IsPermanent`	*Type:* boolean Indicates whether the operator is permanently active.

ComplianceOperatorTenant Table

`ComplianceOperatorTenant` stores the list of people (operators) authorized to access a tenant.

Table 16: Database columns for `ComplianceOperatorTenant` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key The operatorID that the permission will be granted for.
`TenantId`	*Type:* integer. Key The tenantID that the operator will be granted access for.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`IsEnabled`	*Type:* boolean When `False`, this operator may not use FlexNet Manager Suite, even if he or she is assigned to roles granting them access.

ComplianceResourceString Table

The `ComplianceResourceString` table holds all the strings that require translation.

Table 17: Database columns for `ComplianceResourceString` table

Database Column	Details
`ResourceString`	*Type:* text (max 256 characters). Key A unique identifier for a string.

ComplianceSetting Table

The `ComplianceSetting` table holds the settings for the configuration and business rules of the application. With the introduction of `SettingName`, `ComplianceTenantSetting` and `OperatorTenantSetting` tables, if new global setting is to be added to `ComplianceSetting` table, the ComplianceSettingID must not overlap with those defined in `SettingName` table.

Table 18: Database columns for `ComplianceSetting` table

Database Column	Details
`ComplianceSettingID`	*Type:* integer. Key. Generated ID A unique identifier for a setting.
`SettingName`	*Type:* text (max 128 characters). Key A primary key for the setting.
`SettingValue`	*Type:* text (max 512 characters) The setting that indicates specified behavior.

ComplianceTenantSetting Table

`ComplianceTenantSetting` is a multi-tenant table that stores configuration and business rules specific to each tenant.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 19: Database columns for `ComplianceTenantSetting` table

Database Column	Details
`SettingNameID`	*Type:* integer. Key ID of the setting name. Foreign key to the `SettingName` table.
`SettingValue`	*Type:* text (max 512 characters). Nullable Value of the setting.

ConfigurationFile Table

The `ConfigurationFile` table stores configuration files generated from the master configuration files used by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 20: Database columns for `ConfigurationFile` table

Database Column	Details
`ConfigurationFileID`	*Type:* integer. Key. Generated ID A unique identifier for a configuration file.
`ConfigurationFileTypeID`	*Type:* integer. Key The configuration file type. Foreign key to the `ConfigurationFileType` table.
`Name`	*Type:* text (max 100 characters) The name of the configuration file.
`Revision`	*Type:* integer The revision of the configuration file.
`XMLFile`	*Type:* text The content of the configuration file.

ConfigurationFileType Table

`ConfigurationFileType` is a static table storing the types of configuration files used by FlexNet Manager Suite.

Table 21: Database columns for `ConfigurationFileType` table

Database Column	Details
`ConfigurationFileTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ConfigurationFileType`. Possible values and the corresponding default strings are: • 1 = SQL Server • 2 = Other (the inventory source is another type of data store, like an Excel sheet or MS Access database).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a configuration file type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ConnectionType Table

`ConnectionType` is a static table storing the types of connection that can be used to import data into FlexNet Manager Suite.

Table 22: Database columns for `ConnectionType` table

Database Column	Details
`ConnectionTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ConnectionType`. Possible values and the corresponding default strings are: • 1 = SQL Server • 2 = Other (the inventory source is another type of data store, like an Excel sheet or MS Access database). • 5 = PowerShell
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a connection type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

Currency Table

`Currency` stores a collection of currencies that can be used for money values.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 23: Database columns for `Currency` table

Database Column	Details
`CurrencyID`	*Type:* integer. Key. Generated ID Unique identifier for a currency.
`CurrencyName`	*Type:* text (max 256 characters) Name of currency.
`CurrencyResourceID`	*Type:* text (max 64 characters). Nullable The resource string containing the name of this currency to display on the user interface.
`CurrencyCode`	*Type:* text (max 32 characters). Key Code assigned to currency.
`LongPrefix`	*Type:* text (max 32 characters) Long prefix to display in front of the money value.
`LongSuffix`	*Type:* text (max 32 characters) Long suffix to display after the money value.
`LongFormat`	*Type:* text (max 80 characters). Nullable Long format of the currency. This is a calculated field.
`ShortPrefix`	*Type:* text (max 32 characters) Short prefix to display in front of the money value.
`ShortSuffix`	*Type:* text (max 32 characters) Short suffix to display after the money value.
`ShortFormat`	*Type:* text (max 80 characters). Nullable Short format of the currency. This is a calculated field.
`IsActive`	*Type:* boolean. Key Indicates whether this currency is enabled.
`Comments`	*Type:* text. Nullable Operator comments about this currency.

Database Column	Details
`Countries`	*Type:* text (max 2048 characters). Nullable A semicolon-separated list of the country codes for countries to which this currency is applicable.
`ActivationDate`	*Type:* datetime. Nullable Date currency was enabled.
`RetirementDate`	*Type:* datetime. Nullable Date that currency was retired.
`UpdatedUser`	*Type:* text (max 256 characters). Nullable Operator who made the latest change to the currency record.
`UpdatedDate`	*Type:* datetime. Nullable Date that the currency record was changed.

MasterConfigurationFile Table

The `MasterConfigurationFile` table stores master configuration files used by FlexNet Manager Suite.

Table 24: Database columns for `MasterConfigurationFile` table

Database Column	Details
`MasterConfigurationFileID`	*Type:* integer. Key. Generated ID A unique identifier for a configuration file.
`ConfigurationFileTypeID`	*Type:* integer. Key The configuration file type. Foreign key to the `ConfigurationFileType` table.
`Name`	*Type:* text (max 100 characters) The name of the configuration file.
`Revision`	*Type:* integer The revision of the configuration file.
`XMLFile`	*Type:* text The content of the configuration file.

MasterConfigurationFileDescription Table

Table 25: Database columns for `MasterConfigurationFileDescription` table

Database Column	Details
`MasterConfigurationFileDescriptionID`	*Type:* integer. Key A unique identifier for the insert type.
`CultureType`	*Type:* text (max 12 characters). Key The unique name of the culture type.
`DescriptionText`	*Type:* text The text description used to show what has changed in an update for the MasterConfigurationFile dataset.

MasterConfigurationFileOptionState Table

`MasterConfigurationFileOptionState` is a static table storing the usage option of the inventory agents settings file.

Table 26: Database columns for `MasterConfigurationFileOptionState` table

Database Column	Details
`MasterConfigurationFileOptionStateID`	*Type:* integer. Key. Generated ID A unique identifier for each `MasterConfigurationFileOptionState`. Possible values and the corresponding default strings are: • 1 = Always download latest version • 2 = Never download • 3 = Only download authorized version.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the culture type.
`DefaultValue`	*Type:* text (max 256 characters) The text description used to show what has changed in an update for the MasterConfigurationFile dataset.

MasterConfigurationFileSelection Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 27: Database columns for `MasterConfigurationFileSelection` table

Database Column	Details
`MasterConfiguration FileSelectionID`	*Type:* integer. Key. Generated ID A unique identifier for the insert type.
`Revision`	*Type:* integer. Nullable The revision of MasterConfirgurationFile that was selected for this tenant.
`UpdatedBy`	*Type:* text (max 400 characters) The compliance user/administrator who updated this data last.
`UpdatedDate`	*Type:* datetime The date and time the record was last updated.
`MasterConfiguration FileOptionStateID`	*Type:* integer The specified selection of the type of update allowed for this tenant.

MasterConfigurationFileVersions Table

Table 28: Database columns for `MasterConfigurationFileVersions` table

Database Column	Details
`MasterConfiguration FileVersionsID`	*Type:* integer. Key. Generated ID A unique identifier for the insert type.
`ConfigurationFileTypeID`	*Type:* integer. Key The configuration file type. Foreign key to the `ConfigurationFileType` table.
`Name`	*Type:* text (max 100 characters) Name column originates in MasterConfigurationFile table.
`Revision`	*Type:* integer. Key Revision column originates in MasterConfigurationFile table.
`XMLFile`	*Type:* text XMLFile column originates in MasterConfigurationFile table.

Database Column	Details
`Depricated`	*Type:* boolean Indicates when a file has been depricated.
`MasterConfiguration FileDescriptionID`	*Type:* integer. Key. Nullable The text description used to show what has changed in an update for the MasterConfigurationFile dataset.
`DateCreated`	*Type:* datetime The date and time the current record was created.

OperatorTenantSetting Table

`OperatorTenantSetting` is a multi-tenant table that stores configuration and preferences for each operator per tenant

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 29: Database columns for `OperatorTenantSetting` table

Database Column	Details
`OperatorTenantSettingID`	*Type:* integer. Key. Generated ID Unique identifier of an operator tenant setting, this is a primary key.
`ComplianceOperatorID`	*Type:* integer. Key The operator of the setting. Foreign key to the `ComplianceOperator` table.
`SettingNameID`	*Type:* integer. Key ID of the setting name. Foreign key to the `SettingName` table.
`SettingValue`	*Type:* text (max 512 characters). Nullable Value of the setting.

ResourceStringCultureType Table

The `ResourceStringCultureType` table holds all translations of all the resource strings.

Table 30: Database columns for `ResourceStringCultureType` table

Database Column	Details
`ResourceString`	*Type:* text (max 256 characters). Key A unique identifier for a resource string. Foreign key to the `ComplianceResourceString` table.
`CultureType`	*Type:* text (max 12 characters). Key A unique identifier for a culture type. Foreign key to the `ComplianceCultureType` table.
`ResourceValue`	*Type:* text (max 1000 characters) A translated resource string.

RightDefinition Table

`RightDefinition` defines additional access rights that supplement the built-in rights.

Table 31: Database columns for `RightDefinition` table

Database Column	Details
`RightDefinitionID`	*Type:* integer. Key. Generated ID A unique identifier for a right definition.
`ResourceName`	*Type:* text (max 16 characters). Key Resource (such as inventory, usage tracking, and so on) that access right relates to. Foreign key to the `Resource` table.
`ActionClassName`	*Type:* text (max 16 characters). Key Action class (such as modify, read, and so on) of access right. Foreign key to the `ActionClass` table.
`ParentFeature`	*Type:* text (max 50 characters) The product feature to which this access right applies.
`Title`	*Type:* text (max 1000 characters) Default value for access right title.
`TitleResourceString`	*Type:* text (max 256 characters). Key. Nullable The unique name of the localizable resource string representing an access right. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
MinAccessType	*Type:* text (max 50 characters). Nullable Minimum access type that allows this right. Possible values include NoAccess, ReadOnlyAccess, NormalAccess, AdministratorAccess and CustomAccess.
DisplayIndex	*Type:* integer. Nullable Order in which rights are displayed (smaller numbers are displayed first). FlexNet Manager Suite built-in rights have the value 100.

SettingName Table

SettingName is a static table containing ids of setting names that are referenced by ComplianceTenantSetting and OperatorTenantSetting tables.

Table 32: Database columns for SettingName table

Database Column	Details
SettingNameID	*Type:* integer. Key. Generated ID A unique identifier for a setting name.
Name	*Type:* text (max 128 characters). Key Name of the setting.

TimezoneType Table

This table stores a collection of timezonetypes.

Table 33: Database columns for TimezoneType table

Database Column	Details
TimezoneTypeID	*Type:* integer. Key. Generated ID Unique identifier for a TimezoneType.
TimezoneID	*Type:* text (max 128 characters) The .NET representation of the time zone id.
ResourceName	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a timezone type. Foreign key to the ComplianceResourceString table.

Database Column	Details
`DefaultValue`	*Type:* text (max 256 characters) The default display timezone name

Compliance.Logic.Assets Tables

The complete set of database tables documented here includes:

- AcquisitionMode table (see AcquisitionMode Table)
- Asset table (see Asset Table)
- AssetComplianceColumn table (see AssetComplianceColumn Table)
- AssetComplianceStatus table (see AssetComplianceStatus Table)
- AssetContract table (see AssetContract Table)
- AssetPropertyValue table (see AssetPropertyValue Table)
- AssetPurchaseOrder table (see AssetPurchaseOrder Table)
- AssetStatus table (see AssetStatus Table)
- AssetType table (see AssetType Table)
- AssetTypeProperty table (see AssetTypeProperty Table)
- AssetWarrantyType table (see AssetWarrantyType Table)
- DepreciationMethod table (see DepreciationMethod Table)
- EndOfLifeReason table (see EndOfLifeReason Table)
- LeaseEndReason table (see LeaseEndReason Table)

AcquisitionMode Table

`AcquisitionMode` is a static table listing all the methods by which a company may obtain an asset.

Table 34: Database columns for `AcquisitionMode` table

Database Column	Details
`AcquisitionModeID`	*Type:* integer. Key. Generated ID A unique identifier for each `AcquisitionMode`. Possible values and the corresponding default strings are: • 1 = Purchased • 2 = Leased • 3 = Rented • 4 = Loaned.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an acquisition mode. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the mode resource string has no translation.

Asset Table

The `Asset` table contains details of all the assets being managed within FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 35: Database columns for `Asset` table

Database Column	Details
`AssetID`	*Type:* integer. Key. Generated ID A unique identifier for an asset.
`ParentAssetID`	*Type:* integer. Key. Nullable The parent asset. Foreign key to another asset in this same `Asset` table.
`ShortDescription`	*Type:* text (max 256 characters) A brief description of the asset.
`SerialNumber`	*Type:* text (max 150 characters). Key. Nullable The serial number of the asset.

Database Column	Details
AssetTypeID	*Type:* integer. Key The asset type. Foreign key to the AssetType table.
AssetTag	*Type:* text (max 256 characters). Nullable A user-defined asset tag for a particular asset. This may be a barcode number.
AssetStatusID	*Type:* integer. Key The status of the asset. Defaults to Purchased. Foreign key to the AssetStatus table.
PurchasePrice	*Type:* currency. Nullable The purchase price of the asset.
PurchasePriceRateID	*Type:* integer. Nullable The currency rate to apply to the purchase price of the asset. Foreign key to the CurrencyRate table.
AcquisitionModeID	*Type:* integer. Nullable The method of acquisition used for the asset. Defaults to Purchased. Foreign key to the AcquisitionMode table.
PrimaryPurchaseOrderNo	*Type:* text (max 50 characters). Nullable The purchase order number which was used to purchase the asset.
PrimaryPurchaseOrderDate	*Type:* datetime. Nullable The date the primary purchase order was made.
VendorID	*Type:* integer. Key. Nullable The vendor from whom the asset was purchased. Foreign key to the Vendor table.
Manufacturer	*Type:* text (max 200 characters). Nullable The manufacturer of the asset.
ManufacturerPartNo	*Type:* text (max 100 characters). Nullable The manufacturer's part number for this asset.
ModelNo	*Type:* text (max 200 characters). Nullable The model number of the asset.
DeliveryDate	*Type:* datetime. Nullable The date the asset was received.
AssetWarrantyTypeID	*Type:* integer The type of warranty for the asset. Defaults to None. Foreign key to the AssetWarrantyType table.

Database Column	Details
`WarrantyExpirationDate`	*Type:* datetime. Nullable The date the warranty expires.
`InstallationDate`	*Type:* datetime. Nullable The date the asset was installed.
`RetirementDate`	*Type:* datetime. Nullable The date the asset was retired.
`DisposalDate`	*Type:* datetime. Nullable The date the asset was disposed of.
`DeletionDate`	*Type:* datetime. Nullable The date the asset was deleted.
`InventoryDate`	*Type:* datetime. Nullable The date the asset last had inventory reported.
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory.
`InventoryDateManual`	*Type:* datetime. Nullable The date the asset last had inventory updated (entered) manually.
`InventoryAgentManual`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last manual inventory.
`RequestNo`	*Type:* text (max 60 characters). Nullable The request number for the asset.
`PartNo`	*Type:* text (max 100 characters). Nullable The vendor's part number for this asset.
`IsLeased`	*Type:* boolean Flag to indicate if this asset is leased. This field is no longer in use in FlexNet Manager Suite.
`LeaseNo`	*Type:* text (max 60 characters). Nullable The contract number of the lease agreement for this asset.
`LeaseName`	*Type:* text (max 100 characters). Nullable A contract name of the lease agreement for this asset.
`LeaseStartDate`	*Type:* datetime. Nullable The start date of the lease for this asset.

Database Column	Details
`LeaseEndDate`	*Type:* datetime. Nullable The end date of the lease for this asset.
`LeaseTerminationDate`	*Type:* datetime. Nullable The date that the lease for this asset is terminated.
`LeaseEndReasonID`	*Type:* integer The reason for the end of lease for this asset.
`LeasePrice`	*Type:* currency. Nullable The purchase price of the lease for this individual asset.
`LeasePriceRateID`	*Type:* integer. Nullable The purchase price of the lease currency rate for this individual asset.
`LeasePeriodicPayment`	*Type:* currency. Nullable The price of periodic payments associated with this contract.
`LeasePeriodicPayment RateID`	*Type:* integer. Nullable The price of periodic payments currency rate associated with this contract.
`LeasePeriodTypeID`	*Type:* integer The frequency with which the lease payments are applicable.
`LeaseBuyoutCost`	*Type:* currency. Nullable The buyout cost of the lease for this asset.
`LeaseBuyoutCostRateID`	*Type:* integer. Nullable The buyout cost of the lease currency rate associated for this asset.
`LeaseComments`	*Type:* text. Nullable Comments recorded about the lease for this asset. This field is no longer in use in FlexNet Manager Suite.
`AssignToUserID`	*Type:* integer. Key. Nullable The end-user the asset has been assigned to. Foreign key to the `ComplianceUser` table.
`Comments`	*Type:* text. Nullable Comments entered about the asset.
`ChargeBackPrice`	*Type:* currency. Nullable Amount to be charged back for the use of this asset. No calculations based on this charge and the frequency will be provided.

Database Column	Details
ChargeBackPriceRateID	*Type:* integer. Nullable The currency rate to be applied to the charge back value of the asset. Foreign key to the CurrencyRate table.
ChargeBackPeriodTypeID	*Type:* integer The frequency with which the charge back price is charged. Defaults to None. Foreign key to the PeriodType table.
EndOfLifeRecipient	*Type:* text (max 128 characters). Nullable The person or organization who received the asset when it was disposed of.
EndOfLifeReasonID	*Type:* integer The reason the asset was disposed of. Foreign key to the EndOfLifeReason table.
ResalePrice	*Type:* currency. Nullable The amount the asset was sold for.
ResalePriceRateID	*Type:* integer. Nullable The currency rate to be applied to the resale price of the asset. Foreign key to the CurrencyRate table.
CreationUser	*Type:* text (max 128 characters). Nullable The operator who created the record.
CreationDate	*Type:* datetime The date the record was created.
UpdatedUser	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
UpdatedDate	*Type:* datetime. Nullable The date the record was last updated.
LocationID	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this asset. Foreign key to the GroupEx table.
BusinessUnitID	*Type:* text (max 128 characters). Key. Nullable Any corporate unit in the enterprise associated with this asset. Foreign key to the GroupEx table.
CostCenterID	*Type:* text (max 128 characters). Key. Nullable Any cost center in the enterprise associated with this asset. Foreign key to the GroupEx table.

Database Column	Details
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise category associated with this asset. Foreign key to the `GroupEx` table.
`DepreciationCurrentValue`	*Type:* currency. Nullable The current value of the asset, after depreciation has been applied.
`DepreciationCurrent ValueRateID`	*Type:* integer. Nullable The currency rate to be applied to the depreciation current value of the asset. Foreign key to the `CurrencyRate` table.
`DepreciationResidualValue`	*Type:* currency. Nullable The residual value of the asset (value when fully depreciated).
`DepreciationResidual ValueRateID`	*Type:* integer. Nullable The currency rate to be applied to the residual value of the asset. Foreign key to the `CurrencyRate` table.
`DepreciationMethodID`	*Type:* integer. Nullable The depreciation method (straight line or residual value). Foreign key to the `DepreciationMethod` table.
`DepreciationPeriod`	*Type:* integer The depreciation period (in years), for customers to use for straight line depreciation.
`DepreciationRate`	*Type:* decimal. Nullable The annual depreciation rate (as a percentage - like 50% per year), for customers to use for residual value depreciation. Stored as a value between 0 (for 0%) and 1 (for 100%).
`WrittenOffValue`	*Type:* currency. Nullable The written-off value is the value of the asset at the time of retirement/disposal.
`WrittenOffValueRateID`	*Type:* integer. Nullable The currency rate to be applied to the written-off value of the asset. Foreign key to the `CurrencyRate` table.

AssetComplianceColumn Table

The `AssetComplianceColumn` table lists the columns (or aspects of the asset record) for which compliance changes can be tracked.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 36: Database columns for `AssetComplianceColumn` table

Database Column	Details
`AssetComplianceColumnID`	*Type:* integer. Key. Generated ID A unique identifier for each `AssetComplianceColumn`. Possible values and the corresponding default strings are: • 1 = Operating System • 2 = Service Pack • 3 = Number of Processors • 4 = Processor Type • 5 = Max Clock Speed • 6 = Total Memory • 7 = Chassis Type • 8 = Number of Hard Drives • 9 = Total Disk Size • 10 = Number of Network Cards • 11 = Number of Display Adapters • 12 = IP Address • 13 = MAC Address • 14 = Host • 15 = Number of Cores • 16 = Number of Threads.
`ColumnNameResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a compliance-tracked column. Foreign key to the `ComplianceResourceString` table.
`ColumnName`	*Type:* text (max 128 characters). Key The text to display if the column resource string has no translation.
`IsColumnNumeric`	*Type:* boolean Indicates whether the column is numeric (`True`) or a string (`False`).
`ComplianceAction`	*Type:* integer Bitwise value to indicate what type of action to track change on.

Database Column	Details
`TrackComplianceBitwise Value`	*Type:* integer Bitwise value indicating which asset types compliance tracking is turned on for.

AssetComplianceStatus Table

`AssetComplianceStatus` is a static table listing possible asset compliance states, such as compliant, new, changed, or ignored.

Table 37: Database columns for `AssetComplianceStatus` table

Database Column	Details
`AssetComplianceStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `AssetComplianceStatus`. Possible values and the corresponding default strings are: • 1 = New • 2 = Compliant • 3 = Changed • 4 = Ignore.
`StatusResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an asset compliance status. Foreign key to the `ComplianceResourceString` table.
`StatusDefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

AssetContract Table

The `AssetContract` table links assets to related contracts.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 38: Database columns for `AssetContract` table

Database Column	Details
`AssetContractID`	*Type:* integer. Key. Generated ID A unique identifier for this record.
`ContractID`	*Type:* integer. Key The contract linked to the asset. Foreign key to the `Contract` table.
`AssetID`	*Type:* integer. Key The asset linked to the contract. Foreign key to the `Asset` table.

AssetPropertyValue Table

For each asset, `AssetPropertyValue` stores the values for the custom properties defined in `AssetTypeProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 39: Database columns for `AssetPropertyValue` table

Database Column	Details
`AssetPropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for this record.
`AssetTypePropertyID`	*Type:* integer. Key The property whose value is being stored. The type of the asset should match the type that the property is associated with. Foreign key to the `AssetTypeProperty` table.
`AssetID`	*Type:* integer. Key The asset associated with the property value. Foreign key to the `Asset` table.
`PropertyValue`	*Type:* text (max 4000 characters) The value of the property for the specified `Asset`.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.

Database Column	Details
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

AssetPurchaseOrder Table

The `AssetPurchaseOrder` table links assets to related purchase order lines.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 40: Database columns for `AssetPurchaseOrder` table

Database Column	Details
`AssetID`	*Type:* integer. Key The asset linked to a purchase order. Foreign key to the `Asset` table.
`PurchaseOrderDetailID`	*Type:* integer. Key The purchase order line linked to an asset. Foreign key to the `PurchaseOrderDetail` table.

AssetStatus Table

`AssetStatus` is a static table storing a list of possible asset states, such as purchased, in storage, installed, retired, disposed and other.

Table 41: Database columns for `AssetStatus` table

Database Column	Details
`AssetStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `AssetStatus`. Possible values and the corresponding default strings are: • 1 = Purchased • 2 = In Storage • 3 = Installed • 4 = Retired • 5 = Disposed • 6 = Other.
`StatusResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an asset status. Foreign key to the `ComplianceResourceString` table.
`StatusDefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

AssetType Table

`AssetType` stores the collection of the types of assets that can be created in FlexNet Manager Suite.

Table 42: Database columns for `AssetType` table

Database Column	Details
`AssetTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `AssetType`. Possible values and the corresponding default strings are: • 1 = Workstation • 2 = Server • 3 = Monitor • 4 = Desk • 5 = Chair • 6 = Printer • 7 = Router • 8 = Switch • 9 = Telephone • 10 = Cell phone • 11 = Laptop. • 12 = Mobile Device.
`AssetTypeResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a document type. Foreign key to the `ComplianceResourceString` table.
`AssetTypeName`	*Type:* text (max 64 characters). Key The text to display if the type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of asset, stored in XML format.
`ParentAssetTypeID`	*Type:* integer. Nullable An asset type which is a parent of this asset type. Foreign key to the same `AssetType` table.
`ManagedType`	*Type:* boolean. Key Set this field to `True` if this type of asset is directly managed by FlexNet Manager Suite (for example, laptops, servers and workstations).
`BitwiseValue`	*Type:* integer The bitwise value of the asset type. This value is used when tracking compliance changes for assets linked to computers.

AssetTypeProperty Table

`AssetTypeProperty` defines extra custom properties for all assets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 43: Database columns for `AssetTypeProperty` table

Database Column	Details
`AssetTypePropertyID`	*Type:* integer. Key. Generated ID A unique identifier for a property of an asset type.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.
`AssetTypeID`	*Type:* integer. Key. Nullable Asset type with which this property is associated. Foreign key to the `AssetType` table.
`HardwareClassName`	*Type:* text (max 256 characters). Nullable The WMI class name associated with this property. This field applies for hardware properties that are mapped to hardware inventory tables.
`HardwarePropertyName`	*Type:* text (max 256 characters) The WMI property name associated with this property. This field applies for hardware properties that are mapped to hardware inventory tables.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

AssetWarrantyType Table

`AssetWarrantyType` is a static table listing all the types of warranties.

Table 44: Database columns for `AssetWarrantyType` table

Database Column	Details
`AssetWarrantyTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `AssetWarrantyType`. Possible values and the corresponding default strings are: • 1 = None • 2 = One year on site • 3 = Three years on site.
`WarrantyTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an asset warranty type. Foreign key to the `ComplianceResourceString` table.
`WarrantyTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

DepreciationMethod Table

`DepreciationMethod` is a static table storing the collection of available depreciation methods.

Table 45: Database columns for `DepreciationMethod` table

Database Column	Details
`DepreciationMethodID`	*Type:* integer. Key. Generated ID A unique identifier for each `DepreciationMethod`. Possible values and the corresponding default strings are: • 1 = Straight line • 2 = Residual value.
`ResourceName`	*Type:* text (max 50 characters). Key The unique name of the localizable resource string representing a depreciation method. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the method resource string has no translation.

EndOfLifeReason Table

`EndOfLifeReason` is a static table storing the collection of all reasons for disposing of an asset.

Table 46: Database columns for `EndOfLifeReason` table

Database Column	Details
`EndOfLifeReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `EndOfLifeReason`. Possible values and the corresponding default strings are: • 1 = [empty string] • 2 = Lost • 3 = Stolen • 4 = Disposed • 5 = Sold • 6 = Donated • 7 = Broken.
`ResourceName`	*Type:* text (max 50 characters). Key The unique name of the localizable resource string representing an end-of-life reason. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the reason resource string has no translation.

LeaseEndReason Table

`LeaseEndReason` is a static table listing all the reasons that a company terminates a lease.

Table 47: Database columns for `LeaseEndReason` table

Database Column	Details
`LeaseEndReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `LeaseEndReason`. Possible values and the corresponding default strings are: • 1 = [empty string] • 2 = Lease Ended - Asset Returned • 3 = Early Termination - Asset Returned • 4 = Buyout • 5 = Early Buyout • 6 = Trade.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a lease-end reason. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the reason resource string has no translation.

Compliance.Logic.Beacon Tables

The complete set of database tables documented here includes:

- ActiveDirectoryComputer table (see ActiveDirectoryComputer Table)
- ActiveDirectoryDomain table (see ActiveDirectoryDomain Table)
- ActiveDirectoryGroup table (see ActiveDirectoryGroup Table)
- ActiveDirectoryMember table (see ActiveDirectoryMember Table)
- ActiveDirectoryUser table (see ActiveDirectoryUser Table)
- AdministrationAccount table (see AdministrationAccount Table)
- AppVPackageMapping table (see AppVPackageMapping Table)
- AvailablePackage table (see AvailablePackage Table)
- AvailablePackageType table (see AvailablePackageType Table)
- BaselineImport table (see BaselineImport Table)
- Beacon table (see Beacon Table)
- BeaconActivityStatus table (see BeaconActivityStatus Table)

- BeaconAdministrationAccount table (see BeaconAdministrationAccount Table)
- BeaconAgentEvent table (see BeaconAgentEvent Table)
- BeaconDiscoveryStatus table (see BeaconDiscoveryStatus Table)
- BeaconDiscoveryTaskSummaryStatus table (see BeaconDiscoveryTaskSummaryStatus Table)
- BeaconDownloadedPolicy table (see BeaconDownloadedPolicy Table)
- BeaconExecutionStatusType table (see BeaconExecutionStatusType Table)
- BeaconFilter table (see BeaconFilter Table)
- BeaconGroup table (see BeaconGroup Table)
- BeaconGroupBeaconMapping table (see BeaconGroupBeaconMapping Table)
- BeaconGroupBeaconTargetMapping table (see BeaconGroupBeaconTargetMapping Table)
- BeaconIssueStatus table (see BeaconIssueStatus Table)
- BeaconIssueStatusType table (see BeaconIssueStatusType Table)
- BeaconPolicy table (see BeaconPolicy Table)
- BeaconPolicyPropertyValue table (see BeaconPolicyPropertyValue Table)
- BeaconPropertyValue table (see BeaconPropertyValue Table)
- BeaconRule table (see BeaconRule Table)
- BeaconRuleAction table (see BeaconRuleAction Table)
- BeaconRuleActionPropertyValue table (see BeaconRuleActionPropertyValue Table)
- BeaconRuleBeaconTargetMapping table (see BeaconRuleBeaconTargetMapping Table)
- BeaconSiteSubnetMapping table (see BeaconSiteSubnetMapping Table)
- BeaconTarget table (see BeaconTarget Table)
- BeaconTargetAgentEvent table (see BeaconTargetAgentEvent Table)
- BeaconTargetDiscoveredDeviceMapping table (see BeaconTargetDiscoveredDeviceMapping Table)
- BeaconTargetPropertyValue table (see BeaconTargetPropertyValue Table)
- BeaconTargetSiteMapping table (see BeaconTargetSiteMapping Table)
- BeaconTargetSiteSubnetMapping table (see BeaconTargetSiteSubnetMapping Table)
- BeaconUpgradeMode table (see BeaconUpgradeMode Table)
- BeaconUpgradeStatus table (see BeaconUpgradeStatus Table)
- BeaconWebServerStatus table (see BeaconWebServerStatus Table)
- DiscoveredDeviceDiscoveredBy table (see DiscoveredDeviceDiscoveredBy Table)

- DiscoveredDeviceDiscoveryStatus table (see DiscoveredDeviceDiscoveryStatus Table)
- DiscoveredDeviceInventoryStatus table (see DiscoveredDeviceInventoryStatus Table)
- DiscoveredDeviceTaskDetailedError table (see DiscoveredDeviceTaskDetailedError Table)
- DiscoveredDeviceTaskStatus table (see DiscoveredDeviceTaskStatus Table)
- DiscoveredDeviceTaskStatusHistory table (see DiscoveredDeviceTaskStatusHistory Table)
- DiscoveredDeviceTaskType table (see DiscoveredDeviceTaskType Table)
- ErrorCategory table (see ErrorCategory Table)
- FNMEAAgent table (see FNMEAAgent Table)
- IncomingBaseline table (see IncomingBaseline Table)
- ReconcileSoftwareLicenseReconcileExemptionReason table (see ReconcileSoftwareLicenseReconcileExemptionReason Table)
- RuleDiscoveryActionSummary table (see RuleDiscoveryActionSummary Table)
- RuleInventoryActionSummary table (see RuleInventoryActionSummary Table)
- SoftwareLicenseReconcileExemptionReasonData table (see SoftwareLicenseReconcileExemptionReasonData Table)
- StatusCodeCategory table (see StatusCodeCategory Table)
- UIAlignmentType table (see UIAlignmentType Table)
- UIFieldType table (see UIFieldType Table)
- UIInsertType table (see UIInsertType Table)
- UIItem table (see UIItem Table)
- UIItemTargetSubType table (see UIItemTargetSubType Table)

ActiveDirectoryComputer Table

The `ActiveDirectoryComputer` table stores the active directory data for computers.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 48: Database columns for `ActiveDirectoryComputer` table

Database Column	Details
`ActiveDirectoryComputerID`	*Type:* integer. Key. Generated ID Auto-generated Active Directory computer ID

Database Column	Details
`GUID`	*Type:* unique identifier. Key The GUID of the computer.
`ComputerName`	*Type:* text (max 64 characters) The computer name.
`ActiveDirectoryDomainID`	*Type:* integer. Key Foreign key to the `ActiveDirectoryDomain` table
`SID`	*Type:* text (max 256 characters). Key. Nullable The SID of the computer.

ActiveDirectoryDomain Table

The `ActiveDirectoryDomain` table stores the active directory domains.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 49: Database columns for `ActiveDirectoryDomain` table

Database Column	Details
`ActiveDirectoryDomainID`	*Type:* integer. Key. Generated ID Auto-generated Active Directory Domain ID
`QualifiedName`	*Type:* text (max 100 characters). Key The fully qualified domain name
`FlatName`	*Type:* text (max 32 characters) The domain flat name

ActiveDirectoryGroup Table

The `ActiveDirectoryGroup` table stores the active directory data.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 50: Database columns for `ActiveDirectoryGroup` table

Database Column	Details
`ActiveDirectoryGroupID`	*Type:* integer. Key. Generated ID Auto-generated Active Directory Group ID
`GUID`	*Type:* unique identifier. Key The GUID of the AD group.
`SID`	*Type:* text (max 256 characters). Key. Nullable The SID of the AD group.
`Name`	*Type:* text (max 128 characters). Nullable The AD group name
`ActiveDirectoryDomainID`	*Type:* integer. Key Foreign key to the `ActiveDirectoryDomain` table

ActiveDirectoryMember Table

The `ActiveDirectoryMember` table stores the active directory data for AD member objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 51: Database columns for `ActiveDirectoryMember` table

Database Column	Details
`GUID`	*Type:* unique identifier. Key The GUID of the member object.
`ParentGroupGUID`	*Type:* unique identifier. Key The parent AD group GUID.

ActiveDirectoryUser Table

The `ActiveDirectoryUser` table stores the active directory data for users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 52: Database columns for `ActiveDirectoryUser` table

Database Column	Details
`ActiveDirectoryUserID`	*Type:* integer. Key. Generated ID Auto-generated Active Directory user ID
`GUID`	*Type:* unique identifier. Key The GUID of the user.
`SAMAccountName`	*Type:* text (max 20 characters). Key The user name.
`ActiveDirectoryDomainID`	*Type:* integer. Key Foreign key to the `ActiveDirectoryDomain` table
`Sid`	*Type:* text (max 256 characters). Key. Nullable The SID of the user.

AdministrationAccount Table

Records the complete set of administration accounts configured on inventory beacons.

Table 53: Database columns for `AdministrationAccount` table

Database Column	Details
`AccountID`	*Type:* integer. Key. Generated ID Unique id for the account.
`AccountName`	*Type:* text (max 256 characters). Key The logical name of the account.

AppVPackageMapping Table

The `AppVPackageMapping` table is a table that maps App-V 4.6 packages to installer evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 54: Database columns for `AppVPackageMapping` table

Database Column	Details
`AppVPackageMappingID`	*Type:* integer. Key. Generated ID Auto-generated App-V 4.6 package mapping ID.
`PackageName`	*Type:* text (max 256 characters). Key The App-V 4.6 package name.
`PackageVersion`	*Type:* text (max 128 characters). Key The App-V 4.6 package version.
`DisplayName`	*Type:* text (max 256 characters) The display name of the software as reported by the installer evidence.
`Version`	*Type:* text (max 72 characters) The version of the software as reported by the installer evidence.
`Publisher`	*Type:* text (max 200 characters) The publisher of the software as reported by the installer evidence.

AvailablePackage Table

Packages which are available to beacons.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 55: Database columns for `AvailablePackage` table

Database Column	Details
`AvailablePackageID`	*Type:* integer. Key. Generated ID The ID of the available package.
`FullName`	*Type:* text (max 256 characters). Key The full path of the package within the repository.
`Version`	*Type:* text (max 32 characters). Key The version of the package.
`AvailablePackageTypeID`	*Type:* integer. Key The type of the package. Foreign key to the `AvailablePackageType` table.

Database Column	Details
`RelativeURLToOSD`	*Type:* text (max 256 characters) The relative URL to the OSD of the package for use in inventory agent policy.
`UseInAgentPolicy`	*Type:* boolean Whether the package should be added to policy for inventory agents.
`Build`	*Type:* text (max 8 characters). Key The build number of the package, necessary for choosing between patched versions of the same release.
`WebUIRelativeURL`	*Type:* text (max 256 characters). Nullable The relative URL to download the package from WebUI
`IsDeprecated`	*Type:* boolean Whether the package is no longer available for use or not.
`DeprecationKBName`	*Type:* text (max 256 characters). Nullable Name of KB article which explains deprecation.
`DeprecationKBLink`	*Type:* text. Nullable Link to kb article which explains deprecation.

AvailablePackageType Table

Table 56: Database columns for `AvailablePackageType` table

Database Column	Details
`AvailablePackageTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `AvailablePackageType`. Possible values are: • 1 = Adoption • 2 = Upgrade • 3 = Inventory agent plugin • 4 = Software • 5 = Other • 6 = Inventory beacon upgrade
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a purchase order line item type. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

BaselineImport Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 57: Database columns for `BaselineImport` table

Database Column	Details
BaselineImportID	*Type:* integer. Key. Generated ID The baseline import ID
Type	*Type:* text (max 16 characters) The baseline type
Date	*Type:* datetime The date of the baseline import
PurchaseOrderID	*Type:* integer. Key. Nullable The purchase order for the baseline import
ComplianceOperatorID	*Type:* integer. Key The compliance operator who performed the baseline import

Beacon Table

The `Beacon` table contains beacon definition.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 58: Database columns for `Beacon` table

Database Column	Details
BeaconID	*Type:* integer. Key. Generated ID Unqiue ID assigned to each beacon.

Database Column	Details
`BeaconUID`	*Type:* unique identifier. Key Unique string ID of the beacon.
`BeaconName`	*Type:* text (max 64 characters) Name of the beacon.
`BeaconDescription`	*Type:* text (max 256 characters). Nullable Description of the beacon.
`BeaconStatus`	*Type:* boolean Boolean indicating to include or exclude Beacon.
`LastKnownActivityTime`	*Type:* datetime. Nullable Last known time that communication has been received from the beacon.
`ActivityStatusID`	*Type:* integer Last known activity status reported by the beacon.
`PolicyDownloadedTime`	*Type:* datetime. Nullable Policy downloaded time
`CurrentPolicyRevisionNo`	*Type:* integer. Nullable Last downloaded policy revision number
`BeaconLocation`	*Type:* text (max 256 characters). Nullable Location field for Beacon.
`PrimaryParentUID`	*Type:* unique identifier. Nullable The parent of the Beacon. For the core Beacon, the PrimaryParentUID is NULL.
`BeaconPassword`	*Type:* text (max 64 characters). Nullable The password used by the beacon to authenticate with.
`HTTPAccessData`	*Type:* text. Nullable The HTTPEndPointStatus object, used for storing a summary of how to access the sahres on this beacon.
`UpgradeModeID`	*Type:* integer The upgrade mode selected for this beacon.
`UpgradeStatusID`	*Type:* integer The latest information reported by a beacon about any upgrade activity or changes.
`LastKnownPolicy`	*Type:* datetime. Nullable The last known time that the beacon has communicated with the server.

Database Column	Details
`Version`	*Type:* text (max 50 characters). Nullable Version of installed beacon on the server
`WebServerStatusID`	*Type:* integer The last known time that the beacon has communicated with the server.
`UpgradeStatusTime`	*Type:* datetime. Nullable The time the last upgrade status was reported.
`AvailablePackageID`	*Type:* integer. Key. Nullable If the beacon upgrade mode is set to specific version, then this stored the specific package to upgrade to.
`ParentServerURL`	*Type:* text. Nullable The parent to which this beacon will communicate with.
`DownloadURL`	*Type:* text. Nullable The download URL of the parent.
`UploadURL`	*Type:* text. Nullable The upload URL of the parent.

BeaconActivityStatus Table

`BeaconActivityStatus` is a static table listing all of the states of a beacon.

Table 59: Database columns for `BeaconActivityStatus` table

Database Column	Details
`BeaconActivityStatusID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the BeaconActivityStatus record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

BeaconAdministrationAccount Table

Records an administration account discovered on an inventory beacon.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 60: Database columns for `BeaconAdministrationAccount` table

Database Column	Details
`AccountID`	*Type:* integer. Key Unique id for the account.
`ServerUID`	*Type:* unique identifier. Key Identifies the distribution server which discovered the account.

BeaconAgentEvent Table

The `BeaconAgentEvent` table contains a list of events that can be included in agent schedules.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 61: Database columns for `BeaconAgentEvent` table

Database Column	Details
`BeaconAgentEventID`	*Type:* integer. Key. Generated ID Unqiue ID assigned to each beacon agent event.
`EventName`	*Type:* text (max 256 characters). Key Event name.
`EventUID`	*Type:* unique identifier. Key Event uid.
`Value`	*Type:* text An XML representation of the agent event data.

BeaconDiscoveryStatus Table

Discovery and remote execution status of Beacon

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 62: Database columns for `BeaconDiscoveryStatus` table

Database Column	Details
`BeaconDiscoveryStatusID`	*Type:* integer. Key. Generated ID The ID of the beacon discovery status.
`ServerUID`	*Type:* unique identifier. Key The inventory beacon that has run the task.
`State`	*Type:* text (max 256 characters) State of the discovery/execution - Running/Finished.
`StartDateTime`	*Type:* datetime Execution start time.
`Duration`	*Type:* integer Duration in Seconds of the discovery execution.
`DiscoveredCount`	*Type:* integer Total number of devices discovered.
`ExecutionSuccess`	*Type:* integer Total number successful remote executions.
`ExecutionFailure`	*Type:* integer Total number failed remote executions.

BeaconDiscoveryTaskSummaryStatus Table

Task summary list for a particular beacon

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 63: Database columns for `BeaconDiscoveryTaskSummaryStatus` table

Database Column	Details
`TaskSummaryStatusID`	*Type:* integer. Key. Generated ID The ID of the device status.
`BeaconDiscoveryStatusID`	*Type:* integer. Key The beacon discovery status table which this refers to.

Database Column	Details
`TaskTypeID`	*Type:* integer The type of task which was run.
`SuccessCount`	*Type:* integer Success count in this particular execution.
`FailureCount`	*Type:* integer Failure count in this particular execution.

BeaconDownloadedPolicy Table

The `BeaconDownloadedPolicy` table contains policies downloaded by inventory beacons.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 64: Database columns for `BeaconDownloadedPolicy` table

Database Column	Details
`BeaconDownloadedPolicyID`	*Type:* integer. Key. Generated ID The ID of the downloaded beacon policy.
`RevisionNumber`	*Type:* integer. Key The revision number of this policy.
`PolicyXML`	*Type:* XML The beacon policy xml downloaded by inventory beacons.

BeaconExecutionStatusType Table

`BeaconExecutionStatusType` is a static table listing possible beacon status values.

Table 65: Database columns for `BeaconExecutionStatusType` table

Database Column	Details
`BeaconExecutionStatusTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `BeaconExecutionStatusType`. Possible values and the corresponding default strings are: • 1 = Unknown • 2 = Started • 3 = Not configured • 4 = Running • 5 = Finished • 6 = Stopped
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a batch process type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

BeaconFilter Table

The `BeaconFilter` table contains target filters.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 66: Database columns for `BeaconFilter` table

Database Column	Details
`BeaconFilterID`	*Type:* integer. Key. Generated ID Unqiue ID automatically assigned to each beacon target filters.
`BeaconTargetID`	*Type:* integer. Key Target this filter refers to.
`Include`	*Type:* boolean Boolean string indicating to include or exclude filter value.

Database Column	Details
`IsLinked`	*Type:* boolean Boolean indicating if the filter is linked to site/subnet/device or an independent filter.
`Value`	*Type:* text (max 256 characters) Filter value.
`FilterType`	*Type:* text (max 64 characters) Filter type set for this filter.

BeaconGroup Table

The `BeaconGroup` table contains beacon groups.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 67: Database columns for `BeaconGroup` table

Database Column	Details
`BeaconGroupID`	*Type:* integer. Key. Generated ID Unqiue ID automatically assigned to each beacon group.
`Name`	*Type:* text (max 100 characters). Key Name identifying the beacon group.
`Description`	*Type:* text (max 256 characters). Nullable Name identifying the beacon group.

BeaconGroupBeaconMapping Table

The `BeaconGroupBeaconMapping` table contains beacon group mapping to beacons.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 68: Database columns for `BeaconGroupBeaconMapping` table

Database Column	Details
`BeaconGroupBeacon MappingID`	*Type:* integer. Key. Generated ID Unqiue ID automatically assigned to each beacon group mapping.
`BeaconGroupID`	*Type:* integer. Key Reference to beacon group defined in the BeaconGroup table.
`BeaconID`	*Type:* integer Reference to beacon defined in the Beacon table.

BeaconGroupBeaconTargetMapping Table

The `BeaconGroupBeaconTargetMapping` table contains beacon group target mapping to beacon targets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 69: Database columns for `BeaconGroupBeaconTargetMapping` table

Database Column	Details
`BeaconGroupBeacon TargetMappingID`	*Type:* integer. Key. Generated ID Unqiue ID automatically assigned to each beacon group target mapping.
`BeaconGroupID`	*Type:* integer. Key Reference to beacon group defined in the BeaconGroup table.
`BeaconTargetID`	*Type:* integer Reference to beacon target defined in the BeaconTarget table.

BeaconIssueStatus Table

Records beacon issue detail information.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 70: Database columns for `BeaconIssueStatus` table

Database Column	Details
`BeaconIssueStatusID`	*Type:* integer. Key. Generated ID Unique id for the BeaconIssueStatus.
`BeaconID`	*Type:* integer. Key Beacon that this issue status relates to
`BeaconIssueStatusTypeID`	*Type:* integer. Key Issue type
`IsActive`	*Type:* boolean Policy downloaded time
`IssueDetail`	*Type:* text. Nullable Detail information about the issue

BeaconIssueStatusType Table

`BeaconIssueStatusType` is a static table listing possible beacon alerts.

Table 71: Database columns for `BeaconIssueStatusType` table

Database Column	Details
`BeaconIssueStatusTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `BeaconIssueStatusType`. Possible values and the corresponding default strings are: • 0 = Unknown • 1 = Policy load • 2 = Policy download • 3 = Discovery execution • 4 = Action execution • 5 = Self hosted web server • 6 = Service exit • 7 = Package download • 8 = Active Directory import • 9 = SAP Inventory import • 10 = SAP recommendation set download • 11 = Beacon self upgrade • 12 = Beacon Parent Configuration
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a batch process type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

BeaconPolicy Table

The `BeaconPolicy` table contains the beacon policy.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 72: Database columns for `BeaconPolicy` table

Database Column	Details
`BeaconPolicyID`	*Type:* integer. Key. Generated ID The ID of the beacon policy.
`RevisionNumber`	*Type:* integer The revision number of this policy.
`AgentScheduleData`	*Type:* text. Nullable The Schedule object, used for storing the global schedule for managed devices.
`CreationDate`	*Type:* datetime Date and time when the policy was created.
`LastChangedOn`	*Type:* datetime. Nullable Date and time that the policy was last modified.
`ApprovedBeaconPackageID`	*Type:* integer. Key. Nullable The beacon upgrade package that has been approved by the customer. NULL indicates to stay always on the latest.
`LastDiscoveryFull ExportTime`	*Type:* datetime. Nullable The last time a discovery export was generated.
`LastDiscoveryFull ExportVersion`	*Type:* integer. Nullable The revision number of the last full discovery export.
`LastTargetRefreshTime`	*Type:* datetime. Nullable The last time special internal targets were recalculated and refreshed.

BeaconPolicyPropertyValue Table

The `BeaconPolicyPropertyValue` table contains beacon policy property value elements.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 73: Database columns for `BeaconPolicyPropertyValue` table

Database Column	Details
`BeaconPolicyPropertyID`	*Type:* integer. Key. Generated ID Unqiue ID assigned to each beacon policy property.

Database Column	Details
`KeyName`	*Type:* text (max 256 characters). Key Property Key.
`Value`	*Type:* text (max 256 characters) Property Value.

BeaconPropertyValue Table

The `BeaconPropertyValue` table contains beacon property value elements.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 74: Database columns for `BeaconPropertyValue` table

Database Column	Details
`BeaconPropertyID`	*Type:* integer. Key. Generated ID Unqiue ID assigned to each beacon property.
`BeaconID`	*Type:* integer. Key Beacon this property refers to.
`KeyName`	*Type:* text (max 256 characters). Key Property Key.
`Value`	*Type:* text (max 256 characters) Property Value.

BeaconRule Table

The `BeaconRule` table contains the details of beacon rules.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 75: Database columns for `BeaconRule` table

Database Column	Details
`BeaconRuleID`	*Type:* integer. Key. Generated ID The ID of the beacon rule.
`BeaconRuleActionID`	*Type:* integer. Key The reference of Action from the beacon rule.
`RuleName`	*Type:* text (max 128 characters) The name of the rule.
`RulePriority`	*Type:* small integer Beacon rules are prioritised according to the rule priority. Higher priority takes presedence over lower priorities.
`MaximumAge`	*Type:* integer. Nullable Maximum age of the rule before it is re-scheduled.
`ExternalID`	*Type:* unique identifier. Key The ID that exists externally.
`BeaconScheduleData`	*Type:* text The Schedule object.
`Include`	*Type:* boolean Boolean string indicating to include or exclude rule.
`Internal`	*Type:* boolean Is this rule used internally, or managed by the user.
`NameResourceName`	*Type:* text (max 256 characters). Nullable Resource for translation of Name column. Foreign key to `ComplianceResourceString` table.

BeaconRuleAction Table

The `BeaconRuleAction` table contains beacon rule action.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 76: Database columns for `BeaconRuleAction` table

Database Column	Details
`BeaconRuleActionID`	*Type:* integer. Key. Generated ID Unique ID automatically assigned to each beacon actions.
`Name`	*Type:* text (max 100 characters). Key Name of Action.
`Description`	*Type:* text (max 256 characters). Nullable Description of Action.
`NameResourceName`	*Type:* text (max 256 characters). Nullable Resource for translation of Name column. Foreign key to the `ComplianceResourceString` table.
`DescriptionResourceName`	*Type:* text (max 256 characters). Nullable Resource for translation of Description column. Foreign key to the `ComplianceResourceString` table.
`Internal`	*Type:* boolean Is this action used internally, or managed by the user.

BeaconRuleActionPropertyValue Table

The `BeaconRuleActionPropertyValue` table contains beacon action property value elements.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 77: Database columns for `BeaconRuleActionPropertyValue` table

Database Column	Details
`BeaconRuleAction PropertyID`	*Type:* integer. Key. Generated ID Unqiue ID assigned to each beacon action property.
`BeaconRuleActionID`	*Type:* integer. Key Beacon action this property refers to.
`KeyName`	*Type:* text (max 256 characters). Key Property Key.
`Value`	*Type:* text Property Value.

BeaconRuleBeaconTargetMapping Table

Table that maps targets to rule.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 78: Database columns for `BeaconRuleBeaconTargetMapping` table

Database Column	Details
`BeaconRuleID`	*Type:* integer. Key Foreign key to the `BeaconRule` table.
`BeaconTargetID`	*Type:* integer. Key Foreign key to the `BeaconTarget` table.

BeaconSiteSubnetMapping Table

Table that maps site to Beacons.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 79: Database columns for `BeaconSiteSubnetMapping` table

Database Column	Details
`BeaconID`	*Type:* integer. Key Foreign key to the `Beacon` table.
`SubnetID`	*Type:* integer. Key Foreign key to the `SiteSubnet` table.

BeaconTarget Table

The `BeaconTarget` table contains beacon rule targets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 80: Database columns for `BeaconTarget` table

Database Column	Details
`BeaconTargetID`	*Type:* integer. Key. Generated ID Unqiue ID automatically assigned to each beacon targets.
`Name`	*Type:* text (max 100 characters). Key Name identifying the target.
`Description`	*Type:* text (max 256 characters). Nullable Name identifying the target.
`Internal`	*Type:* boolean. Key Is this target used internally, or managed by the user.
`Visible`	*Type:* boolean Can this target be displayed to the user for selection etc. This does not apply to the actual Targets page.

BeaconTargetAgentEvent Table

Table that maps agent events to targets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 81: Database columns for `BeaconTargetAgentEvent` table

Database Column	Details
`BeaconTargetID`	*Type:* integer. Key Foreign key to the `BeaconTarge`; table.
`BeaconAgentEventUID`	*Type:* unique identifier. Key Foreign key to the `BeaconAgentEvent` table.

BeaconTargetDiscoveredDeviceMapping Table

Table that maps site to targets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 82: Database columns for `BeaconTargetDiscoveredDeviceMapping` table

Database Column	Details
`BeaconTargetID`	*Type:* integer. Key Foreign key to the `BeaconTarget` table.
`DeviceID`	*Type:* integer. Key Foreign key to the `DiscoveredDevice` table.
`Include`	*Type:* boolean Boolean string indicating to include or exclude Device.

BeaconTargetPropertyValue Table

The `BeaconTargetPropertyValue` table contains beacon target property value elements.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 83: Database columns for `BeaconTargetPropertyValue` table

Database Column	Details
`BeaconTargetPropertyID`	*Type:* integer. Key. Generated ID Unqiue ID assigned to each beacon target property.
`BeaconTargetID`	*Type:* integer. Key Beacon target this property refers to.
`KeyName`	*Type:* text (max 256 characters). Key Property Key.
`Value`	*Type:* text (max 4000 characters) Property Value.

BeaconTargetSiteMapping Table

Table that maps site to targets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 84: Database columns for `BeaconTargetSiteMapping` table

Database Column	Details
`BeaconTargetID`	*Type:* integer. Key Foreign key to the `BeaconTarget` table.
`SiteID`	*Type:* integer. Key Foreign key to the `Site` table.
`Include`	*Type:* boolean Boolean string indicating to include or exclude Device.

BeaconTargetSiteSubnetMapping Table

Table that maps site to targets.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 85: Database columns for `BeaconTargetSiteSubnetMapping` table

Database Column	Details
`BeaconTargetID`	*Type:* integer. Key Foreign key to the `BeaconTarget` table.
`SubnetID`	*Type:* integer. Key Foreign key to the `SiteSubnet` table.
`Include`	*Type:* boolean Boolean string indicating to include or exclude Device.

BeaconUpgradeMode Table

`BeaconUpgradeMode` is a static table listing all of the styles of upgrade that a beacon can follow.

Table 86: Database columns for `BeaconUpgradeMode` table

Database Column	Details
`BeaconUpgradeModeID`	*Type:* integer. Key. Generated ID

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the BeaconUpgradeMode record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

BeaconUpgradeStatus Table

`BeaconUpgradeStatus` is a static table listing all of the upgrade states that a beacon can be in.

Table 87: Database columns for `BeaconUpgradeStatus` table

Database Column	Details
`BeaconUpgradeStatusID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the BeaconUpgradeStatus record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

BeaconWebServerStatus Table

`BeaconWebServerStatus` is a static table listing all of the states of the beacons web server.

Table 88: Database columns for `BeaconWebServerStatus` table

Database Column	Details
`BeaconWebServerStatusID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the BeaconWebServerStatus record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

DiscoveredDeviceDiscoveredBy Table

By which inventory beacon was this device discovered? Sometimes useful when other identifying features are duplicated, and when the distribution server should do something to the device.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 89: Database columns for `DiscoveredDeviceDiscoveredBy` table

Database Column	Details
`DeviceID`	*Type:* integer. Key The id of the device discovered.
`ServerUID`	*Type:* unique identifier. Key The inventory beacon that discovered it.
`RuleID`	*Type:* integer. Key. Nullable The RuleID executed on the beacon that discovered the device.
`CanAdminister`	*Type:* boolean. Nullable Does the distribution server have administrative privileges for the device?
`LastUpdate`	*Type:* datetime The date and time that the distribution server last reported its discovery of this device.
`AccountID`	*Type:* integer. Key. Nullable Account that can administer the device.
`AccountIDOverride`	*Type:* integer. Key. Nullable Account that can administer the device, overridden by the user.

DiscoveredDeviceDiscoveryStatus Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 90: Database columns for `DiscoveredDeviceDiscoveryStatus` table

Database Column	Details
`DeviceID`	*Type:* integer. Key
`TaskTypeID`	*Type:* integer. Key
`BeaconRuleID`	*Type:* integer. Key Rule that executed this task.
`BeaconPolicyRevision Number`	*Type:* integer The beacon policy revision number where rule is found
`SessionUID`	*Type:* unique identifier. Nullable
`DiscoveryDate`	*Type:* datetime. Nullable
`RuleDiscoveryAction SummaryID`	*Type:* integer Rule discovery summary.
`BeaconUID`	*Type:* unique identifier. Key. Nullable The inventory beacon that ran the task.

DiscoveredDeviceInventoryStatus Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 91: Database columns for `DiscoveredDeviceInventoryStatus` table

Database Column	Details
`DeviceID`	*Type:* integer. Key
`TaskTypeID`	*Type:* integer. Key
`BeaconRuleID`	*Type:* integer. Key. Nullable Rule that executed this task.
`BeaconPolicyRevision Number`	*Type:* integer. Nullable The beacon policy revision number where rule is found
`SessionUID`	*Type:* unique identifier. Nullable
`InventoryDate`	*Type:* datetime. Nullable

Database Column	Details
`RuleInventoryActionSummaryID`	*Type:* integer Rule action summary.
`BeaconUID`	*Type:* unique identifier. Key. Nullable The inventory beacon that ran the task.

DiscoveredDeviceTaskDetailedError Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 92: Database columns for `DiscoveredDeviceTaskDetailedError` table

Database Column	Details
`DiscoveredDeviceTaskDetailedErrorID`	*Type:* integer. Key. Generated ID The ID of the discovered device error.
`DiscoveredDeviceTaskStatusHistoryID`	*Type:* integer. Key Discovered device task status.
`Status`	*Type:* text (max 256 characters). Key The status code of task.
`DetailedStatus`	*Type:* text. Nullable The detailed error status.

DiscoveredDeviceTaskStatus Table

Records any task status information for `DiscoveredDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 93: Database columns for `DiscoveredDeviceTaskStatus` table

Database Column	Details
`DiscoveredDeviceTaskStatusID`	*Type:* integer. Key. Generated ID The ID of the discovered device task.

Database Column	Details
`DeviceID`	*Type:* integer. Key Device identity number.
`TaskTypeID`	*Type:* integer. Key The type of task which was run on the device.
`BeaconUID`	*Type:* unique identifier. Key. Nullable The inventory beacon that has run the task.
`BeaconRuleID`	*Type:* integer. Key. Nullable Rule that executed this task.
`Success`	*Type:* boolean. Key Status of the task. It can be Success OR Failed
`Credential`	*Type:* text (max 256 characters). Nullable The credential name for the task performed.
`Status`	*Type:* text (max 256 characters) The status code of task.
`DetailedStatus`	*Type:* text. Nullable The detailed error status.
`StartDateTime`	*Type:* datetime Date and time the task was started.
`BeaconPolicyRevision Number`	*Type:* integer. Nullable The beacon policy revision number where rule is found
`SessionUID`	*Type:* unique identifier. Nullable An identifier `TaskExecutionStatus` table
`IsSkipTask`	*Type:* boolean Determines whether the task status is a skip task
`IsDiscoveryTask`	*Type:* boolean Determines whether the task status is a discovery task

DiscoveredDeviceTaskStatusHistory Table

Records any task status information for `DiscoveredDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 94: Database columns for `DiscoveredDeviceTaskStatusHistory` table

Database Column	Details
`DiscoveredDeviceTaskStatusHistoryID`	*Type:* integer. Key. Generated ID The ID of the discovered device task.
`DeviceID`	*Type:* integer. Key Device identity number.
`TaskTypeID`	*Type:* integer. Key The type of task which was run on the device.
`SessionUID`	*Type:* unique identifier. Key An identifier `TaskExecutionStatus` table
`BeaconUID`	*Type:* unique identifier. Key The inventory beacon that has run the task.
`BeaconRuleID`	*Type:* integer. Key. Nullable Rule that executed this task.
`Success`	*Type:* boolean. Key Status of the task. It can be Success OR Failed
`Credential`	*Type:* text (max 256 characters). Nullable The credential name for the task performed.
`Status`	*Type:* text (max 256 characters) The status code of task.
`DetailedStatus`	*Type:* text. Nullable The detailed error status.
`StartDateTime`	*Type:* datetime Date and time the task was started.
`BeaconPolicyRevisionNumber`	*Type:* integer. Nullable The beacon policy revision number where rule is found
`IsSkipTask`	*Type:* boolean. Key Determines whether the task status is a skip task
`IsDiscoveryTask`	*Type:* boolean. Key Determines whether the task status is a discovery task

DiscoveredDeviceTaskType Table

This table stores the information about different types of tasks executed on a discovered device and their associated IDs.

Table 95: Database columns for `DiscoveredDeviceTaskType` table

Database Column	Details
`TaskTypeID`	*Type:* integer. Key. Generated ID The id for the task.
`TaskTypeName`	*Type:* text (max 32 characters). Key The name of the task.

ErrorCategory Table

Reported error category

Table 96: Database columns for `ErrorCategory` table

Database Column	Details
`ErrorCategoryID`	*Type:* integer. Key. Generated ID The ID of the error category.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a error category name. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

FNMEAAgent Table

The `FNMEAAgent` table stores the FNM-EA connection defined in inventory beacons.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 97: Database columns for `FNMEAAgent` table

Database Column	Details
`FNMEAAgentID`	*Type:* integer. Key. Generated ID Auto-generated FNMEA agent connection ID
`BeaconID`	*Type:* integer. Key. Nullable Beacon where the FNM-EA agent connection is defined
`AgentIdentifier`	*Type:* unique identifier. Key The GUID of the FNM-EA agent defined on inventory beacon.
`AgentName`	*Type:* text (max 128 characters) The FNM-EA agent name defined on inventory beacon.
`LastReportedLogRotation`	*Type:* datetime. Nullable Date time of the last report log rotation.
`LastReportedAgentStatus`	*Type:* datetime. Nullable Date time of the last reported status.

IncomingBaseline Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 98: Database columns for `IncomingBaseline` table

Database Column	Details
`Type`	*Type:* text (max 16 characters). Key The baseline type
`Date`	*Type:* datetime. Key The date of the baseline data
`ProductPool`	*Type:* text (max 128 characters). Key The license product pool
`ProductFamily`	*Type:* text (max 256 characters). Key The license product family
`ProductVersion`	*Type:* text (max 50 characters). Key The license product version

Database Column	Details
`EffectiveQuantity`	*Type:* integer The effective quantity of the license
`UpgradeQuantity`	*Type:* integer The upgrade quantity of the license
`UpgradeWithMaintenance Quantity`	*Type:* integer The upgrade with maintenance quantity of the license
`ActiveSAQuantity`	*Type:* integer The active software assurance quantity of the license
`ExpiringSA0To12Months`	*Type:* integer The software assurance quantity expiring within 0-12 months
`ExpiringSA12To24Months`	*Type:* integer The software assurance quantity expiring within 12-24 months
`ExpiringSA24PlusMonths`	*Type:* integer The software assurance quantity expiring greater than 24 months

ReconcileSoftwareLicenseReconcileExemptionReason Table

The `ReconcileSoftwareLicenseReconcileExemptionReason` table stores the staging license reconcile generated exemption reasons.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 99: Database columns for `ReconcileSoftwareLicenseReconcileExemptionReason` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key Foreign key to the `SoftwareLicense` table
`ComplianceComputerID`	*Type:* integer. Key. Nullable Foreign key to the `ComplianceComputer` table
`ComplianceUserID`	*Type:* integer. Key. Nullable Foreign key to the `ComplianceUser` table

Database Column	Details
SoftwareLicense ExemptionReasonID	*Type:* integer. Key Foreign key to the SoftwareLicenseExemptionReason table
AccessingUserID	*Type:* integer. Key. Nullable Foreign key to the AccessingUser table

RuleDiscoveryActionSummary Table

Summary of the discovery action.

Table 100: Database columns for RuleDiscoveryActionSummary table

Database Column	Details
RuleDiscoveryAction SummaryID	*Type:* integer. Key. Generated ID The ID of the discovery action summary.
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a discovery action summary. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

RuleInventoryActionSummary Table

Summary of the inventory gathering action.

Table 101: Database columns for RuleInventoryActionSummary table

Database Column	Details
RuleInventoryAction SummaryID	*Type:* integer. Key. Generated ID The ID of the inventory gathering action summary.
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a discovery action summary. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

SoftwareLicenseReconcileExemptionReasonData Table

The `SoftwareLicenseReconcileExemptionReasonData` table stores the exemption reasons generated by the license reconcile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 102: Database columns for `SoftwareLicenseReconcileExemptionReasonData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key Foreign key to the `SoftwareLicenseSnapshot` table
`ComplianceComputerID`	*Type:* integer. Key. Nullable Foreign key to the `ComplianceComputerSnapshot` table
`ComplianceUserID`	*Type:* integer. Key. Nullable Foreign key to the `ComplianceUserSnapshot` table
`SoftwareLicense ExemptionReasonID`	*Type:* integer. Key Foreign key to the `SoftwareLicenseExemptionReason` table
`AccessingUserID`	*Type:* integer. Key. Nullable Foreign key to the `AccessingUserSnapshot` table
`LicenseMeasurementID`	*Type:* integer. Key The snapshot ID. Foreign key to the `LicenseMeasurement` table.

StatusCodeCategory Table

Reported error category

Table 103: Database columns for `StatusCodeCategory` table

Database Column	Details
`StatusCodeCategoryID`	*Type:* integer. Key. Generated ID The ID of the error category.
`StatusCode`	*Type:* text (max 256 characters). Key Status code.

Database Column	Details
`ErrorCategoryID`	*Type:* integer. Nullable An identifier `ErrorCategory` table

UIAlignmentType Table

Table 104: Database columns for `UIAlignmentType` table

Database Column	Details
`UIAlignmentTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `UIAlignmentType`. Possible values are: • 1 = UseAvailableSpace • 2 = ForceLeft • 3 = ForceRight
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a insert type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

UIFieldType Table

Table 105: Database columns for `UIFieldType` table

Database Column	Details
`UIFieldTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `UIFieldType`. Possible values are: • 1 = Tab • 2 = Section • 3 = Integer • 4 = Text box • 5 = Text area • 6 = Date • 7 = Date and time • 8 = Combo box • 9 = Check box
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a connection type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

UIInsertType Table

Table 106: Database columns for `UIInsertType` table

Database Column	Details
`UIInsertTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `UIInsertType`. Possible values are: • 1 = Before • 2 = After • 3 = Start of

Database Column	Details
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a insert type. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

UIItem Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 107: Database columns for UIItem table

Database Column	Details
UIItemID	*Type:* integer. Key. Generated ID
TargetTypeID	*Type:* integer. Key Type of object. Foreign key to the TargetType table.
ItemResourceName	*Type:* text (max 256 characters). Key Name of the item
ItemName	*Type:* text (max 256 characters) Name of the item
UIFieldTypeID	*Type:* integer. Nullable UI field type if the elemet type is of type ‘field’. Foreign key to the UIFieldType table.
UIInsertTypeID	*Type:* integer Insert type. Foreign key to UIInsertType table.
UIAlignmentTypeID	*Type:* integer. Nullable Alignment type. Foreign key to UIAlignmentType table.
TabName	*Type:* text (max 80 characters) Name of the object to place the UI item.
RelativePositionTo	*Type:* text (max 80 characters) Name of the object to place the UI item.

Database Column	Details
`Position`	*Type:* integer
`Width`	*Type:* integer
`DataSource`	*Type:* XML. Nullable Date source for item of element type ‘field’ and of field type combo box . .
`SequenceNumber`	*Type:* integer Sequence where items to be added into UI
`FromTable`	*Type:* text. Nullable The name of the database table where the field can be found.
`SelectName`	*Type:* text. Nullable The name of the field in the database.
`WhereClause`	*Type:* text. Nullable The SQL “WHERE” statement that limits the information returned.
`Required`	*Type:* boolean Is the field a mandatory field.
`StringLength`	*Type:* integer String length.
`ReadOnly`	*Type:* boolean Is the field a readonly field.

UIItemTargetSubType Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 108: Database columns for `UIItemTargetSubType` table

Database Column	Details
`UIItemTargetSubTypeID`	*Type:* integer. Key. Generated ID
`UIItemID`	*Type:* integer. Key Type of object. Foreign key to the `UIItem` table.
`TargetSubTypeID`	*Type:* integer. Key object subtype. Foreign key to the various object type tables.

Compliance.Logic.Core Tables

The complete set of database tables documented here includes:

- Activity table (see Activity Table)
- ActivitySource table (see ActivitySource Table)
- ActivityTraceLog table (see ActivityTraceLog Table)
- ActivityType table (see ActivityType Table)
- Alert table (see Alert Table)
- AlertCategory table (see AlertCategory Table)
- AlertTarget table (see AlertTarget Table)
- AlertType table (see AlertType Table)
- AssetContractPaymentSchedule table (see AssetContractPaymentSchedule Table)
- Attribute table (see Attribute Table)
- AvailabilityZone table (see AvailabilityZone Table)
- BusinessImportLogDetail table (see BusinessImportLogDetail Table)
- BusinessImportLogObject table (see BusinessImportLogObject Table)
- BusinessImportLogSummary table (see BusinessImportLogSummary Table)
- BusinessImportResult table (see BusinessImportResult Table)
- CloudServiceInstance table (see CloudServiceInstance Table)
- CloudServiceInstanceConnection table (see CloudServiceInstanceConnection Table)
- CloudServiceInstanceMatchingRule table (see CloudServiceInstanceMatchingRule Table)
- CloudServiceInstanceType table (see CloudServiceInstanceType Table)
- CloudServiceProvider table (see CloudServiceProvider Table)
- CloudServiceRegion table (see CloudServiceRegion Table)
- ComplianceComputer table (see ComplianceComputer Table)
- ComplianceComputerConnection table (see ComplianceComputerConnection Table)
- ComplianceComputerContract table (see ComplianceComputerContract Table)
- ComplianceComputerInventorySourceType table (see ComplianceComputerInventorySourceType Table)
- ComplianceComputerPropertyValue table (see ComplianceComputerPropertyValue Table)
- ComplianceComputerRole table (see ComplianceComputerRole Table)
- ComplianceComputerRuleResult table (see ComplianceComputerRuleResult Table)

- ComplianceComputerStatus table (see ComplianceComputerStatus Table)
- ComplianceComputerType table (see ComplianceComputerType Table)
- ComplianceComputerTypeProperty table (see ComplianceComputerTypeProperty Table)
- ComplianceComputerUsage table (see ComplianceComputerUsage Table)
- ComplianceEvent table (see ComplianceEvent Table)
- ComplianceEventAction table (see ComplianceEventAction Table)
- ComplianceEventHistory table (see ComplianceEventHistory Table)
- ComplianceEventState table (see ComplianceEventState Table)
- ComplianceEventType table (see ComplianceEventType Table)
- ComplianceHistory table (see ComplianceHistory Table)
- ComplianceHistoryColumn table (see ComplianceHistoryColumn Table)
- ComplianceHistoryType table (see ComplianceHistoryType Table)
- ComplianceImage table (see ComplianceImage Table)
- ComplianceLicenseUser table (see ComplianceLicenseUser Table)
- CompliancePredefinedSearch table (see CompliancePredefinedSearch Table)
- ComplianceResponsibility table (see ComplianceResponsibility Table)
- ComplianceSavedSearch table (see ComplianceSavedSearch Table)
- ComplianceSchedule table (see ComplianceSchedule Table)
- ComplianceSearchFolder table (see ComplianceSearchFolder Table)
- ComplianceSearchType table (see ComplianceSearchType Table)
- ComplianceSearchTypeColumn table (see ComplianceSearchTypeColumn Table)
- ComplianceSearchTypeRelation table (see ComplianceSearchTypeRelation Table)
- ComplianceTask table (see ComplianceTask Table)
- ComplianceTenantSettingHistory table (see ComplianceTenantSettingHistory Table)
- ComplianceUserPropertyValue table (see ComplianceUserPropertyValue Table)
- ComplianceUserTypeProperty table (see ComplianceUserTypeProperty Table)
- ComputerChassisType table (see ComputerChassisType Table)
- ConsolidatedLicenseUser table (see ConsolidatedLicenseUser Table)
- ConsolidationType table (see ConsolidationType Table)
- Contract table (see Contract Table)

- ContractNote table (see ContractNote Table)
- ContractNotification table (see ContractNotification Table)
- ContractNotificationResponsibility table (see ContractNotificationResponsibility Table)
- ContractProperty table (see ContractProperty Table)
- ContractPropertyValue table (see ContractPropertyValue Table)
- ContractScopingData table (see ContractScopingData Table)
- ContractSecurityUser table (see ContractSecurityUser Table)
- ContractState table (see ContractState Table)
- ContractStatus table (see ContractStatus Table)
- ContractType table (see ContractType Table)
- ContractUseRight table (see ContractUseRight Table)
- ContractUseRightIBM table (see ContractUseRightIBM Table)
- ContractVendor table (see ContractVendor Table)
- CurrencyRate table (see CurrencyRate Table)
- CurrencyRateSnapshot table (see CurrencyRateSnapshot Table)
- CustomPropertyDisplayXML table (see CustomPropertyDisplayXML Table)
- DisplayXML table (see DisplayXML Table)
- Document table (see Document Table)
- DocumentHistory table (see DocumentHistory Table)
- DocumentNote table (see DocumentNote Table)
- DocumentType table (see DocumentType Table)
- Event table (see Event Table)
- EventLogCategory table (see EventLogCategory Table)
- EventLogDetail table (see EventLogDetail Table)
- EventLogLevel table (see EventLogLevel Table)
- EventLogStatus table (see EventLogStatus Table)
- EventLogSummary table (see EventLogSummary Table)
- EventParameter table (see EventParameter Table)
- EventParameterType table (see EventParameterType Table)
- EventSeverity table (see EventSeverity Table)

- EventTarget table (see EventTarget Table)
- EventType table (see EventType Table)
- EventTypeStatus table (see EventTypeStatus Table)
- ILMTPVUCounts table (see ILMTPVUCounts Table)
- ImportResolverErrorResult table (see ImportResolverErrorResult Table)
- ImportResolverType table (see ImportResolverType Table)
- InstalledSoftwareAttribute table (see InstalledSoftwareAttribute Table)
- Instance table (see Instance Table)
- InstanceAttribute table (see InstanceAttribute Table)
- InstanceEnvironment table (see InstanceEnvironment Table)
- InstancePropertyValue table (see InstancePropertyValue Table)
- InstanceRole table (see InstanceRole Table)
- InstanceTenancy table (see InstanceTenancy Table)
- InstanceType table (see InstanceType Table)
- InstanceTypeProperty table (see InstanceTypeProperty Table)
- InstanceUser table (see InstanceUser Table)
- IntervalType table (see IntervalType Table)
- LicenseUser table (see LicenseUser Table)
- LicenseUserConnection table (see LicenseUserConnection Table)
- LicenseUserExcluded table (see LicenseUserExcluded Table)
- LicenseUserType table (see LicenseUserType Table)
- LogFile table (see LogFile Table)
- MSEAARLSoftwareTitleEdition table (see MSEAARLSoftwareTitleEdition Table)
- MSSelectLevel table (see MSSelectLevel Table)
- MSSelectPool table (see MSSelectPool Table)
- MobileDevice table (see MobileDevice Table)
- NotificationItem table (see NotificationItem Table)
- NotificationTemplate table (see NotificationTemplate Table)
- NotificationType table (see NotificationType Table)
- OperatorManageState table (see OperatorManageState Table)

- OperatorTaskTypeSetting table (see OperatorTaskTypeSetting Table)
- OracleInstance table (see OracleInstance Table)
- PaymentSchedule table (see PaymentSchedule Table)
- PaymentScheduleCategory table (see PaymentScheduleCategory Table)
- PaymentScheduleDetail table (see PaymentScheduleDetail Table)
- PaymentScheduleDetailPaymentStatus table (see PaymentScheduleDetailPaymentStatus Table)
- PaymentScheduleTerm table (see PaymentScheduleTerm Table)
- PaymentScheduleType table (see PaymentScheduleType Table)
- Project table (see Project Table)
- PurchaseOrder table (see PurchaseOrder Table)
- PurchaseOrderDetail table (see PurchaseOrderDetail Table)
- PurchaseOrderDetailProperty table (see PurchaseOrderDetailProperty Table)
- PurchaseOrderDetailPropertyValue table (see PurchaseOrderDetailPropertyValue Table)
- PurchaseOrderDetailStatus table (see PurchaseOrderDetailStatus Table)
- PurchaseOrderDetailType table (see PurchaseOrderDetailType Table)
- PurchaseOrderProperty table (see PurchaseOrderProperty Table)
- PurchaseOrderPropertyValue table (see PurchaseOrderPropertyValue Table)
- PurchaseOrderStatus table (see PurchaseOrderStatus Table)
- PurchaseOrderType table (see PurchaseOrderType Table)
- PurchaseProgram table (see PurchaseProgram Table)
- QuerySnapshot table (see QuerySnapshot Table)
- RelationType table (see RelationType Table)
- ResponsibilityType table (see ResponsibilityType Table)
- RestrictedAccessType table (see RestrictedAccessType Table)
- RulesEngineRuleDefinition table (see RulesEngineRuleDefinition Table)
- RulesEngineRuleType table (see RulesEngineRuleType Table)
- SAMLConfiguration table (see SAMLConfiguration Table)
- SecurityType table (see SecurityType Table)
- SerialNumberBlackList table (see SerialNumberBlackList Table)
- SessionUIDBeacon table (see SessionUIDBeacon Table)

- ShippingMethod table (see ShippingMethod Table)
- SoftwareLicenseContractPaymentSchedule table (see SoftwareLicenseContractPaymentSchedule Table)
- SystemShutdown table (see SystemShutdown Table)
- TDSComputer table (see TDSComputer Table)
- TaskExecutionStatus table (see TaskExecutionStatus Table)
- TaskExecutionStatusStep table (see TaskExecutionStatusStep Table)
- TaskStep table (see TaskStep Table)
- TaskStepEventType table (see TaskStepEventType Table)
- TermAndCondition table (see TermAndCondition Table)
- TermAndConditionTask table (see TermAndConditionTask Table)
- TermAndConditionType table (see TermAndConditionType Table)
- UserNameBlacklist table (see UserNameBlacklist Table)
- VMEnabledState table (see VMEnabledState Table)
- VMHostDatastore table (see VMHostDatastore Table)
- VMHostManagedBySoftware table (see VMHostManagedBySoftware Table)
- VMHostProperty table (see VMHostProperty Table)
- VMPool table (see VMPool Table)
- VMPoolType table (see VMPoolType Table)
- VMSourceType table (see VMSourceType Table)
- VMState table (see VMState Table)
- VMType table (see VMType Table)
- Vendor table (see Vendor Table)
- VendorContact table (see VendorContact Table)
- VendorProperty table (see VendorProperty Table)
- VendorPropertyValue table (see VendorPropertyValue Table)
- VirtualMachine table (see VirtualMachine Table)
- XMLInsertType table (see XMLInsertType Table)
- ZoneResourceManagementMethodType table (see ZoneResourceManagementMethodType Table)

Activity Table

The `Activity` table stores errors and events processed by the beacon, devices, rules etc.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 109: Database columns for `Activity` table

Database Column	Details
`ActivityID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`SourceTypeID`	*Type:* integer The source type ID such as Beacon, External and so on
`SourceTypeName`	*Type:* text (max 256 characters) The source type name such as Beacon, External and so on
`ActivityTypeID`	*Type:* integer. Key Foreign key to the `ActivityType` table.
`ActivityUID`	*Type:* unique identifier. Key UID to uniquely identify the activity.
`DateCreated`	*Type:* datetime Time that the activity is created in the database.

ActivitySource Table

`ActivitySource` is a static table listing all of the Sources that can generate the activity logs.

Table 110: Database columns for `ActivitySource` table

Database Column	Details
`ActivitySourceID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the ActivitySource record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

ActivityTraceLog Table

The ActivityTraceLog table stores the logs generated by the trace logger for the corresponding activity.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 111: Database columns for `ActivityTraceLog` table

Database Column	Details
`TraceID`	*Type:* integer. Key. Generated ID The unique row identifier.
`ActivityUID`	*Type:* unique identifier. Nullable The Guid of the activity that trace logger is logging the events for.
`DateCreated`	*Type:* datetime. Nullable The date and time when teh event occurred.
`LogMessage`	*Type:* text. Nullable The actual message logged by the trace logger.
`LogLevel`	*Type:* integer. Nullable The log level that the trace logger is logging to.
`EventID`	*Type:* integer. Key. Nullable The unique row identifier in negative form.

ActivityType Table

The `ActivityType` table stores details about the different types of Activities.

Table 112: Database columns for `ActivityType` table

Database Column	Details
`ActivityTypeID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`ActivityTypeName`	*Type:* text (max 256 characters). Key A short piece of text representing the Activity Type. Internal use only- not to be displayed to the operator.
`ActivityMessageResource`	*Type:* text (max 256 characters) A resource name used to look up a description for this Activity

Database Column	Details
`IsMonitored`	*Type:* boolean Flag that determines whether to track this activity

Alert Table

The `Alert` table stores alerts and notifications that the system can attach to different objects to be displayed to the operator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 113: Database columns for `Alert` table

Database Column	Details
`AlertID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`AlertTypeID`	*Type:* integer. Key Foreign key to the `AlertType` table.
`Ignored`	*Type:* boolean. Key This flag indicates whether this alert has been ignored by an operator. If so, then the IgnoredDate and IgnoredOperator values will be populated.
`IgnoredDate`	*Type:* datetime. Nullable TIf the alert has been ignored by an operator, then this field shows the date when this was done.
`IgnoredOperator`	*Type:* text (max 256 characters). Nullable If the alert has been ignored by an operator, then this field shows which operator ignored the alert.
`CreationDate`	*Type:* datetime Date and time (UTC) when alert was created.

AlertCategory Table

The `AlertCategory` table stores the different catogories of alerts.

Table 114: Database columns for `AlertCategory` table

Database Column	Details
`AlertCategoryID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`DefaultName`	*Type:* text (max 128 characters) The default name for this alert category
`ResourceName`	*Type:* text (max 128 characters). Key A resource name used to look up a description for this alert category

AlertTarget Table

The `AlertTarget` table stores the links between alerts and other tables in the database.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 115: Database columns for `AlertTarget` table

Database Column	Details
`AlertID`	*Type:* integer. Key Link to the `Alert` table
`TargetTypeID`	*Type:* integer. Key A link the the `TargetType` table. this value specifies which kind of object the alert is linked to.
`TargetID`	*Type:* integer. Key used to attach the `Alert` to its target. The target table depends on the TargetTypeID of the linked AlertType.
`FieldName`	*Type:* text. Nullable A semi-colon separated list of view-model names that represent the fields that the alert is attached to. A null value indicates that the alert applies to the overall object as a whole.

AlertType Table

The `AlertType` table stores details about the different types of alerts.

Table 116: Database columns for `AlertType` table

Database Column	Details
`AlertTypeID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`AlertTypeName`	*Type:* text (max 256 characters). Key A short piece of text representing the Alert Type. Internal use only- not to be displayed to the operator.
`AlertMessageResource`	*Type:* text (max 256 characters) A resource name used to look up a description for this alert
`AlertCategoryID`	*Type:* integer The category of this type of alert

AssetContractPaymentSchedule Table

`AssetContractPaymentSchedule` links a payment schedule to an asset, via a link from that asset to a contract.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 117: Database columns for `AssetContractPaymentSchedule` table

Database Column	Details
`AssetContractPaymentScheduleID`	*Type:* integer. Key. Generated ID Unique identifier to represent a link between a payment schedule and an asset. This allows an asset to link multiple times to a payment schedule, each time with its own start and end dates.
`AssetContractID`	*Type:* integer. Key Identifies a link between an asset and a contract. Foreign key to the `AssetContract` table.
`PaymentScheduleID`	*Type:* integer. Key Identifies a payment schedule. Foreign key to the `PaymentSchedule` table.
`ActiveStartDate`	*Type:* datetime Start date of the association between the payment schedule and asset.
`ActiveEndDate`	*Type:* datetime. Nullable End date of the association between the payment schedule and asset.

Attribute Table

`Attribute` holds the collection of possible attributes of database instances.

Table 118: Database columns for `Attribute` table

Database Column	Details
`AttributeID`	*Type:* integer. Key. Generated ID A unique identifier for an attribute.
`AttributeName`	*Type:* text (max 256 characters). Key The name of the attribute.

AvailabilityZone Table

`AvailabilityZone` is a table listing the possible availability zone in a cloud service provider.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 119: Database columns for `AvailabilityZone` table

Database Column	Details
`AvailabilityZoneID`	*Type:* integer. Key. Generated ID A unique identifier for a cloud service availability zone.
`Name`	*Type:* text (max 256 characters). Nullable Location of the instance.
`IsShared`	*Type:* boolean

BusinessImportLogDetail Table

The `BusinessImportLogDetail` table stores per record import execution details for a business import execution.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 120: Database columns for `BusinessImportLogDetail` table

Database Column	Details
`ImportDetailID`	*Type:* integer. Key. Generated ID Surrogate ID that uniquely identifies an import execution detail.
`ImportID`	*Type:* integer. Key Business import ID this execution detail relates to, foreign key to `BusinessImportLogSummary` table.
`RecordNumber`	*Type:* integer. Nullable Row number of source data in staging table that this execution detail related to.
`Action`	*Type:* text (max 10 characters). Nullable The trace action of the import execution detail.
`MGSRecordKey`	*Type:* text (max 50 characters). Nullable ID of matching FNMS table record the Record Number is matched against.
`ImportObjectID`	*Type:* integer. Key. Nullable Import object that this execution detail is related to, foreign key to `BusinessImportLogObject` table.
`RecordDescription`	*Type:* text (max 255 characters). Nullable Value of the trace field specified in the import element of business adapter xml if any.
`Message`	*Type:* text (max 3000 characters). Nullable Messages related to this import execution detail.

BusinessImportLogObject Table

The `BusinessImportLogObject` table stores summary data for the execution of individual object imports within a business import execution.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 121: Database columns for `BusinessImportLogObject` table

Database Column	Details
`ImportObjectID`	*Type:* integer. Key. Generated ID Surrogate ID that uniquely identifies an object in a business import execution.

Database Column	Details
`ImportID`	*Type:* integer. Key Business import ID this object belongs, foreign key to `BusinessImportLogSummary` table.
`ObjectName`	*Type:* text (max 50 characters). Nullable Name of the business import object.
`ObjectType`	*Type:* text (max 50 characters). Nullable Type of the business import object.
`StartDate`	*Type:* datetime. Nullable Date and time when the object began to be imported on FNMS server.
`EndDate`	*Type:* datetime. Nullable Date and time when import of the object is completed on FNMS server.
`Status`	*Type:* integer. Nullable Status of object import: 0 - Not completed, 1 - Completed.
`Processed`	*Type:* integer. Nullable Number of rows from data source that are processed for the object import.
`Matched`	*Type:* integer. Nullable Number of rows in the staging table that match records in the corresponding FNMS table for the object.
`Rejected`	*Type:* integer. Nullable Number of rows in the staging table that are rejected for the object import.
`Updated`	*Type:* integer. Nullable Number of rows in the staging table that are updated for the object import.
`Created`	*Type:* integer. Nullable Number of rows in the staging table that are created for the object import.
`Deleted`	*Type:* integer. Nullable Number of rows in the staging table that are deleted for the object import.

BusinessImportLogSummary Table

The `BusinessImportLogSummary` table stores summary data for each business import execution.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 122: Database columns for `BusinessImportLogSummary` table

Database Column	Details
`ImportID`	*Type:* integer. Key. Generated ID Surrogate ID that uniquely identifies a business import.
`ImportName`	*Type:* text (max 255 characters). Nullable Import name of the business import.
`ImportType`	*Type:* text (max 50 characters). Nullable Import type of the business import.
`Action`	*Type:* text (max 20 characters). Nullable The mode the business import is operating in e.g. Import, Simulation.
`StartDate`	*Type:* datetime. Nullable Date and time when the business import is started on FNMS server.
`EndDate`	*Type:* datetime. Nullable Date and time when the business import is completed on FNMS server.
`Status`	*Type:* integer. Nullable Status of the business import: 0 - Not completed, 1 - Completed.
`Processed`	*Type:* integer. Nullable Number of rows from data source that are processed for import.
`Rejected`	*Type:* integer. Nullable Number of rows from data source that are rejected from importing.
`SessionUID`	*Type:* unique identifier. Key. Nullable Unique task run identifier of the business import, nullable for business import initiated on the server.

BusinessImportResult Table

The `BusinessImportResult` table contains the results of all business imports executed on the batch server.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 123: Database columns for `BusinessImportResult` table

Database Column	Details
`BusinessImportResultID`	*Type:* integer. Key. Generated ID A unique identifier for the business import result.
`ImportName`	*Type:* text (max 256 characters) The name of the business import.
`BeaconID`	*Type:* integer. Key A link to `Beacon` from which this import was uploaded.
`ImportStarted`	*Type:* datetime The time at which the import was executed.
`ImportEnded`	*Type:* datetime The time at which the import was completed.
`Result`	*Type:* boolean Whether the import succeeded.

CloudServiceInstance Table

`CloudServiceInstance` stores information for cloud service instances.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 124: Database columns for `CloudServiceInstance` table

Database Column	Details
`CloudServiceInstanceID`	*Type:* big integer. Key. Generated ID A unique identifier for cloud service instance on a computer.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The computer where this cloud service instance is linked to. Foreign key to the `ComplianceComputer` table.
`HostComplianceComputerID`	*Type:* integer. Key. Nullable The instance's host computer. Foreign key to the `ComplianceComputer` table.
`HostID`	*Type:* text (max 256 characters). Nullable The ID of the dedicated host instance.

Database Column	Details
`CloudServiceProviderID`	*Type:* integer. Key The cloud service provider for this instance. Foreign key to the `CloudServiceProvider` table.
`InstanceCloudID`	*Type:* text (max 256 characters). Key The ID of the cloud instance.
`CloudServiceInstance TypeID`	*Type:* integer. Nullable Cloud instance type defined by provider. Foreign key to the `CloudServiceInstanceType` table
`CloudServiceRegionID`	*Type:* integer. Nullable Region of the instance. Foreign key to the `CloudServiceRegion` table.
`AvailabilityZoneID`	*Type:* integer. Nullable Location of the instance. Foreign key to the `AvailabilityZone` table
`InstanceTenancyID`	*Type:* integer. Nullable Instance tenancy of the instance. Foreign key to the `InstanceTenancy` table
`InstanceAffinity`	*Type:* text (max 256 characters). Nullable The affinity setting for the instance on the Dedicated Host.
`ImageID`	*Type:* text (max 256 characters). Nullable The ID of the image used to launch the instance.
`LaunchTime`	*Type:* datetime. Nullable The time the cloud instance was launched or the Reserved Instance started.
`NetworkID`	*Type:* text (max 256 characters). Nullable The ID of the Virtual Private Cloud.
`LifecycleMode`	*Type:* text (max 256 characters). Nullable The lifecycle state of the instance.
`ExpiryTime`	*Type:* datetime. Nullable The time when the Reserved Instance expires.
`InstanceCount`	*Type:* integer. Nullable The number of reservations purchased.
`OfferingClass`	*Type:* text (max 256 characters). Nullable The offering class of the Reserved Instance.
`OfferingType`	*Type:* text (max 256 characters). Nullable The Reserved Instance offering type.

Database Column	Details
`Scope`	*Type:* text (max 256 characters). Nullable The scope of the Reserved Instance.
`Account`	*Type:* text (max 256 characters). Nullable The Account that is used to create the instance.
`CoreCount`	*Type:* integer. Nullable The number of core of the instance.
`ThreadsPerCore`	*Type:* integer. Nullable The number of thread per core of the instance.
`VMEnabledStateID`	*Type:* integer. Nullable The operational state of the instance. Foreign key to the `VMEnabledState` table.
`MacAddress`	*Type:* text (max 256 characters). Nullable The MAC address of the computer. This may be a comma-separated list if there is more than one active network adapter in the system. Do not include inactive network adapters and network adapters with invalid MAC addresses.
`MatchingRuleID`	*Type:* integer. Nullable Matching rule is used to match the instance with an inventory in Compliance Computer. Foreign key to the `CloudServiceInstanceMatchingRule` table.
`InventoryDate`	*Type:* datetime. Key. Nullable The time the cloud instance was reported by inventory.
`CloudServiceProvider InventoryDate`	*Type:* datetime. Nullable The time the cloud instance was reported by an adapter.

CloudServiceInstanceConnection Table

`CloudServiceInstanceConnection` stores a link between cloud service instance in `CloudServiceInstance` which have been reported in inventory, and ComplianceConnectionID that can be used to identify which inventory sources that the cloud service instance come from.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 125: Database columns for `CloudServiceInstanceConnection` table

Database Column	Details
`CloudServiceInstanceID`	*Type:* big integer. Key A unique identifier for the cloud service instance. Foreign key to the `CloudServiceInstance` table.
`ComplianceConnectionID`	*Type:* integer. Key The inventory source where the cloud service instance was reported. Foreign key to the `ComplianceConnection` table.

CloudServiceInstanceMatchingRule Table

The `CloudServiceInstanceMatchingRule` table contains the list of rules to match cloud service instance with an inventory in ComplianceComputer table.

Table 126: Database columns for `CloudServiceInstanceMatchingRule` table

Database Column	Details
`CloudServiceInstance MatchingRuleID`	*Type:* integer. Key. Generated ID A unique identifier for each `CloudServiceInstanceMatchingRule`. Possible values and the corresponding default strings are: • 1 = Inventory • 2 = MACAddress • 3 = MGSBI • 4 = UI • 5 = InstanceCloudID
`Name`	*Type:* text (max 256 characters). Key Name of the matching rule for cloud service instance.

CloudServiceInstanceType Table

`CloudServiceInstanceType` is a table listing the possible types of a cloud service instance.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 127: Database columns for `CloudServiceInstanceType` table

Database Column	Details
`CloudServiceInstanceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a cloud service instance type.
`Name`	*Type:* text (max 256 characters). Nullable Cloud instance type defined by cloud service provider.
`IsShared`	*Type:* boolean

CloudServiceProvider Table

The `CloudServiceProvider` table contains the Cloud Service providers for the virtual machines.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 128: Database columns for `CloudServiceProvider` table

Database Column	Details
`CloudServiceProviderID`	*Type:* integer. Key. Generated ID A unique identifier for a cloud service provider record.
`Name`	*Type:* text (max 256 characters). Key Name of the cloud service provider.
`IsShared`	*Type:* boolean

CloudServiceRegion Table

`Region` is a table listing the possible regions in a cloud service provider.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 129: Database columns for `CloudServiceRegion` table

Database Column	Details
`CloudServiceRegionID`	*Type:* integer. Key. Generated ID A unique identifier for a cloud service region.
`Name`	*Type:* text (max 256 characters). Nullable Region of the instance.
`IsShared`	*Type:* boolean

ComplianceComputer Table

`ComplianceComputer` stores information about computers used in the enterprise, including hardware details, inventory source information and computer types.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 130: Database columns for `ComplianceComputer` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key. Generated ID A unique identifier for a `ComplianceComputer`.
`ComplianceComputerTypeID`	*Type:* integer. Key A unique identifier for the type of computer. Foreign key to the `ComplianceComputerType` table.
`IsComplianceComputer TypeIDFromInventory`	*Type:* boolean This is true for records sourced from inventory, where the inventory source has specified the value of the ComplianceComputerTypeID. A true value will exclude this record from some processes that infer the type of a record. This value is set by the import process.
`ComputerName`	*Type:* text (max 256 characters). Key. Nullable The name of the computer.
`ComplianceDomainID`	*Type:* integer. Key. Nullable The domain to which the computer belongs. Foreign key to the `ComplianceDomain` table.

Database Column	Details
`ComplianceComputer StatusID`	*Type:* integer. Key The last recorded status for this computer. Foreign key to the `ComplianceComputerStatus` table.
`ComplianceComputerRoleID`	*Type:* integer. Key The functional role of this computer. Foreign key to the `ComplianceComputerRole` table.
`ComplianceComputer InventorySourceTypeID`	*Type:* integer. Key Whether this computer has ever been reported in inventory, or has been manually created and maintained. Foreign key to the `ComplianceComputerInventorySourceType` table.
`AssetID`	*Type:* integer. Key. Nullable When the computer is being managed as an asset, this is a foreign key to the `Asset` table; and is otherwise null.
`OperatingSystem`	*Type:* text (max 128 characters). Nullable The operating system of the computer.
`ServicePack`	*Type:* text (max 128 characters). Nullable The latest service pack reported as installed on the operating system.
`NumberOfProcessors`	*Type:* integer. Nullable The number of processors in the computer.
`NumberOfProcessorsDefault`	*Type:* integer. Nullable The inventoried number of processors in the computer.
`ProcessorType`	*Type:* text (max 256 characters). Nullable The type of processor in the computer.
`ProcessorTypeDefault`	*Type:* text (max 256 characters). Nullable The inventoried type of processor in the computer.
`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed of the fastest processor in the computer in megahertz.
`MaxClockSpeedDefault`	*Type:* integer. Nullable The inventoried maximum clock speed of the fastest processor in the computer in megahertz.
`TotalMemory`	*Type:* big integer. Nullable The total RAM in the computer.

Database Column	Details
`ChassisTypeID`	*Type:* integer. Key The type of case for the computer, as reported in hardware inventory, defaulting to `Unknown` if no chassis type is reported. Foreign key to the `ComputerChassisType` table.
`AssignedChassisTypeID`	*Type:* integer. Nullable The type of case for the computer, as set by an operator. Foreign key to the `ComputerChassisType` table.
`NumberOfHardDrives`	*Type:* integer. Nullable The number of hard drives in the computer.
`TotalDiskSpace`	*Type:* big integer. Nullable The total size of all hard drives in the computer.
`NumberOfNetworkCards`	*Type:* integer. Nullable The number of network cards in the computer.
`NumberOfDisplayAdapters`	*Type:* integer. Nullable The number of graphics cards in the computer.
`IPAddress`	*Type:* text (max 256 characters). Nullable The IP address of the computer.
`MACAddress`	*Type:* text (max 256 characters). Nullable The MAC Addresses of the computer.
`Manufacturer`	*Type:* text (max 128 characters). Key. Nullable The manufacturer of the computer.
`ModelNo`	*Type:* text (max 128 characters). Nullable The model number of the computer.
`ModelNoDefault`	*Type:* text (max 128 characters). Nullable The inventoried model number of the computer.
`SerialNo`	*Type:* text (max 100 characters). Key. Nullable The serial number of the computer.
`ComplianceUserID`	*Type:* integer. Key. Nullable The end-user who last logged onto the computer. Foreign key to the `ComplianceUser` table.
`AssignedUserID`	*Type:* integer. Key. Nullable The end-user assigned to this computer by an operator. Foreign key to the `ComplianceUser` table.

Database Column	Details
`CalculatedUserID`	*Type:* integer. Key. Nullable An end-user of this computer, calculated by looking at usage. Foreign key to the `ComplianceUser` table.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this computer. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any corporate unit in the enterprise associated with this computer. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any cost center in the enterprise associated with this computer. Foreign key to the `GroupEx` table.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise category associated with this computer. Foreign key to the `GroupEx` table.
`InventoryDate`	*Type:* datetime. Key. Nullable The date the computer last had inventory reported.
`HardwareInventoryDate`	*Type:* datetime. Nullable The date when the hardware was last reported.
`ServicesInventoryDate`	*Type:* datetime. Nullable The date when a service was last reported.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The name of the operator who last updated the computer details.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the computer was created.
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory.
`NumberOfCores`	*Type:* integer. Nullable The number of cores in the computer.

Database Column	Details
`NumberOfCoresDefault`	*Type:* integer. Nullable The inventoried number of cores in the computer.
`NumberOfSockets`	*Type:* integer. Nullable The number of sockets in the computer.
`NumberOfSocketsDefault`	*Type:* integer. Nullable The inventoried number of sockets in the computer.
`AssetComplianceStatusID`	*Type:* integer. Nullable For computers managed as assets, the latest compliance status of the computer. Foreign key to the `AssetComplianceStatus` table.
`PartialNumberOfProcessors`	*Type:* decimal. Nullable The fractional processor count available to this computer.
`PartialNumberOf ProcessorsDefault`	*Type:* decimal. Nullable The inventoried fractional processor count available to this computer.
`UntrustedSerialNo`	*Type:* boolean Is this computer known to have a serial number from a data source that should not be trusted.
`ILMTAgentID`	*Type:* big integer. Key. Nullable Store the unique ID used by the ILMT agent on this device, if the inventory source is aware of this value.
`FNMPComputerUID`	*Type:* unique identifier. Key. Nullable The unique identifier generated for the computer from the IM database. This property should only be populated by the ManageSoft inventory adapter.
`UUID`	*Type:* unique identifier. Nullable The computer's UUID, in the byte order reported in inventory.
`HostIdentifyingNumber`	*Type:* text (max 128 characters). Key. Nullable Virtual hosts may have an identifier that is unique only across that hardware model. It is less unique than the true hardware serial number, for example.
`HostType`	*Type:* text (max 128 characters). Key. Nullable The type (similar to model number) of the host, used for matching.
`NumberOfLogicalProcessors`	*Type:* integer. Nullable The number of logical processors in the computer.
`NumberOfLogical ProcessorsDefault`	*Type:* integer. Nullable The inventoried number of logical processors in the computer.

Database Column	Details
`PrimaryComplianceUserID`	*Type:* integer. Key. Nullable Primary user of the computer based off the assigned user and calculated user.
`MDScheduleGeneratedDate`	*Type:* datetime. Nullable The last time the managed device schedule was regenerated.
`MDScheduleContainsPVUScan`	*Type:* boolean. Nullable Does this managed device include an event in its current schedule for running extra IBM PVU hardware scans.
`HostID`	*Type:* text (max 100 characters). Key. Nullable Numeric identifier of the current host
`FirmwareSerialNumber`	*Type:* text (max 100 characters). Key. Nullable Serial number in the system firmware such as BIOS, EEPROM etc.
`MachineID`	*Type:* text (max 100 characters). Key. Nullable For AIX, it is the System ID. For HP-UX, it is the Machine/Software ID. It is unset for other platforms.
`CloudServiceProviderID`	*Type:* integer. Key. Nullable The cloud service provider for the virtual machine. Foreign key to the `CloudServiceProvider` table.

ComplianceComputerConnection Table

`ComplianceComputerConnection` stores a link between computers in `ComplianceComputer` which have been reported in inventory, and external IDs that can be used to identify them in their inventory sources. Computers reported in multiple inventory sources will appear multiple times in this table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 131: Database columns for `ComplianceComputerConnection` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key A unique identifier for the computer. Foreign key to the `ComplianceComputer` table.

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The inventory source where the computer was reported. Foreign key to the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer The (hopefully unique) identifier for the computer in the external inventory source.

ComplianceComputerContract Table

`ComplianceComputerContract` stores links between computers and contracts, some of which may influence license compliance.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 132: Database columns for `ComplianceComputerContract` table

Database Column	Details
`ComplianceComputer ContractID`	*Type:* integer. Key. Generated ID A unique identifier for this record.
`ContractID`	*Type:* integer. Key A unique identifier for a contract linked to a computer. Foreign key to the `Contract` table.
`ComplianceComputerID`	*Type:* integer. Key A unique identifier for a computer linked to a contract. Foreign key to the `ComplianceComputer` table.

ComplianceComputerInventorySourceType Table

`ComplianceComputerInventorySourceType` is a static table used to define possible computer inventory source values (that is, whether the computer was created manually or reported by the compliance importer).

Table 133: Database columns for `ComplianceComputerInventorySourceType` table

Database Column	Details
`ComplianceComputer InventorySourceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceComputerInventorySourceType`. Possible values and the corresponding default strings are: • 1 = Automatic (computer was recently updated during an inventory import) • 2 = VM Host (a dummy or "light" computer created using the host inventory of a virtual machine) • 3 = Manual (computer was created manually by an operator, using FlexNet Manager Suite, and has never been updated by the compliance importer).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a computer inventory source. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the inventory resource string has no translation.

ComplianceComputerPropertyValue Table

For each computer, `ComplianceComputerPropertyValue` stores the values for the custom properties defined in `ComplianceComputerTypeProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 134: Database columns for `ComplianceComputerPropertyValue` table

Database Column	Details
`ComplianceComputer PropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`ComplianceComputerID`	*Type:* integer. Key The computer associated with this property value. Foreign key to the `ComplianceComputer` table
`ComplianceComputerType PropertyID`	*Type:* integer. Key The property whose value is being stored. The type of the computer should match the type that the property is associated with. Foreign key to the `ComplianceComputerTypeProperty` table.

Database Column	Details
`PropertyValue`	*Type:* text (max 4000 characters) The value of the custom property.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

ComplianceComputerRole Table

`ComplianceComputerRole` is a static table listing all the different roles to which computers can be assigned, and which may impact licensing terms.

Table 135: Database columns for `ComplianceComputerRole` table

Database Column	Details
`ComplianceComputerRoleID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceComputerRole`. Possible values and the corresponding default strings are: • 1 = Production • 2 = Warm Standby / Passive Failover • 3 = Hot Standby / Active Failover • 4 = Backup / Archive • 5 = Test • 6 = Training • 7 = Cold Standby / Disaster recovery • 8 = Development.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a computer role. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the inventory resource string has no translation.
`ManageLicenses`	*Type:* boolean Set this to `True` if computers in this role are to be included in compliance calculations, and to `False` if this role exempts a computer from the license management process. Of the computer roles listed above, only `Active` computers have their licenses managed.

ComplianceComputerRuleResult Table

`ComplianceComputerRuleResult` stores rules results from Inventory Manager.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 136: Database columns for `ComplianceComputerRuleResult` table

Database Column	Details
`ComplianceComputerRuleResultID`	*Type:* integer. Key. Generated ID The identifier for a rule result.
`ComplianceComputerID`	*Type:* integer. Key The computer associated with this result. Foreign key to the `ComplianceComputer` table
`RecognitionRule`	*Type:* text (max 256 characters). Key. Nullable The recognition rule.
`Revision`	*Type:* integer. Nullable The revision number of the recognition rule.
`InventoryDate`	*Type:* datetime. Nullable The date the recognition rule ran.
`Result`	*Type:* text. Nullable The result of the recognition rule script.

ComplianceComputerStatus Table

`ComplianceComputerStatus` is a static table used to define possible values for the status of computers reported in

FlexNet Manager Suite.

Table 137: Database columns for `ComplianceComputerStatus` table

Database Column	Details
`ComplianceComputer StatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceComputerStatus`. Possible values and the corresponding default strings are: • 1 = New (this is the first appearance of this computer in inventory) • 2 = Ignored (an operator has marked this computer to be ignored)
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a computer status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

ComplianceComputerType Table

`ComplianceComputerType` is a static table listing all types of computers that can be created.

Table 138: Database columns for `ComplianceComputerType` table

Database Column	Details
`ComplianceComputerTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceComputerType`. Possible values and the corresponding default strings are: • 1 = Computer • 2 = VM Host • 3 = Virtual Machine • 4 = Remote Device. • 5 = Mobile Device. • 6 = VDI Template.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a computer role. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of computer, stored in XML format.
`CanCreate`	*Type:* boolean. Key Whether the end-user can manually create computers of this type.
`CanEdit`	*Type:* boolean. Key Whether the end-user can manually edit computers of this type.

ComplianceComputerTypeProperty Table

`ComplianceComputerTypeProperty` defines extra custom properties for computers of the specified type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 139: Database columns for `ComplianceComputerTypeProperty` table

Database Column	Details
`ComplianceComputerTypePropertyID`	*Type:* integer. Key. Generated ID A unique identifier for a property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.
`ComplianceComputerTypeID`	*Type:* integer. Key Computer type with which this property is associated. Foreign key to the `ComplianceComputerType` table.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

ComplianceComputerUsage Table

This table links user IDs with computer IDs, allowing ECM to determine who uses a computer most frequently; and this is one factor in determining the assigned user for a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 140: Database columns for `ComplianceComputerUsage` table

Database Column	Details
`ComplianceComputerUsageID`	*Type:* integer. Key. Generated ID Unique identifier for a `ComplianceComputerUsage` record.
`ComplianceComputerID`	*Type:* integer. Key Foreign key to the `ComplianceComputer` table.
`ComplianceUserID`	*Type:* integer. Key Foreign key to the `ComplianceUser` table.
`DateRecorded`	*Type:* datetime. Key The date and time that the record was inserted.

ComplianceEvent Table

The `ComplianceEvent` table lists all the 'compliance events' that FlexNet Manager Suite has detected. These are any event, such as the arrival of a new application version or a change in primary application for a license, that should trigger recalculation of linked applications through upgrade and downgrade rights. Depending on license properties, some of these events trigger automatic recalculation, and others trigger a proposal to the operator for manual response. This table records the current state for each event, with a history of state changes available in the `ComplianceEventHistory` table. Where the compliance event results in changes to the applications linked to a license, further details are recorded in the `SoftwareLicenseChangeEvent` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 141: Database columns for `ComplianceEvent` table

Database Column	Details
`ComplianceEventID`	*Type:* integer. Key. Generated ID A unique identifier for an event.
`EventTypeID`	*Type:* integer The type of event. Foreign key to the `ComplianceEventType` table.
`Priority`	*Type:* integer. Nullable The priority of the event.

Database Column	Details
`Severity`	*Type:* integer. Nullable The severity of the event.
`EventActionID`	*Type:* integer The proposed action for the event. Foreign key to the `ComplianceEventAction` table.
`EventStateID`	*Type:* integer The current state of the event. Foreign key to the `ComplianceEventState` table.
`UpdatedBy`	*Type:* text (max 200 characters) The last operator to update the event.
`UpdatedDate`	*Type:* datetime The date the event was last updated.

ComplianceEventAction Table

The `ComplianceEventAction` table holds the list of possible actions in the handling of 'compliance events'. These are any event, such as the arrival of a new application version or a change in primary application for a license, that should trigger recalculation of linked applications through upgrade and downgrade rights.

Table 142: Database columns for `ComplianceEventAction` table

Database Column	Details
`EventActionID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceEventAction`. Possible values and the corresponding default strings are: • 1 = Notification (the event is automatically managed, and the operator is to be advised of the result) • 2 = Request for Action (the license is not managed automatically, and the operator receives a suggested action).
`EventActionResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an event type. Foreign key to the `ComplianceResourceString` table.
`EventActionDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ComplianceEventHistory Table

`ComplianceEventHistory` stores a history of state changes for each compliance event.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 143: Database columns for `ComplianceEventHistory` table

Database Column	Details
`ComplianceEventHistoryID`	*Type:* integer. Key. Generated ID Unique identifier for an event history record.
`ComplianceEventID`	*Type:* integer. Key The event whose history is being recorded. Foreign key to the `ComplianceEvent` table.
`UserName`	*Type:* text (max 60 characters) The operator who made the change.
`HistoryDate`	*Type:* datetime The date of the change.
`FieldName`	*Type:* text (max 256 characters). Nullable The field name that has been updated. Foreign key to the `ComplianceResourceString` table.
`OldValue`	*Type:* text (max 500 characters). Nullable The value before the change.
`NewValue`	*Type:* text (max 500 characters). Nullable The value after the change.

ComplianceEventState Table

`ComplianceEventState` is a static table holding all possible event states.

Table 144: Database columns for `ComplianceEventState` table

Database Column	Details
`EventStateID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceEventState`. Possible values and the corresponding default strings are: • 1 = New (action needs to be taken for this event) • 2 = Postponed (no action needs to be taken at this time) • 3 = Accepted (the proposed action has been taken for this event) • 4 = Rejected (the proposed action will not be taken).
`EventStateResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an event state. Foreign key to the `ComplianceResourceString` table.
`EventStateDefaultValue`	*Type:* text (max 100 characters) The text to display if the state resource string has no translation.

ComplianceEventType Table

`ComplianceEventType` is a static table that holds all possibles types of event.

Table 145: Database columns for `ComplianceEventType` table

Database Column	Details
`EventTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceEventType`. Reserved for future expansion. Possible values and the corresponding default strings are: • 1 = Software License Change.
`EventTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an event type. Foreign key to the `ComplianceResourceString` table.
`EventTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ComplianceHistory Table

The `ComplianceHistory` table records changes to many entities used in FlexNet Manager Suite. This table has a

series of ID columns, any one (or sometimes more) of which may be set to associate the history with a particular item. These ID columns no longer have foreign keys to other tables. This allows us to retain history of deleted objects in order to maintain an audit trail (as yet, there is no UI around this information), and also to improve performance when deleting objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 146: Database columns for `ComplianceHistory` table

Database Column	Details
`ComplianceHistoryID`	*Type:* big integer. Key. Generated ID Unique identifier for a history record.
`AssetID`	*Type:* integer. Key. Nullable ID from the `Asset` table.
`ComplianceComputerID`	*Type:* integer. Key. Nullable ID from the `ComplianceComputer` table.
`ContractID`	*Type:* integer. Key. Nullable ID from the `Contract` table.
`VendorID`	*Type:* integer. Key. Nullable ID from the `Vendor` table.
`VirtualMachineID`	*Type:* integer. Nullable ID from the `VirtualMachine` table.
`PurchaseOrderID`	*Type:* integer. Nullable ID from the `PurchaseOrder` table.
`PurchaseOrderDetailID`	*Type:* integer. Key. Nullable ID from the `PurchaseOrderDetail` table.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable ID from the `SoftwareLicense` table
`SoftwareTitleID`	*Type:* integer. Key. Nullable ID from the `SoftwareTitle` table
`PaymentScheduleID`	*Type:* integer. Key. Nullable ID from the `PaymentSchedule` table
`InstanceID`	*Type:* integer. Key. Nullable ID from the `Instance` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key. Nullable ID from the `ComplianceUser` table
`ComplianceOperatorID`	*Type:* integer. Nullable ID from the `ComplianceOperator` table
`DocumentID`	*Type:* integer. Key. Nullable ID from the `Document` table
`DocumentNoteID`	*Type:* integer. Nullable ID from the `DocumentNote` table
`ContractNoteID`	*Type:* integer. Nullable ID from the `ContractNote` table
`ProjectID`	*Type:* integer. Key. Nullable ID from the `Project` table
`FieldName`	*Type:* text (max 256 characters). Nullable The field name that has been updated. Foreign key to the `ComplianceResourceString` table.
`OldValue`	*Type:* text (max 4000 characters). Nullable Typically the value before the change, although at times, when multiple pieces of information are required to identify the action taking place, this field may store other supporting information. For example, when an operator is granted rights to access a contract, this field stores the type of access (such as “Normal” or “Administrator”) while the `NewValue` field stores the name of the contract.
`NewValue`	*Type:* text (max 4000 characters). Nullable Typically the value after the change, although refer to the above definition of the `OldValue` column for a description of extenuating circumstances.
`NeedsApproval`	*Type:* boolean Set this field to `True` if the change requires approval. Used usually to track changes to computer hardware.
`ValuesAreResourceStrings`	*Type:* boolean Set this field to `True` if the old and new values should be looked up as resource strings.
`ComplianceHistoryTypeID`	*Type:* integer Foreign key to the `HistoryType` table.
`UserName`	*Type:* text (max 60 characters) The operator who made the change.

Database Column	Details
`HistoryDate`	*Type:* datetime. Key The date of the change.
`Comments`	*Type:* text (max 2000 characters). Nullable Comments recorded about the change after it was made.

ComplianceHistoryColumn Table

The `ComplianceHistoryColumn` table lists the fields (columns) for which history details can be recorded.

Table 147: Database columns for `ComplianceHistoryColumn` table

Database Column	Details
`ComplianceHistoryColumnID`	*Type:* integer. Key. Generated ID A unique identifier for a history column.
`TableName`	*Type:* text (max 128 characters). Key The name of the database table to which the history-record settings apply. This may have a suffix of .1 or .2. These suffixes are used for grouping purposes. Do not edit this field.
`ColumnName`	*Type:* text (max 128 characters). Key A description of the column in the specified `TableName` for which the history record settings apply. If this row relates to an entire table, the `ColumnName` will contain the word “History”, for example, “Asset History” or “Contract History”.
`BitwiseValue`	*Type:* integer. Key The bitwise value uniquely identifies each row relating to a single `TableName`. Typically, a value of 1 indicates that this row relates to an entire table. A value greater than 1 indicates that this row relates to a single field in the table. Do not edit this field.
`RecordHistory`	*Type:* boolean Boolean field to indicate if history should be recorded. Set this value to `1` (`True`) to record history details. Set this value to `0` (`False`) if no history details should be recorded.

ComplianceHistoryType Table

`ComplianceHistoryType` is a static table listing all valid types of history records.

Table 148: Database columns for `ComplianceHistoryType` table

Database Column	Details
`ComplianceHistoryTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a history type.

Database Column	Details
`ComplianceHistoryType Description`	*Type:* text (max 100 characters) A unique identifier for each `ComplianceHistoryType`. Possible values and the corresponding default strings are: • 1 = Insert • 2 = Delete • 3 = Update • 4 = Link • 5 = Unlink • 6 = Allocated • 7 = Unallocated • 8 = Assigned • 9 = Unassigned • 10 = Operator unlinked from user due to duplicate login (operator history) • 11 = Operator unlinked from user due to duplicate login (user history) • 12 = Rights to contract granted • 13 = Rights to contract updated • 14 = Rights to contract removed • 15 = Rights to document granted • 16 = Rights to document updated • 17 = Rights to document removed • 18 = Receives (referring to escalations or alerts) • 19 = No longer receives (referring to escalations or alerts) • 20 = Assigned responsibility • 21 = Unassigned responsibility • 22 = Final state of entity when deleted • 23 = Rights to contract removed because contract was deleted • 24 = Rights to document removed because document was deleted • 25 = No longer receives (referring to escalations or alerts) because entity deleted • 26 = Unassigned responsibility because entity was deleted

Database Column	Details
	• 27 = Responsibility type changed.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a history type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ComplianceImage Table

The `ComplianceImage` table stores a collection of images to use on property display dialogs.

Table 149: Database columns for `ComplianceImage` table

Database Column	Details
`ComplianceImageName`	*Type:* text (max 50 characters). Key The name of the image.
`ComplianceImageFile`	*Type:* text The binary representation of the image.

ComplianceLicenseUser Table

If external end-users, reported by systems such as SAP and stored in the `LicenseUser` table, can be matched to existing end-users in the enteprise (stored in the `ComplianceUser` table), the link between them is recorded in the `ComplianceLicenseUser` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 150: Database columns for `ComplianceLicenseUser` table

Database Column	Details
`LicenseUserID`	*Type:* integer. Key A unique identifier for the external end-user. Foreign key to the `LicenseUser` table.

Database Column	Details
ComplianceUserID	*Type:* integer. Key A unique identifier for the end-user in the enterprise. Foreign key to the ComplianceUser table.

CompliancePredefinedSearch Table

The CompliancePredefinedSearch holds a list of the predefined asset and licenses searches available to the operator. Each predefined search has its own grid in the FlexNet Manager Suite UI, and is accessed from a node which is a child of either Licenses or Assets nodes.

Table 151: Database columns for CompliancePredefinedSearch table

Database Column	Details
CompliancePredefined SearchID	*Type:* integer. Key. Generated ID A unique identifier for each CompliancePredefinedSearch. Possible values and the corresponding default strings are: • 1 = New Inventory • 2 = Changed Assets • 3 = Lease Expiry • 4 = Warranty Expiry • 5 = Missing Computers • 6 = License At Risk • 7 = License Expiry • 8 = License Contract Expiry • 9 = License Unused • 10 = UnLicensed Apps • 11 = UnLicensed Installs • 12 = License Group At Risk • 13 = License Upgrade Downgrade.
SearchNameResource	*Type:* text (max 128 characters). Key Resource string identifying the predefined search.
SearchNameDefault	*Type:* text (max 128 characters) The name of the predefined search.

Database Column	Details
`AmberThreshold`	*Type:* integer Indicates when the amber state should be shown in the related traffic light summary.
`RedThreshold`	*Type:* integer Indicates when the red state should be shown in the related traffic light summary.
`DateSearch`	*Type:* boolean. Key `True` indicates that the search is date based. `False` means count based.
`ComplianceSearchType`	*Type:* text (max 128 characters). Key. Nullable Type of search. Matches the name of a row in the `ComplianceSearchType` table.

ComplianceResponsibility Table

`ComplianceResponsibility` links end-users to a contract with various responsibility types.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 152: Database columns for `ComplianceResponsibility` table

Database Column	Details
`Compliance ResponsibilityID`	*Type:* integer. Key. Generated ID A unique identifier for a record.
`ResponsibilityTypeID`	*Type:* integer The particular type of responsibility. Foreign key to the `ResponsibilityType` table.
`ContractID`	*Type:* integer. Key The contract for which this end-user has some responsibility. Foreign key to the `Contract` table.
`ComplianceUserID`	*Type:* integer. Key The end-user who has this responsibility for (or relationship to) the contract. Foreign key to the `ComplianceUser` table.
`Comment`	*Type:* text (max 500 characters). Nullable Any operator comments related to the user responsibility.

ComplianceSavedSearch Table

The `ComplianceSavedSearch` table holds the name of a custom view and any descriptive information about it.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 153: Database columns for `ComplianceSavedSearch` table

Database Column	Details
`ComplianceSavedSearchID`	*Type:* integer. Key. Generated ID A unique identifier for a custom view.
`SearchName`	*Type:* text (max 64 characters). Nullable The name of the custom view.
`Description`	*Type:* text (max 1000 characters). Nullable A description of the custom view.
`SearchGridLayout`	*Type:* text. Nullable The grid layout used in the custom view.
`SearchSQL`	*Type:* text. Nullable SQL statement that generates the data set for the custom view.
`SearchSQLConnection`	*Type:* text (max 500 characters) SQL connection to use to execute search SQL: 'Live', 'DataWarehouse', 'QuerySnapshot', 'ExternalFNMEA', or connection string.
`SearchMapping`	*Type:* XML. Nullable Search query XML to SQL mapping.
`SearchXML`	*Type:* XML. Nullable Search query XML.
`CreatedBy`	*Type:* text (max 128 characters) The operator who created the custom view.
`CreationDate`	*Type:* datetime The date the custom view was created.
`ModifiedBy`	*Type:* text (max 128 characters). Nullable The operator who last modified the custom view.
`ModificationDate`	*Type:* datetime. Nullable The date the custom view was last modified.

Database Column	Details
`ComplianceSearchTypeID`	*Type:* integer. Key The type of the custom view. Foreign key to the `ComplianceSearchType` table.
`ComplianceSearchFolderID`	*Type:* integer. Key The folder in which this custom view is stored. Foreign key to the `ComplianceSearchFolder` table.
`CreatedByOperatorID`	*Type:* integer. Key. Nullable ID of the operator who created the view. Foreign key to the `ComplianceOperator` table.
`RestrictedAccessTypeID`	*Type:* integer. Key Defined access type to the view. Foreign key to the `RestrictedAccessType` table.
`CanDelete`	*Type:* boolean Set this to `False` for predefined custom views which an operator is not allowed to delete.
`CanChangeMasterObject`	*Type:* boolean Set this to `False` if the this view has a fixed master object.
`ComplianceSavedSearch SystemID`	*Type:* integer. Key. Nullable An identifier for a system custom view.
`SearchNameResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a column name. Foreign key to the `ComplianceResourceString` table.
`DescriptionResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a column name. Foreign key to the `ComplianceResourceString` table.
`SavedSearchLink`	*Type:* text. Nullable The saved built in report or view link.
`SavedSearchFilter`	*Type:* text. Nullable The saved filter for report or view

ComplianceSchedule Table

`ComplianceSchedule` defines schedules that take place repeatedly at a specified interval.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 154: Database columns for `ComplianceSchedule` table

Database Column	Details
`ComplianceScheduleID`	*Type:* integer. Key. Generated ID A unique identifier for the schedule.
`TermAndConditionID`	*Type:* integer. Key. Nullable The term/condition that the schedule is associated with. Foreign key to the `TermAndCondition` table.
`StartDate`	*Type:* datetime The date on which this schedule first applies.
`EndDate`	*Type:* datetime The date on which this schedule ends.
`RepeatIntervalTypeID`	*Type:* integer. Key. Nullable The type of repeat interval. Foreign key to the `IntervalType` table.
`RepeatInterval`	*Type:* integer. Nullable The interval between repeats of this schedule.

ComplianceSearchFolder Table

The `ComplianceSearchFolder` table identifies a folder for storing a custom search (or view), and tracks the parent-child relationships of folders to establish their hierarchy.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 155: Database columns for `ComplianceSearchFolder` table

Database Column	Details
`ComplianceSearchFolderID`	*Type:* integer. Key. Generated ID A unique identifier for a saved search folder.
`Name`	*Type:* text (max 128 characters). Key. Nullable The name of the folder.
`ParentFolderID`	*Type:* integer. Key. Nullable Identifies the parent that contains this folder. Foreign key to another folder in this `ComplianceSearchFolder` table.

Database Column	Details
`ComplianceSearchTypeID`	*Type:* integer. Key The kind of custom view stored in this folder. Foreign key to the `ComplianceSearchType` table.
`Path`	*Type:* text (max 128 characters). Key. Nullable The internal path to the folder.
`PredefinedSearchesCreated`	*Type:* boolean. Nullable Set this field to `True` to indicate that this folder holds generated searches.
`CanDelete`	*Type:* boolean. Nullable Set this field to `False` for predefined folders which operators are not allowed to deleted.
`CreatedByOperatorID`	*Type:* integer. Key. Nullable ID of the operator who created the view. Foreign key to the `ComplianceOperator` table.
`RestrictedAccessTypeID`	*Type:* integer. Key Defined access type to the view. Foreign key to the `RestrictedAccessType` table.
`ComplianceSearchFolderSystemID`	*Type:* integer. Key. Nullable An identifier for a system custom view folder.
`NameResourceName`	*Type:* text (max 256 characters). Key. Nullable The unique name of the localizable resource string representing a folder name. Foreign key to the `ComplianceResourceString` table.

ComplianceSearchType Table

`ComplianceSearchType` is a static table holding the name of the basic objects, such as an asset or license, for which custom views can be created.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 156: Database columns for `ComplianceSearchType` table

Database Column	Details
`ComplianceSearchTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a type of compliance search. Possible values, together with the associated names of the object being searched, are: • -1 = Custom • 1 = Asset • 2 = License • 3 = Contract • 4 = Vendor • 5 = PurchaseOrder • 6 = SoftwareTitle • 7 = User • 8 = Computer • 13 = PurchaseOrderDetail • 14 = VirtualMachine • 15 = InstalledSoftware • 16 = SoftwareLicenseAllocation • 17 = PaymentSchedule • 18 = PaymentScheduleDetail • 19 = OracleInstance • 20 = OracleComponent • 21 = Suite • 22 = SuiteMember • 23 = TermAndCondition • 24 = ContractHistoryView • 25 = ContractDocumentView • 26 = DocumentNote • 27 = ComplianceResponsibility • 28 = ContractNote • 29 = Location

Database Column	Details
	• 30 = CostCenter • 31 = CorporateStructure • 32 = Category • 33 = VendorContact • 34 = Cluster • 35 = CloudServiceInstance.
`TypeName`	*Type:* text (max 64 characters). Key The name of the objects being searched.
`TypeNameResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a type name. Foreign key to the `ComplianceResourceString` table.
`QuerySetup`	*Type:* text. Nullable Query pre-calculation statement executed before custom view query.
`QueryFilter`	*Type:* text. Nullable Query filter template executed before custom view query.
`QueryTemplate`	*Type:* text. Nullable Query template for this search type.
`IsCustom`	*Type:* boolean False if the relation is out of the box, false otherwise.

ComplianceSearchTypeColumn Table

The `ComplianceSearchTypeColumn` table identifies all columns that may be used in custom views.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 157: Database columns for `ComplianceSearchTypeColumn` table

Database Column	Details
`ComplianceSearchType ColumnID`	*Type:* integer. Key. Generated ID A unique identifier for a custom view column.

Database Column	Details
`ColumnName`	*Type:* text (max 128 characters). Key The default value of the display column name.
`ColumnNameResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a column name. Foreign key to the `ComplianceResourceString` table.
`QuerySetup`	*Type:* text. Nullable Query pre-calculation statement executed before the custom view query.
`FromTable`	*Type:* text. Nullable The name of the database table where the column can be found.
`SelectName`	*Type:* text. Nullable The name of the column in the database.
`JoinClause`	*Type:* text. Nullable The SQL join that links other tables to provide the relevant data for this column.
`WhereClause`	*Type:* text. Nullable The SQL “WHERE” statement that limits the information returned by the custom view.
`SelectOptionsSQL`	*Type:* text. Nullable The SQL that selects the predefined list that the user can display when filtering on this column.
`FilterGroupType`	*Type:* integer. Nullable An ID that indicates the kind of value expected in this column, which in turn determines what kinds of filter options (such as `Contains`, `Starts With`) will be offered for this column. Possible values (and their associated meanings) are: • 1 = string • 2 = number • 3 = list • 4 = date • 5 = group • 6 = money • 7 = boolean.

Database Column	Details
`DefaultFilterType`	*Type:* integer. Nullable The type of field that should be used to search for information in this column. Possible values (and their associated meanings) are the same as for the previous field.
`ComplianceSearchTypeID`	*Type:* integer. Key The type of that the column is related to. Foreign key to the `ComplianceSearchType` table.
`RequiresSearchTypeID`	*Type:* integer. Nullable For special cases, a column may need data from another compliance object as well. Foreign key to the `ComplianceSearchType` table.
`Mandatory`	*Type:* boolean Set this field to `True` if this column must always be returned in the SQL “SELECT” statement.
`PrimaryKey`	*Type:* boolean Set this field to `True` if this column is the primary key of the SQL “SELECT” statement.
`SelectByDefault`	*Type:* boolean Set this field to `True` if this column should be included (checked) by default when the operator is creating a custom view. If `False`, the operator may include it manually.
`IsCustom`	*Type:* boolean False if the relation is out of the box, false otherwise.
`LinkAction`	*Type:* text (max 64 characters). Nullable The action to be used for the drill through link on this column.
`LinkController`	*Type:* text (max 64 characters). Nullable The controller to be used for the drill through link on this column.
`LinkIndicateOrigin`	*Type:* boolean Whether the drill through link on this column contains the report page URL as the origin URL.
`LinkFragmentField`	*Type:* text (max 64 characters). Nullable The fragment field name to be used for the drill through link on this column.
`IsMultiEditEnabled`	*Type:* boolean Whether the multiple object drill through is enabled on this object type.
`MultiEditConditionField`	*Type:* text. Nullable Field on which the multiple object drill through will be evaluated against.

ComplianceSearchTypeRelation Table

The `ComplianceSearchTypeRelation` table tracks relationships between different objects for which operators can create custom views.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 158: Database columns for `ComplianceSearchTypeRelation` table

Database Column	Details
`ComplianceSearchTypeRelationID`	*Type:* integer. Key. Generated ID A unique identifier for a relationship.
`RelationName`	*Type:* text (max 256 characters). Key The unique internal name of this relation.
`DescriptionResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a relationship name. Foreign key to the `ComplianceResourceString` table.
`DescriptionDefault`	*Type:* text (max 256 characters) The default description of the relationship.
`FromSearchTypeID`	*Type:* integer. Key The `ComplianceSearchType` that represents the source of the relationship.
`ToSearchTypeID`	*Type:* integer. Key The `ComplianceSearchType` that represents the destination of the relationship.
`ToMany`	*Type:* boolean Set this field to `True` to allow more than one related row in the destination table for each row in the source table. If this field is `False`, rows have a one-to-one relationship.
`JoinClause`	*Type:* text The SQL join clause used to join the source object with a related object.
`FilterClause`	*Type:* text The SQL filter clause used to filter the source object with a related object.
`IsCustom`	*Type:* boolean False if the relation is out of the box, false otherwise.

ComplianceTask Table

`ComplianceTask` holds a collection of tasks, which are audit responsibilities generated by settings on a `TermAndCondition`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 159: Database columns for `ComplianceTask` table

Database Column	Details
`ComplianceTaskID`	*Type:* integer. Key. Generated ID A unique identifier for the task.
`ComplianceScheduleID`	*Type:* integer. Key. Nullable The schedule the task is associated with. Foreign key to the `ComplianceSchedule` table.
`TaskDate`	*Type:* datetime. Nullable The date for the task.

ComplianceTenantSettingHistory Table

The `ComplianceTenantSettingHistory` table records changes to many tenant settings. This table has a series of ID columns, any one (or sometimes more) of which may be set to associate the history with a particular item. These ID columns don't have foreign keys to other tables. This allows us to retain history of deleted objects in order to maintain an audit trail (as yet, there is no UI around this information), and also to improve performance when deleting objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 160: Database columns for `ComplianceTenantSettingHistory` table

Database Column	Details
`ComplianceTenant SettingHistoryID`	*Type:* big integer. Key. Generated ID Unique identifier for a history record.
`ComplianceOperatorID`	*Type:* integer. Nullable ID from the `ComplianceOperator` table
`SettingNameID`	*Type:* integer. Nullable The setting ID that has been updated.

Database Column	Details
`AvailablePackageID`	*Type:* integer. Nullable The package ID that has been updated.
`FieldName`	*Type:* text (max 256 characters). Nullable The field name that has been updated. Foreign key to the `ComplianceResourceString` table.
`OldValue`	*Type:* text (max 4000 characters). Nullable Typically the value before the change, although at times, when multiple pieces of information are required to identify the action taking place, this field may store other supporting information. For example, when an operator is granted rights to access a contract, this field stores the type of access (such as “Normal” or “Administrator”) while the `NewValue` field stores the name of the contract.
`NewValue`	*Type:* text (max 4000 characters). Nullable Typically the value after the change, although refer to the above definition of the `OldValue` column for a description of extenuating circumstances.
`ValuesAreResourceStrings`	*Type:* boolean Set this field to `True` if the old and new values should be looked up as resource strings.
`ComplianceHistoryTypeID`	*Type:* integer Foreign key to the `HistoryType` table.
`HistoryDate`	*Type:* datetime The date of the change.
`Comments`	*Type:* text (max 2000 characters). Nullable Comments recorded about the change after it was made.

ComplianceUserPropertyValue Table

For each end-user, `ComplianceUserPropertyValue` stores the values for the custom properties defined in `ComplianceUserTypeProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 161: Database columns for `ComplianceUserPropertyValue` table

Database Column	Details
`ComplianceUserPropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for the property value.
`ComplianceUserTypePropertyID`	*Type:* integer. Key The property whose value is being stored. Foreign key to the `ComplianceUserTypeProperty` table.
`ComplianceUserID`	*Type:* integer. Key The end-user associated with this property value. Foreign key to the `ComplianceUser` table.
`PropertyValue`	*Type:* text (max 4000 characters) The value of the property for the specified `ComplianceUser`.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

ComplianceUserTypeProperty Table

`ComplianceUserTypeProperty` defines extra custom properties for all end-users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 162: Database columns for `ComplianceUserTypeProperty` table

Database Column	Details
`ComplianceUserTypePropertyID`	*Type:* integer. Key. Generated ID A unique identifier for the property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.

Database Column	Details
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

ComputerChassisType Table

`ComputerChassisType` is a static table listing all possible computer chassis (case) types.

Table 163: Database columns for `ComputerChassisType` table

Database Column	Details
`ChassisTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComputerChassisType`. Possible values and the corresponding default strings are: • 1 = Other • 2 = Unknown • 3 = Desktop • 4 = Low Profile Desktop • 5 = Pizza Box • 6 = Mini Tower • 7 = Tower • 8 = Portable • 9 = Laptop • 10 = Notebook • 11 = Other Hand Held • 12 = Docking Station • 13 = All in One • 14 = Sub Notebook • 15 = Space-Saving • 16 = Lunch Box • 17 = Main System Chassis • 18 = Expansion Chassis • 19 = Sub-Chassis • 20 = Bus Expansion Chassis • 21 = Peripheral Chassis • 22 = Storage Chassis • 23 = Rack Mount Chassis • 24 = Sealed-Case PC. • 25 = Smart Phone • 26 = Tablet

Database Column	Details
`WMIChassisTypeID`	*Type:* integer. Nullable The identifier for the chassis type identified in WMI.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a computer role. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 128 characters) The text to display if the chassis type resource string has no translation.
`IncludeInLicenseRec SecondUseDefault`	*Type:* boolean Determines whether or not a second installation of an application on a computer of this chassis type (as well as on a primary computer assigned to the same end-user) may be counted as a legal second use under the Right of Second Use granted by some licenses. Currently, this field is used to group together chassis types that can be treated as “laptops” for this purpose.
`SecondUseBitwiseValue`	*Type:* integer Reserved for future use. Do not edit.

ConsolidatedLicenseUser Table

This table stores the data specific to a consolidated license user.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 164: Database columns for `ConsolidatedLicenseUser` table

Database Column	Details
`ConsolidatedLicenseUserID`	*Type:* integer. Key. Generated ID A unique identifier for the consolidated license user.
`LicenseUserID`	*Type:* integer Foreign key to the `LicenseUser` table.
`ConsolidatedGroupNumber`	*Type:* integer The unique identifier showing which users are duplicates of one another.
`ConsolidatedName`	*Type:* text The name of the consolidated user. If consolidated by rules engine, this column stores the name of the user with the lowest LicenseUserID

Database Column	Details
`ConsolidationTypeID`	*Type:* integer Foreign key to the ConsolidationType table.

ConsolidationType Table

This table stores consolidation type.

Table 165: Database columns for `ConsolidationType` table

Database Column	Details
`ConsolidationTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the consolidation type.
`ResourceName`	*Type:* text (max 256 characters). Key. Nullable A localizable resource string representing a consolidation type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the consolidation type resource string has no translation.

Contract Table

The `Contract` table contains a list of all the contracts in the system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 166: Database columns for `Contract` table

Database Column	Details
`ContractID`	*Type:* integer. Key. Generated ID A unique identifier for the contract.
`ContractNo`	*Type:* text (max 60 characters) The contract number assigned by the operator.
`ContractName`	*Type:* text (max 100 characters) A contract name assigned by the operator.

Database Column	Details
`ContractTypeID`	*Type:* integer. Key Identifies the type of contract. Foreign key to the `ContractType` table.
`ContractStatusID`	*Type:* integer Identifies the status of the contract. Foreign key to the `ContractStatus` table.
`NeverExpires`	*Type:* boolean If set to `True`, this contract never expires. If `False`, the contract expires at the date specified in the `EndDate` field.
`StartDate`	*Type:* datetime. Nullable The start date of the contract.
`EndDate`	*Type:* datetime. Nullable The end date of the contract.
`PreExpiryDate`	*Type:* datetime. Nullable The date at which a contract should be reviewed prior to its expiry date.
`RenewalDate`	*Type:* datetime. Nullable The date at which a contract is due to be renewed.
`Price`	*Type:* currency. Nullable The price of the contract.
`PriceRateID`	*Type:* integer. Nullable The currency rate to be applied to the above contract price. Foreign key to the `CurrencyRate` table.
`PeriodTypeID`	*Type:* integer. Nullable The frequency with which the period payments are applicable. Foreign key to the `PeriodType` table.
`BuyoutCost`	*Type:* currency. Nullable The buyout cost of the contract.
`BuyoutCostRateID`	*Type:* integer. Nullable The currency rate to be applied to the above buyout cost. Foreign key to the `CurrencyRate` table.
`ManagerID`	*Type:* integer. Key. Nullable The person who manages the contract. Foreign key to the `ComplianceUser` table.
`Comments`	*Type:* text. Nullable Comments recorded about the contract.

Database Column	Details
`PeriodicPayment`	*Type:* currency. Nullable The price of periodic payments associated with this contract.
`PeriodicPaymentRateID`	*Type:* integer. Nullable The currency rate to be applied to the periodic payments figure above. Foreign key to the `CurrencyRate` table.
`VendorID`	*Type:* integer. Key. Nullable The vendor with which the contract agreement has been made. Foreign key to the `Vendor` table.
`MasterContractID`	*Type:* integer. Key. Nullable The contract that is the master of this contract. Foreign key to another contract in this `Contract` table.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this contract. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise corporate unit associated with this contract. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise cost center associated with this contract. Foreign key to the `GroupEx` table.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any category used in this enterprise that is associated with this contract. Foreign key to the `GroupEx` table.
`LicenseDowngradeEnabled`	*Type:* boolean If this field is set to `True`, licenses can inherit downgrade rights from this contract. If `False` (the default), licenses cannot inherit downgrade rights.
`LicenseDowngradeToVersion`	*Type:* boolean If this field is set to `True`, any license inheriting downgrade rights from this contract can cover all previous releases (with the same edition) of the primary application. If `False`, licenses inheriting downgrade rights may not downgrade to earlier versions.
`LicenseDowngradeToEdition`	*Type:* boolean If this field is set to `True`, any license inheriting downgrade rights from this contract can cover all lower editions of this version of the primary application. If `False`, licenses inheriting downgrade rights may not downgrade to lower editions.

Database Column	Details
`LicenseUpgradeEnabled`	*Type:* boolean If this field is set to `True`, licenses can inherit upgrade rights from this contract. If `False` (the default), licenses cannot inherit upgrade rights.
`LicenseUpgradeToVersion`	*Type:* boolean If this field is set to `True`, any license inheriting upgrade rights from this contract can cover all later releases (with the same edition) of the primary application. If `False`, licenses inheriting upgrade rights may not upgrade to later versions.
`LicenseUpgradeUntil ContractExpiry`	*Type:* boolean If this field is set to `True`, any license inheriting upgrade rights from this contract can cover all later releases (with the same edition) of the primary application, as long as they were released before the expiry date (`EndDate`) of the contract. If `False`, licenses inheriting upgrade rights do not take the application release date into consideration.
`GrantSecondUseToLicense`	*Type:* boolean If this field is set to `True`, licenses can inherit the right of second use from this contract. If `False` (the default), licenses cannot inherit the right of second use.
`SecondUsageWorkLaptop`	*Type:* boolean If this field is set to `True`, any license inheriting from this contract will confer the right of second use on a work laptop. If `False`, licenses inheriting from this contract will not confer the right of second use.
`SecondUsageAtHome`	*Type:* boolean If this field is set to `True`, any license inheriting from this contract will confer the right of second use on a home computer, for the same end-user as the primary end-user of the license entitlement consumed at work. If `False`, licenses inheriting from this contract will not confer the right of second use on a home computer.
`GrantVirtualInstallsTo License`	*Type:* boolean If this field is set to `True`, licenses can inherit the virtual machine licensing rights from this contract. If `False` (the default), licenses cannot inherit virtual machine licensing rights.
`CoverInstallsOnVirtual Machines`	*Type:* boolean If this field is set to `True`, any license inheriting virtual machine rights from this contract may be used to account for installations on virtual machines. If `False`, licenses inheriting virtual machine rights may only account for installations on physical machines.

Database Column	Details
`LimitNumberOfVirtual Installs`	*Type:* boolean If this field is set to `True`, there is a limit to the number of virtual machine installations that may be covered by any license inheriting virtual machine rights from this contract. If this field is `False`, one license entitlement may cover any use on virtual machines (typically within one host computer).
`NumberOfAllowedVirtual Installs`	*Type:* integer. Nullable If this contracts confers the right for an inheriting license to cover installations on virtual machines, this field specifies how many installations per host are allowed before an additional license entitlement (or point) is consumed.
`LimitVirtualInstalls IncludesHost`	*Type:* boolean If this field is `True`, the host operating system installations are included in the overall count of operating systems on the host when there is a limit on the number of allowed virtual installs for each license. If `False`, the host operating system is not considered when determining virtual install limits.
`UseHostProcessor Information`	*Type:* boolean If virtual installs are allowed, this field controls whether host information is used by an inheriting license when calculating the license points consumed.
`GrantLimitPointsToLicense`	*Type:* boolean If this field is set to `True`, licenses can inherit the right of multiple use from this contract. If `False` (the default), licenses cannot inherit the right of multiple use.
`LimitNumberOf ApplicationsEach LicensePointCovers`	*Type:* boolean If this field is set to `True`, there is a limit, for any inheriting license, to the number of application installations allowed per license entitlement (or point). If this bit is `False` (the default), an inheriting license entitles you to any number of installations of software linked to this license on the one computer.
`NumberOfApplication InstallsAllowedPer LicensePoint`	*Type:* integer. Nullable Where the previous field is set to `True`, this column defines the limited number of application installations allowed per entitlement (or point).
`LimitNumberOfComputers UserLicenseCanBe InstalledOn`	*Type:* boolean If this field is set to `True`, there is a limit, for an inheriting user-based license, to the number of computers that an end-user can use per entitlement (or point) consumed. If this field is `False` (the default), a single end-user is entitled to install related software for his/her own use on any number of computers.
`NumberOfComputers AllowedPerUserLicense Point`	*Type:* integer. Nullable Where the previous field is set to `True`, this column defines the limited number of application installations an end-user is allowed per entitlement (or point).

Database Column	Details
`InitialPlatformQuantity`	*Type:* integer. Nullable The number of desktops covered by the Microsoft Enterprise Agreement platform license at the start of the agreement.
`PurchaseProgramID`	*Type:* integer. Nullable Identifies the purchase program of contract. Foreign key to the `PurchaseProgram` table.
`MSSelectApplication LevelID`	*Type:* integer. Nullable Identifies the Microsoft Select level for applications. Foreign key to the `MSSelectLevel` table.
`MSSelectSystemLevelID`	*Type:* integer. Nullable Identifies the Microsoft Select level for systems. Foreign key to the `MSSelectLevel` table.
`MSSelectServerLevelID`	*Type:* integer. Nullable Identifies the Microsoft Select level for servers. Foreign key to the `MSSelectLevel` table.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`TotalValue`	*Type:* currency. Nullable The total value of the contract.
`TotalValueRateID`	*Type:* integer. Nullable The rate for the total value. Foreign key to the `CurrencyRate` table.
`MonthlyValue`	*Type:* currency. Nullable The cost of the contract per month.
`MonthlyValueRateID`	*Type:* integer. Nullable The rate for the monthly cost. Foreign key to the `CurrencyRate` table.
`ProjectID`	*Type:* integer. Key. Nullable A project for the `Contract`. Foreign key to the `Project` table.

Database Column	Details
SecurityTypeID	*Type:* integer. Nullable The type of security to use when determining which operators have access to the contract. Foreign key to the SecurityType table.
PreviousContractID	*Type:* integer. Key. Nullable A link to a contract that this contract has replaced. Foreign key to the Contract table.
ContractStateID	*Type:* integer. Nullable The state of the contract. Foreign key to the ContractState table.
LastRenewedDate	*Type:* datetime. Nullable The date when the contract was last renewed.
LicenseConsumptionEnabled	*Type:* boolean If this field is set to True, licenses can inherit consumption rules from this contract. If False (the default), licenses cannot inherit consumption rules.
LicenseMobilityEnabled	*Type:* boolean If this field is set to True, licenses can inherit mobility rights from this contract. If False (the default), licenses cannot inherit mobility rights.
ProcessorLimitsEnabled	*Type:* boolean If this field is set to True, licenses can inherit rights related to processor limits from this contract. If False (the default), licenses cannot inherit rigts related to processor limits.

ContractNote Table

ContractNote stores a list of notes attached to a contract.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 167: Database columns for ContractNote table

Database Column	Details
ContractNoteID	*Type:* integer. Key. Generated ID A unique identifier for the contract note.
ContractID	*Type:* integer. Key The contract that the note is for. Foreign key to the Contract table.

Database Column	Details
`ShortDescription`	*Type:* text (max 100 characters) In the user interface, this maps to the contract reference to which the note relates.
`LongDescription`	*Type:* text. Nullable The content of the note.
`CreationUser`	*Type:* text (max 128 characters) The operator who created the note.
`CreationDate`	*Type:* datetime The date of creation of the note.
`UpdatedUser`	*Type:* text (max 128 characters) The operator who last updated the note.
`UpdatedDate`	*Type:* datetime The date of the last update to the note.

ContractNotification Table

`ContractNotification` lists the notifications that need to be sent for a contract.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 168: Database columns for `ContractNotification` table

Database Column	Details
`ContractNotificationID`	*Type:* integer. Key. Generated ID A unique identifier for the contract notification.
`ContractID`	*Type:* integer. Key The contract this record is associated with. Foreign key to the `Contract` table.
`NotificationInterval`	*Type:* integer Defines how long before the contract notification is sent.
`NotificationInterval TypeID`	*Type:* integer Defines the interval type used to work out how long before a contract notification is sent. Foreign key to the `IntervalType` table.

Database Column	Details
NotificationTypeID	*Type:* integer. Key Defines the type of notification (contract renewal or contract expiry). Foreign key to the NotificationType table.

ContractNotificationResponsibility Table

ContractNotificationResponsibility keeps track of which responsibility groups need to be notified for contract expiry or renewals.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 169: Database columns for ContractNotificationResponsibility table

Database Column	Details
ContractNotificationResponsibilityID	*Type:* integer. Key. Generated ID Unique identifier for contract notification responsibility groups.
ContractID	*Type:* integer. Key The contract generating notifications. Foreign key to the Contract table.
ResponsibilityTypeID	*Type:* integer. Key The responsibility type of the end-users receiving notifications about the contract. Foreign key to the ResponsibilityType table.
NotificationTypeID	*Type:* integer. Key The type of notification (renewal or expiry) that these responsibility groups should receive notifications for. Foreign key to the NotificationType table.

ContractProperty Table

ContractProperty defines extra custom properties for contracts of a specified type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 170: Database columns for `ContractProperty` table

Database Column	Details
`ContractPropertyID`	*Type:* integer. Key. Generated ID Unique identifier for a contract property.
`ContractTypeID`	*Type:* integer. Key The type of contract to which this property may apply. Foreign key to the `ContractType` table.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the custom property. A unique identifier for a resource string. Foreign key to the `ComplianceResourceString` table.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Reference to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

ContractPropertyValue Table

For each contract, `ContractPropertyValue` stores the values for the custom properties defined in `ContractProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 171: Database columns for `ContractPropertyValue` table

Database Column	Details
`ContractPropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`ContractID`	*Type:* integer. Key The individual contract to which this value applies. Foreign key to the `Contract` table.
`ContractPropertyID`	*Type:* integer. Key The property that contains this value. The contract should have the same type as the type associated with this property. Foreign key to the `ContractProperty` table.
`PropertyValue`	*Type:* text (max 4000 characters) The property value.

Database Column	Details
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

ContractScopingData Table

`ContractScoping` links contracts to the enterprise groups to which they apply. Exactly one of GroupExID and CategoryID must be non-NULL.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 172: Database columns for `ContractScopingData` table

Database Column	Details
`ContractID`	*Type:* integer. Key The contract the scoping applies to. Foreign key to the `Contract` table.
`GroupExID`	*Type:* text (max 128 characters). Key. Nullable The enterprise group that the scoping applies to. Foreign key to the `GroupEx` table.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable The category that the scoping applies to. Foreign key to the `Category` table.

ContractSecurityUser Table

`ContractSecurityUser` stores a list of permissions granted to an operator for a contract with `Restricted` security.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 173: Database columns for `ContractSecurityUser` table

Database Column	Details
`ContractID`	*Type:* integer. Key The contract with `Restricted` security. Foreign key to the `Contract` table.
`ActionClassID`	*Type:* integer. Key The type of permission being granted to the operator. Foreign key to the `ActionClass` table.
`ComplianceOperatorID`	*Type:* integer. Key The operator that the permission is granted to. Foreign key to the `ComplianceOperator` table.

ContractState Table

`ContractState` holds the different states a contract can be in.

Table 174: Database columns for `ContractState` table

Database Column	Details
`ContractStateID`	*Type:* integer. Key. Generated ID A unique identifier for each `ContractState`. Possible values and the corresponding default strings are: • 1 = Draft • 2 = Suspended • 3 = Active • 4 = Archived • 5 = Cancelled • 6 = Expired • 7 = Completed.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a contract state. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the state resource string has no translation.

ContractStatus Table

`ContractStatus` is a static table listing all contract status values in the system.

Table 175: Database columns for `ContractStatus` table

Database Column	Details
`ContractStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `ContractStatus`. Possible values and the corresponding default strings are: • 1 = Active • 2 = Archived • 3 = Draft • 4 = Suspended • 5 = Cancelled • 6 = Expired • 7 = Completed.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a contract status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

ContractType Table

`ContractType` is a static table listing all contract types in the system.

Table 176: Database columns for `ContractType` table

Database Column	Details
`ContractTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ContractType`. Possible values and the corresponding default strings are: • 1 = General • 2 = Lease • 3 = Hardware Maintenance and Support • 4 = Software License • 5 = Software Maintenance and Support • 6 = Blanket purchase • 7 = Consulting services • 8 = Insurance • 9 = Rent • 10 = Subscription • 11 = Microsoft Business and Services Agreement • 12 = Microsoft Select License Agreement • 13 = Microsoft Select Plus Agreement • 14 = Microsoft Select License Enrollment • 15 = Microsoft Select Plus Affiliate • 16 = Microsoft Enterprise Agreement • 17 = Microsoft Enterprise Subscription Agreement.
`ContractTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a contract type. Foreign key to the `ComplianceResourceString` table.
`ContractTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of computer, stored in XML format.

Database Column	Details
PathResourceName	*Type:* text (max 256 characters) The unique name of the localizable resource string representing the parent contract type under which this contract type should be displayed. Foreign key to the ComplianceResourceString table.
PathDefaultValue	*Type:* text (max 256 characters) The default parent contract type text to display if the resource string has no translation.
PurchaseProgramID	*Type:* integer. Nullable The default purchase program for this contract type.
CanCreate	*Type:* boolean Whether the end-user can manually create contracts of this type.

ContractUseRight Table

ContractUseRight contains licensing rules most of which can be set by PURL.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 177: Database columns for ContractUseRight table

Database Column	Details
ContractUseRightID	*Type:* integer. Key. Generated ID A unique identifier
ContractID	*Type:* integer. Key A unique identifier for a contract.
ReassignmentTimeLimit AppliesDevice	*Type:* boolean If 1 then the license cannot be reassigned for some period of time (example is Microsoft 90 day rule)
ReassignmentTimeLimit AppliesUser	*Type:* boolean If 1 then the license cannot be reassigned for some period of time (example is Microsoft 90 day rule)
ReassignmentTimeLimit Device	*Type:* integer. Nullable The period (in days) within which the license cannot be reassigned

Database Column	Details
`ReassignmentTimeLimitUser`	*Type:* integer. Nullable The period (in days) within which the license cannot be reassigned
`LicenseMobilityApplies`	*Type:* boolean 1 if eligible for bringing your own license to cloud environment
`NumberOfOSEPerLicense`	*Type:* integer. Nullable Number of OSE per license
`NumberOfProcessorsPerOSE`	*Type:* integer. Nullable Number of processors per OSE
`TotalNumberOfCoresPerVMPerLicense`	*Type:* integer. Nullable Total number of cores per VM per license
`NumberOfCoresPerSocket`	*Type:* integer. Nullable Number of cores per socket
`ThirdPartyAccessAllowed`	*Type:* boolean Access to applications is allowed to third party users. This field is defaulted to `True`
`AllowExternalRoamingUse`	*Type:* boolean. Nullable Set this field to `True` if license allows external roaming use. This field is defaulted to `False`. This is applicable for both device and user licenses and is related to virtual application access. If 1, this license will consume 1 entitlement per each user. If 0, this license will consume 1 license per each user device. And, if NULL, ignore virtual application access. This can be used in conjunction with VirtualApplicationAccessMaximumUsagePeriod.
`MeasurementDate`	*Type:* datetime. Nullable The date of the license measurment.
`ConsumptionUnit`	*Type:* text. Nullable Unit description to describe the consumption amount.
`TargetOperatingSystemTypeID`	*Type:* integer Type of Operating Systems to target
`VirtualApplicationAccessMaximumUsagePeriodDevice`	*Type:* integer. Nullable This is a rule for virtual application access. This is used in conjunction with the AllowExternalRoamingUse. For Device licenses, a license will consume 1 entitlement per each user device when used in period specified here.

Database Column	Details
`VirtualApplication AccessMaximumUsage PeriodUser`	*Type:* integer. Nullable This is a rule for virtual application access. This is used in conjunction with the AllowExternalRoamingUse. For user licenses, if 1, this license will consume only when used in period specified here.
`AlwaysInstalled`	*Type:* boolean If this field is `True`, this license is considered in to be used whenever it is allocated. If `False`, software usage is considered separately, and allocation merely defines the corporation's modelling of who is expected to consume entitlements.
`MinimumNumberOf LicensesPerVM`	*Type:* integer When licensing a Virtual Hardware System with a `MSServerCore` license (`LicenseTypeID` = 33), consume license entitlements as though the virtual machine had at least this number of virtual threads.
`AllowIBMPVUSubCapacity FromNonILMT`	*Type:* boolean If the license does not use host processor information (not full capacity), set this field to `True` to allow non-ILMT sub-capacity PVU consumption calculations to be used.
`NumberOfAllowed ProcessorsPerHost`	*Type:* integer. Nullable This field specifies how many processors per host are allowed before an additional license entitlement (or point) is consumed. Null provides the default of 1. Zero provides unlimited.
`MinimumNumberOfProcessors`	*Type:* integer The minimum number of processors that this license is for. This field is only used where the `SoftwareLicenseType` is `MSServerProcessor` (`LicenseTypeID` = 22).

ContractUseRightIBM Table

`ContractUseRightIBM` contains IBM licensing rules most of which can be set by PURL.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 178: Database columns for `ContractUseRightIBM` table

Database Column	Details
`ContractUseRightIBMID`	*Type:* integer. Key. Generated ID A unique identifier
`ContractID`	*Type:* integer. Key A unique identifier for a contract.
`PVULimitApplies`	*Type:* boolean If 1 then PVU limits apply
`PVULimit`	*Type:* integer. Nullable PVU limit

ContractVendor Table

`ContractVendor` stores the links between vendors and contracts.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 179: Database columns for `ContractVendor` table

Database Column	Details
`ContractVendorID`	*Type:* integer. Key. Generated ID A unique identifier for the link.
`ContractID`	*Type:* integer. Key The contract that the vendor is linked to. Foreign key to the `Contract` table.
`VendorID`	*Type:* integer. Key. Nullable The vendor that the contract is linked to. Foreign key to the `Vendor` table.
`ThirdParty`	*Type:* boolean Set this field to `True` if this vendor is third-party.

CurrencyRate Table

`CurrencyRate` stores the exchange rates assigned to any currency.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 180: Database columns for `CurrencyRate` table

Database Column	Details
`CurrencyRateID`	*Type:* integer. Key. Generated ID Unique identifier for each record.
`SnapshotID`	*Type:* integer. Key Snapshot associated with this exchange rate. Foreign key to the `CurrencyRateSnapshot` table.
`CurrencyID`	*Type:* integer. Key Currency associated with this exchange rate. Foreign key to the `Currency` table.
`Rate`	*Type:* decimal Exchange rate assigned to the currency for the selected snapshot.
`UpdatedUser`	*Type:* text (max 256 characters). Nullable Operator who last modified the record.
`UpdatedDate`	*Type:* datetime. Nullable Date that the record was last modified.

CurrencyRateSnapshot Table

Each record in `CurrencyRateSnapshot` represents a single currency snapshot.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 181: Database columns for `CurrencyRateSnapshot` table

Database Column	Details
`CurrencyRateSnapshotID`	*Type:* integer. Key. Generated ID Unique identifier for this record.
`SnapshotName`	*Type:* text (max 256 characters) Name of the currency snapshot.
`SnapshotResourceID`	*Type:* text (max 64 characters). Nullable The resource string containing the name of the snapshot to display on the user interface.

Database Column	Details
SnapshotDate	*Type:* datetime. Nullable Start date of the currency snapshot.
SnapshotReference CurrencyID	*Type:* integer. Nullable Reference currency used for this snapshot. Foreign key to the Currency table.
IsStandardRateSnapshot	*Type:* boolean. Key Set to True if this is the default standard rate snapshot, which is created for each FNMP installation.
UpdatedUser	*Type:* text (max 256 characters). Nullable Operator who last modified this record.
UpdatedDate	*Type:* datetime. Nullable Date this record was last modified.

CustomPropertyDisplayXML Table

CustomPropertyDisplayXML stores XML snippets with layout information for custom properties. The XML snippets in this table will be inserted into the default XML layout for the appropriate property dialog. Storing snippets in this table, rather than manually updating the default XML layout, ensures that custom properties will continue to be applied even after upgrading the product (since during a product upgrade, we typically overwrite all property display XML layout with the new defaults for that version of the product).

Table 182: Database columns for CustomPropertyDisplayXML table

Database Column	Details
CustomPropertyDisplayXMLID	*Type:* integer. Key. Generated ID A unique identifier for this XML snippet.
XMLSnippet	*Type:* text An XML snippet that describes how to show this property in the properties dialog.
InsertXPath	*Type:* text XPath which selects an XML node where the snippet will be inserted.
XMLInsertTypeID	*Type:* integer How to insert this property at the selected XPath node. Foreign key to the XMLInsertType table.

Database Column	Details
`InsertOrder`	*Type:* integer The order in which to insert the XML snippet for this property into the XML layout file. If this value is higher than another, it will be inserted after it. Useful when the XML snippet for this property is to be inserted inside another - for instance, if a property creates a tab or group.

DisplayXML Table

The static `DisplayXML` table stores the default XML code representing the property dialog layout for non-type-specific objects such as purchase orders, vendors and evidence. The XML files for type-specific entities (such as assets) are stored in the static type tables (such as `AssetType`) for those objects.

Table 183: Database columns for `DisplayXML` table

Database Column	Details
`XMLType`	*Type:* text (max 30 characters). Key A unique identifier for the type of object associated with the XML. Possible values are: • Contract (not in use any longer - the contract XML files are now stored in `ContractType`) • Vendor • VendorContact • PurchaseOrder • PurchaseOrderDetail • SoftwareTitle • FileEvidence • InstallerEvidence • User • TermAndCondition • Operator • LicensePointsRuleSet.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of entity, stored in XML format.

Document Table

The `Document` table stores details of documents or files relating to assets, contracts, purchase orders, licenses and terms and conditions.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 184: Database columns for `Document` table

Database Column	Details
`DocumentID`	*Type:* integer. Key. Generated ID A unique identifier for the document.
`DocumentTypeID`	*Type:* integer The way that the document is stored and referenced. Foreign key to the `DocumentType` table.
`DocumentName`	*Type:* text (max 500 characters) The name of the document.
`DocumentFile`	*Type:* image. Nullable The binary data for the document (if it is stored in the FlexNet Manager Suite database).
`OpenWith`	*Type:* text (max 500 characters). Nullable The program to attempt to open the document with.
`DocumentDescription`	*Type:* text (max 3000 characters) A description of the document.
`PhysicalLocation`	*Type:* text (max 500 characters). Nullable Physical location of a (possibly hard) copy of this document. NOTE: for compatibility with the FlexNet Manager Suite console, when the document type is 3 (`Reference`), the `DocumentName` column should be used instead, and this field set to null.
`DocumentSize`	*Type:* integer. Nullable Document size in bytes.
`ContentType`	*Type:* text (max 256 characters). Nullable The MIME-type of the document file.
`AssetID`	*Type:* integer. Key. Nullable The asset to which this document may be linked. Foreign key to the `Asset` table.

Database Column	Details
PurchaseOrderID	*Type:* integer. Key. Nullable The purchase order to which the document may be linked. Foreign key to the PurchaseOrder table.
PurchaseOrderDetailID	*Type:* integer. Key. Nullable The purchase order detail (or PO line) to which the document may be linked. Foreign key to the PurchaseOrderDetail table.
ContractID	*Type:* integer. Key. Nullable The contract to which the document may be linked. Foreign key to the Contract table.
SoftwareLicenseID	*Type:* integer. Nullable The license to which the document may be linked. Foreign key to the SoftwareLicense table.
ComplianceUserID	*Type:* integer. Key. Nullable The end-user to which the document may be linked. Foreign key to the ComplianceUser table.
AttachDate	*Type:* datetime The date and time this document was linked.
UserName	*Type:* text (max 256 characters) Operator who created the link between this document and the other object.
DocumentNoteID	*Type:* integer. Key. Nullable The note to which this document may be linked. Foreign key to the DocumentNote table.
ContractNoteID	*Type:* integer. Key. Nullable The contract note to which this document may be linked. Foreign key to the ContractNote table.
TermAndConditionID	*Type:* integer. Key. Nullable The term/condition to which this document may be linked. Foreign key to the TermAndCondition table.
SecurityTypeID	*Type:* integer. Key. Nullable Security type for this document (role-based or individual access). Foreign key to the SecurityType table.
FileType	*Type:* text (max 20 characters). Nullable The type of the file that has been uploaded, if any. This is used to provide full-text indexing.

Database Column	Details
ComplianceOperatorID	*Type:* integer. Nullable The end-operator to which the documnet may be linked. Foreign key to the ComplianceOperator table.
IsOracleLMS	*Type:* boolean If set to True, this field indicates that this document is relating to Oracle LMS. If False, then the document is not relating to Oracle LMS.

DocumentHistory Table

The DocumentHistory table stores history of documents or files relating to assets, contracts, purchase orders, licenses, and terms and conditions.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 185: Database columns for DocumentHistory table

Database Column	Details
DocumentHistoryID	*Type:* integer. Key. Generated ID A unique identifier for the document history.
DocumentID	*Type:* integer. Key The corresponding document. Foreign key to the Document table.
DocumentTypeID	*Type:* integer The way that the document is stored and referenced. Foreign key to the DocumentType table.
DocumentName	*Type:* text (max 500 characters) The name of the document.
DocumentFile	*Type:* image. Nullable The binary data for the document (if it is stored in the FlexNet Manager Suite database).
OpenWith	*Type:* text (max 500 characters). Nullable The program to attempt to open the document with.
DocumentDescription	*Type:* text (max 3000 characters) A description of the document.

Database Column	Details
PhysicalLocation	*Type:* text (max 500 characters). Nullable Physical location of a (possibly hard) copy of this document. NOTE: for compatibility with the FlexNet Manager Suite console, when the document type is 3 (Reference), the DocumentName column should be used instead, and this field set to null.
DocumentSize	*Type:* integer. Nullable Document size in bytes.
ContentType	*Type:* text (max 256 characters). Nullable The MIME-type of the document file.
UserName	*Type:* text (max 256 characters) Operator who created the link between this document and the other object.
AttachDate	*Type:* datetime The date and time this document was linked to the other object.
FileType	*Type:* text (max 20 characters). Nullable The type of the file that has been uploaded, if any. This is used to provide full-text indexing.

DocumentNote Table

DocumentNote stores a list of notes attached to a document. The document itself is attached to a contract.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 186: Database columns for DocumentNote table

Database Column	Details
DocumentNoteID	*Type:* integer. Key. Generated ID A unique identifier for the document note.
DocumentID	*Type:* integer The document that the note is for. Foreign key to the Document table.
ShortDescription	*Type:* text (max 100 characters) In the user interface, this maps to the document reference to which the note relates.

Database Column	Details
LongDescription	*Type:* text. Nullable The content of the note.
CreationUser	*Type:* text (max 128 characters) The operator who created the note.
CreationDate	*Type:* datetime The date of creation of the note.
UpdatedUser	*Type:* text (max 128 characters) The operator who last updated the note.
UpdatedDate	*Type:* datetime The date of the last update to the note.

DocumentType Table

DocumentType is a static value listing the alternative ways that a document can be saved in the database.

Table 187: Database columns for DocumentType table

Database Column	Details
DocumentTypeID	*Type:* integer. Key. Generated ID A unique identifier for each DocumentType. Possible values and the corresponding default strings are: • 1 = Document upload • 2 = File location • 3 = Physical location • 4 = URL.
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a document type. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

Event Table

The Event table stores errors and events processed by the beacon, devices, rules etc.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 188: Database columns for `Event` table

Database Column	Details
`EventID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`ActivityID`	*Type:* integer. Key Foreign key to the `Activity` table.
`EventUID`	*Type:* unique identifier. Key UID to uniquely identify the event.
`EventTypeID`	*Type:* integer. Key Foreign key to the `EventType` table.
`CreationDate`	*Type:* datetime Date and time (UTC) when the Event was created.
`SessionUID`	*Type:* unique identifier. Key. Nullable UID to uniquely identify the the session.

EventLogCategory Table

The `EventLogCategory` table holds the different categories of events created by the system.

Table 189: Database columns for `EventLogCategory` table

Database Column	Details
`EventLogCategoryID`	*Type:* integer. Key. Generated ID A unique identifier for each `EventLogCategory`. Possible values and the corresponding default strings are: • 1 = Email Notification.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a event category. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) A description of the event category.

EventLogDetail Table

The `EventLogDetail` table holds details of the events created by the system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 190: Database columns for `EventLogDetail` table

Database Column	Details
`EventLogDetailID`	*Type:* integer. Key. Generated ID A unique identifier for an event detail.
`EventLogSummaryID`	*Type:* integer. Key The unique identifier for an event. Foreign key to the `EventLogSummary` table.
`EventLogLevelID`	*Type:* integer. Key The level of event. Foreign key to the `EventLogLevel` table.
`MessageTime`	*Type:* datetime. Key The time that the event was raised.
`Message`	*Type:* text (max 256 characters) The brief event message.
`Details`	*Type:* text. Nullable The full event message.
`ParentEventLogDetailID`	*Type:* integer. Key. Nullable The parent event log detail. Foreign key to another event log detail in this same `EventLogDetail` table.

EventLogLevel Table

The `EventLogLevel` table holds the different levels of events created by the system.

Table 191: Database columns for `EventLogLevel` table

Database Column	Details
`EventLogLevelID`	*Type:* integer. Key. Generated ID A unique identifier for each `EventLogLevel`. Possible values and the corresponding default strings are: • 1 = Information • 2 = Warning • 3 = Error. • 4 = Performance.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an event level. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) A description of the event level.

EventLogStatus Table

The `EventLogStatus` table holds the different statuses of events created by the system.

Table 192: Database columns for `EventLogStatus` table

Database Column	Details
`EventLogStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `EventLogStatus`. Possible values and the corresponding default strings are: • 1 = In Progress • 2 = Success • 3 = Failed.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an event status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) A description of the event status.

EventLogSummary Table

The `EventLogSummary` table holds the top level summary of events created by the system.

Table 193: Database columns for `EventLogSummary` table

Database Column	Details
`EventLogSummaryID`	*Type:* integer. Key. Generated ID A unique identifier for an event.
`StartTime`	*Type:* datetime. Key The time that the event started.
`EndTime`	*Type:* datetime. Key. Nullable The time that the event finished.
`EventLogCategoryID`	*Type:* integer. Key The category of event. Foregin key to the `EventLogCategory` table.
`EventName`	*Type:* text (max 128 characters) Brief description of the event.
`EventLogStatusID`	*Type:* integer. Key The status of the event. Foreign key to the `EventLogStatus` table.

EventParameter Table

The `EventParameter` table stores the links between Activities and EventParameterTypes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 194: Database columns for `EventParameter` table

Database Column	Details
`EventParameterID`	*Type:* integer. Key. Generated ID Primary key for the `EventParameter` table
`EventID`	*Type:* integer. Key A link to the `Event` table

Database Column	Details
`EventParameterTypeID`	*Type:* integer A link the the `EventParameterType` table. this value specifies which kind of object the EventParameter is linked to.
`Value`	*Type:* text stores the value of this parameter.

EventParameterType Table

The `EventParameterType` table stores details about the different types of Event Parameters.

Table 195: Database columns for `EventParameterType` table

Database Column	Details
`EventParameterTypeID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`EventParameterTypeName`	*Type:* text (max 256 characters). Key A short piece of text representing the Event Parameter. Internal use only- not to be displayed to the operator.
`IsResourceString`	*Type:* boolean A short piece of text representing the Event Parameter. Internal use only- not to be displayed to the operator.

EventSeverity Table

`EventSeverity` is a static table listing all of the severity levels that an event type can have.

Table 196: Database columns for `EventSeverity` table

Database Column	Details
`EventSeverityID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the EventSeverity record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

EventTarget Table

The `EventTarget` table stores the links between Activities and other tables in the database.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 197: Database columns for `EventTarget` table

Database Column	Details
`EventID`	*Type:* integer. Key Link to the `Event` table
`TargetTypeID`	*Type:* integer. Key A link to the `TargetType` table. this value specifies which kind of object the Event is linked to.
`TargetUID`	*Type:* unique identifier. Key. Nullable used to attach the `Event` to its target. The target table depends on the TargetTypeID of the linked EventType.
`TargetID`	*Type:* integer. Nullable ID of the target. Referenced if the UID is not available.
`TargetName`	*Type:* text (max 128 characters). Nullable TargetName used to record the name of the target. Can be used when the UID or ID is not available.

EventType Table

The `EventType` table stores details about the different types of Events.

Table 198: Database columns for `EventType` table

Database Column	Details
`EventTypeID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`EventTypeName`	*Type:* text (max 256 characters). Key Short text representing the Event Type. Internal use only- not to be displayed to the operator.

Database Column	Details
`EventSeverityID`	*Type:* integer The severity of the Event. 1 = information, 2 = warning, 3 = error, 4 = critical.
`EventMessageResource`	*Type:* text (max 256 characters) A resource name used to look up a description for this Event
`EventTypeStatusID`	*Type:* integer. Key Foreign key to the `EventTypeStatus` table
`ActivityTypeID`	*Type:* integer. Key Foreign key to the `ActivityType` table

EventTypeStatus Table

The `EventTypeStatus` table stores progress stages for different processes.

Table 199: Database columns for `EventTypeStatus` table

Database Column	Details
`EventTypeStatusID`	*Type:* integer. Key. Generated ID Auto-generated status ID
`EventTypeStatus ResourceName`	*Type:* text (max 255 characters). Key Status name resource name
`EventTypeStatusDefault Value`	*Type:* text (max 255 characters). Nullable Default value for status

ILMTPVUCounts Table

This table allows the summarised PVU sub capacity numbers to be imported from ImportedILMTPVUCounts.”.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 200: Database columns for `ILMTPVUCounts` table

Database Column	Details
`ILMTPVUCountsTableID`	*Type:* integer. Key. Generated ID The ID of the ILMTPVUCounts Table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key ID from the `ComplianceComputer` table.
`TitleName`	*Type:* text (max 512 characters). Key The name of the title these points apply to.
`Publisher`	*Type:* text (max 254 characters). Key The name of the publisher of the title these points apply to.
`SubCapacityCores`	*Type:* integer The number of sub-capacity licensable cores for the license on the computer.
`FullCapacityCores`	*Type:* integer The number of full-capacity licensable cores for the license on the computer.
`SubCapacityPVU`	*Type:* integer The number of sub-capacity PVU counts consumed for the license on the computer.
`FullCapacityPVU`	*Type:* integer The number of full-capacity PVU counts consumed for the license on the computer.
`PeakSubCapacityPVU`	*Type:* integer The peak number of sub-capacity PVU counts consumed for the license on the computer.
`PeakFullCapacityPVU`	*Type:* integer The peak number of full-capacity PVU counts consumed for the license on the computer.

ImportResolverErrorResult Table

The `ImportResolverErrorResult` table stores all resolver error message

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 201: Database columns for `ImportResolverErrorResult` table

Database Column	Details
`ImportResolverError ResultID`	*Type:* integer. Key. Generated ID Auto-generated ID for `ImportResolverErrorResult` table

Database Column	Details
`FileName`	*Type:* text (max 255 characters) Name of the file
`DateCreated`	*Type:* datetime Date time where file was resolved.
`ErrorMessage`	*Type:* text. Nullable error message
`ImportResolverTypeID`	*Type:* integer. Key Foreign key to the `ImportResolverType` table

ImportResolverType Table

The `ImportResolverType` table stores all the resolver types.

Table 202: Database columns for `ImportResolverType` table

Database Column	Details
`ImportResolverTypeID`	*Type:* integer. Key. Generated ID Auto-generated ID for `ImportResolverType` table
`ImportResolverTypeName`	*Type:* text (max 255 characters). Key Name of the resolver
`ImportResolverType Resource`	*Type:* text (max 256 characters) A resource name used to look up a description for this resolver type

InstalledSoftwareAttribute Table

`InstalledSoftwareAttribute` stores the attribute values for each installation of an application. Reserved for future expansion.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 203: Database columns for `InstalledSoftwareAttribute` table

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key The installation whose attribute value is being stored. Foreign key to the `InstalledSoftware` table.
`AttributeID`	*Type:* integer. Key The attribute whose value is being stored. Foreign key to the `Attribute` table.
`Value`	*Type:* text (max 400 characters) The value of this attribute of the installed application.

Instance Table

`Instance` stores information about database instances.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 204: Database columns for `Instance` table

Database Column	Details
`InstanceID`	*Type:* integer. Key. Generated ID A unique identifier for an instance.
`ParentInstanceID`	*Type:* integer. Key. Nullable The parent of the instance. Foreign key to another instance in the `Instance` table.
`EnterpriseManager InstanceID`	*Type:* integer. Key. Nullable The Oracle Enterprise Manager database that manages this Oracle instance. Foreign key to the `Instance` table.
`InstalledSoftwareID`	*Type:* integer. Key. Nullable The installation associated with the instance. Foreign key to the `InstalledSoftware` table.
`InstanceTypeID`	*Type:* integer The type of this database instance. Foreign key to the `InstanceType` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The host server running this database instance. Foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key The instance's application. Foreign key to the `SoftwareTitle` table
`InstanceName`	*Type:* text (max 256 characters). Key. Nullable The name of the database instance.
`SerialNo`	*Type:* text (max 256 characters). Nullable The serial number of the database instance.
`InstallationPath`	*Type:* text (max 512 characters). Nullable The installation path of the database instance.
`BusinessApplicationName`	*Type:* text (max 512 characters). Nullable The business application that uses the database instance.
`IsLicensable`	*Type:* boolean Set this to `False` if this instance does not require a license. The default is `True`, which means a license is required.
`IsLicensableForLicenseRec`	*Type:* boolean Set this to `True` if this instance should be included in license reconciliation. `False` means that this instance will not be accounted for in license reconciliation.
`NeverDelete`	*Type:* boolean When a computer does not return any inventory for a specified period of time, it may be deleted. Set this field to `True` to ensure that the instance record does not get deleted when there is no inventory.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable The software license covering this instance. Foreign key to the `SoftwareLicense` table.
`UsedInInventory`	*Type:* boolean If the inventory importer detects that this database instance instance is used, it will set thie field to `True`.
`UsedOverride`	*Type:* boolean. Nullable An operator may manually specify whether this database instance is to be considered used (set this field to `True`), or not (set this field to `False`). This overrides the importer result (`UsedInInventory`) described above.

Database Column	Details
`InventorySourceTypeID`	*Type:* integer Whether this instance has ever been reported in inventory, or has been manually created and maintained. Foreign key to the `ComplianceComputerInventorySourceType` table.
`AuditEvidence`	*Type:* binary. Nullable Oracle LMS CVS files in zip archive.
`AuditEvidenceDate`	*Type:* datetime. Nullable Date and time the Oracle LMS audit evidence was collected by Flexera Inventory Manager
`CreationUser`	*Type:* text (max 256 characters) The operator who created the database instance record.
`CreationDate`	*Type:* datetime The date and time when this instance record was created.
`UpdatedUser`	*Type:* text (max 256 characters). Nullable The operator who most recently updated the database instance record.
`UpdatedDate`	*Type:* datetime The date and time when this instance record was last updated.

InstanceAttribute Table

`InstanceAttribute` stores the attribute values for each installed database instance.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 205: Database columns for `InstanceAttribute` table

Database Column	Details
`InstanceID`	*Type:* integer. Key The database instance whose attribute value is being stored. Foreign key to the `Instance` table.
`AttributeID`	*Type:* integer. Key The attribute whose value is being stored. Foreign key to the `Attribute` table.
`Value`	*Type:* text (max 400 characters) The value of this attribute of the database instance.

InstanceEnvironment Table

`InstanceEnvironment` is a static table listing the possible environments in which database instances may be deployed. For some vendors, the environment affects the costs of licensing the database instance.

Table 206: Database columns for `InstanceEnvironment` table

Database Column	Details
`InstanceEnvironmentID`	*Type:* integer. Key. Generated ID A unique identifier for an `InstanceEnvironment`. Possible values and the corresponding default names are: • 1 = Development • 2 = Test • 3 = Staging • 4 = Production • 5 = Other.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an instance environment. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the environment resource string has no translation.

InstancePropertyValue Table

For each instance, `InstancePropertyValue` stores the values for the custom properties defined in `InstanceTypeProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 207: Database columns for `InstancePropertyValue` table

Database Column	Details
`InstancePropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`InstanceID`	*Type:* integer. Key The instance associated with this property. Foreign key to the `Instance` table.

Database Column	Details
`InstanceTypePropertyID`	*Type:* integer. Key The property whose value is being stored. The type of the instance should match the type that the property is associated with. Foreign key to the `InstanceTypeProperty` table.
`PropertyValue`	*Type:* text (max 4000 characters) The value of the property.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date and time when the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date and time when the record was last updated.

InstanceRole Table

`InstanceRole` is a static tbale listing the possible roles of database instances. For some vendors, the role of the database instance affects the costs of licensing.

Table 208: Database columns for `InstanceRole` table

Database Column	Details
`InstanceRoleID`	*Type:* integer. Key. Generated ID A unique identifier for an `InstanceRole`. Possible values and the corresponding default names are: • 1 = None • 2 = Backup • 3 = Failover • 4 = Mirroring • 5 = Standby • 6 = Other • 7 = Primary.

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an instance role. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the role resource string has no translation.

InstanceTenancy Table

The `InstanceTenancy` table contains the Cloud Service providers tenancy for the instance.

Table 209: Database columns for `InstanceTenancy` table

Database Column	Details
`InstanceTenancyID`	*Type:* integer. Key. Generated ID A unique identifier for each `InstanceTenancy`. Possible values and the corresponding default strings are: • 1 = Default (the instance is running in a VPC) • 2 = Dedicated (the instance is running on a single-tenant hardware). • 3 = Host (the instance is running in a single host affinity mode).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a virtual machine source type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the source type resource string has no translation.
`ImporterString`	*Type:* text (max 100 characters) This is the string which is coming from the data source.

InstanceType Table

`InstanceType` is a static table listing the possible types of database instance.

Table 210: Database columns for `InstanceType` table

Database Column	Details
`InstanceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for an `InstanceType`. Possible values and the corresponding default names are: • 1 = General (for non-Oracle applications) • 2 = Oracle • 3 = Application (for instances created for non-Oracle applications manually flagged as Oracle). • 4 = Oracle EBS Server • 5 = Oracle EBS Module
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an instance type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the instance type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of instance, stored in XML format.

InstanceTypeProperty Table

`InstanceTypeProperty` defines extra custom properties for instances of the specified type.

Table 211: Database columns for `InstanceTypeProperty` table

Database Column	Details
`InstanceTypePropertyID`	*Type:* integer. Key. Generated ID A unique identifier for each property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.
`InstanceTypeID`	*Type:* integer. Key Foreign key to the `InstanceType` table.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

InstanceUser Table

`InstanceUser` links end-users in `LicenseUser` with a particular instance of a database for license counting purposes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 212: Database columns for `InstanceUser` table

Database Column	Details
`InstanceID`	*Type:* integer. Key The instance used by the end-user. Foreign key to a database instance in the `Instance` table.
`LicenseUserID`	*Type:* integer. Key The end-user using the instance. Foreign key to the account name in the `LicenseUser` table.
`Quantity`	*Type:* integer The number of actual end-users of the database instance logging in to the Oracle database through this account. For example, if there is one “Shop Floor” account for all fork lift drivers, this field stores the number of individual drivers that must be accounted for.
`AccountStatus`	*Type:* text (max 256 characters). Nullable The current status of the end-user account.
`CreationDate`	*Type:* datetime. Nullable Date and time when the end-user was created.
`LastLogonDate`	*Type:* datetime. Nullable Date and time when the end-user last logged on.
`DefaultTablespace`	*Type:* text (max 256 characters). Nullable The default tablespace for an Oracle user.
`TempTablespace`	*Type:* text (max 256 characters). Nullable The temporary tablespace for an Oracle user.
`IsManualUser`	*Type:* boolean Whether or not the user was created manually (or through Oracle).

IntervalType Table

`IntervalType` stroes the types of interval used by schedules and by terms and conditions.

Table 213: Database columns for `IntervalType` table

Database Column	Details
`IntervalTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `IntervalType`. Possible values and the corresponding default strings are: • 1 = Day • 2 = Week • 3 = Month.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an interval type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

LicenseUser Table

The `LicenseUser` table lists account names (for end-users and other resources) that have been extracted from other products (such as Oracle databases). These external accounts cannot be reconciled with the end-users listed in the `ComplianceUser` table. Nevertheless, these accounts can be very important for licensing costs.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 214: Database columns for `LicenseUser` table

Database Column	Details
`LicenseUserID`	*Type:* integer. Key. Generated ID A unique identifier for an external end-user.
`LicenseUserLogin`	*Type:* text (max 400 characters). Key The user login extracted from the original listing (for example, from an Oracle database).

Database Column	Details
`Description`	*Type:* text (max 400 characters) The description is usually a group name.
`EmployeeNumber`	*Type:* text (max 256 characters). Nullable The employee number of the external end-user.
`FirstName`	*Type:* text (max 256 characters). Nullable The first name of the end-user extracted from the original listing.
`LastName`	*Type:* text (max 256 characters). Nullable The last name of the end-user extracted from the original listing.
`Email`	*Type:* text (max 400 characters). Nullable The email of the end-user extracted from the original listing.
`SAPClientCode`	*Type:* text (max 2 characters). Nullable The end-user's SAP client code, where applicable.
`SAPInstallationNumber`	*Type:* text (max 10 characters). Nullable The end-user's SAP installation number, where applicable.
`CostCenter`	*Type:* text (max 128 characters). Nullable The SAP cost center that the end-user belongs to
`LicenseUserTypeID`	*Type:* integer The type of external end-user. Foreign key to the `LicenseUserType` table.

LicenseUserConnection Table

`ComplianceUserConnection` stores a link between external end-users in `LicenseUser` which have been reported in inventory, and external IDs which can be used to identify them in their inventory sources. End-users reported in multiple inventory sources will appear multiple times in this table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 215: Database columns for `LicenseUserConnection` table

Database Column	Details
`LicenseUserID`	*Type:* integer. Key A unique identifier for the external end-user. Foreign key to the `LicenseUser` table.

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The inventory source where the end-user was reported. Foreign key to the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key A (hopefully unique) identifier for the end-user in the external inventory source.

LicenseUserExcluded Table

Similarly to the `LicenseUser` table, `LicenseUserExcluded` lists account names extracted from other products (such as Oracle databases); but these accounts are to be excluded from license counts. The accounts are listed in full here since it is possible that they do not already appear in the `LicenseUser` table. Any that do appear in both tables, matched on the login names, are excluded from license counts.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 216: Database columns for `LicenseUserExcluded` table

Database Column	Details
`LicenseUserExcludedID`	*Type:* integer. Key. Generated ID A unique identifier for an excluded end-user.
`LicenseUserLogin`	*Type:* text (max 400 characters). Key The user login extracted from the original listing (for example, from an Oracle database). For the account to be excluded from license counts, this must exactly match a `LicenseUserLogin` from the `LicenseUser` table.
`DefaultQuantity`	*Type:* integer The number of actual users of the database instance logging in through this account. For example, a “SYSTEM” account may allow for a number of administrators to log in. In this table, the default quantity is zero. If this field is non-zero and the end-user matches a `LicenseUser` record, then in some cases, we may exclude this number of end-users from license counting, but include any further accounts covered by the `LicenseUser` record.

LicenseUserType Table

`LicenseUserType` is a static table listing possible types of external end-users (in the `LicenseUser` table).

Table 217: Database columns for `LicenseUserType` table

Database Column	Details
`LicenseUserTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseUserType`. Possible values and the corresponding default strings are: • 1 = Default • 2 = Developer.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an external end-user type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

LogFile Table

The `LogFile` table stores all the log file

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 218: Database columns for `LogFile` table

Database Column	Details
`LogFileID`	*Type:* integer. Key. Generated ID Primary key of the table `LogFile`
`SessionUID`	*Type:* unique identifier. Key Identified of the file
`TaskStepID`	*Type:* integer. Key. Nullable Foreign key to the `TaskStep` table
`FileContent`	*Type:* image holds the log file content
`FileExtension`	*Type:* text (max 10 characters) Extension of the file

MSEAARLSoftwareTitleEdition Table

`MSEAARLSoftwareTitleEdition` contains a list of available product editions for a Microsoft Enterprise Agreement.

Table 219: Database columns for `MSEAARLSoftwareTitleEdition` table

Database Column	Details
`SoftwareRecognitionID`	*Type:* text (max 32 characters). Key The factory unique ID (an MD5 digest) for the product edition in the Application Recognition Library.
`IsPlatform`	*Type:* boolean Whether this edition should be covered by the platform license.

MSSelectLevel Table

`MSSelectLevel` is a static table listing all Microsoft Select price levels.

Table 220: Database columns for `MSSelectLevel` table

Database Column	Details
`MSSelectLevelID`	*Type:* integer. Key. Generated ID A unique identifier for each `MSSelectLevel`. Possible values and the corresponding default strings are: • 1 = A • 2 = B • 3 = C • 4 = D
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a price level. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 16 characters) The text to display if the price level resource string has no translation.
`NumberOfPoints`	*Type:* integer The umber of points that must be purchased to achieve the price level.

MSSelectPool Table

`MSSelectPool` is a static table listing all Microsoft Select pools.

Table 221: Database columns for `MSSelectPool` table

Database Column	Details
`MSSelectPoolID`	*Type:* integer. Key. Generated ID A unique identifier for each `MSSelectPool`. Possible values and the corresponding default strings are: • 1 = Applications • 2 = Systems • 3 = Servers
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a pool. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 64 characters) The text to display if the pool resource string has no translation.

MobileDevice Table

`MobileDevice` extends the `ComplianceComputer` table to store mobile device related property values.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 222: Database columns for `MobileDevice` table

Database Column	Details
`MobileDeviceID`	*Type:* integer. Key. Generated ID A unique identifier for a `MobileDevice` Column use to specify clustered index.
`ComplianceComputerID`	*Type:* integer. Key A unique identifier for a `MobileDevice`. Foreign key to the `ComplianceComputer` table.
`IMEI`	*Type:* text (max 256 characters). Nullable IMEI value of the mobile device.

Database Column	Details
PhoneNo	*Type:* text (max 128 characters). Nullable Phone number of the mobile device.
EmailAddress	*Type:* text (max 256 characters). Nullable The stmp email account associated to a mobile device when the device is connected to ActiveSync.

NotificationItem Table

NotificationItem lists notifications that were sent to end-users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 223: Database columns for NotificationItem table

Database Column	Details
NotificationItemID	*Type:* integer. Key. Generated ID A unique identifier for this notification.
NotificationTypeID	*Type:* integer. Key The type of notification to be sent. Foreign key to the NotificationType table.
NotificationDate	*Type:* datetime. Key The date the notification should be sent.
TaskID	*Type:* integer. Key. Nullable The task the notification is for, if any. Foreign key to the TermAndConditionTask table.
ContractID	*Type:* integer. Key. Nullable The contract the notification is for, if any. Foreign key to the Contract table.
ComplianceUserID	*Type:* integer. Key The end-user that is receiving the notification. Foreign key to the ComplianceUser table.
SentDate	*Type:* datetime. Key. Nullable The date the notification was actually sent.

NotificationTemplate Table

`NotificationTemplate` stores a list of email templates used to generate notification emails.

Table 224: Database columns for `NotificationTemplate` table

Database Column	Details
`NotificationTemplateID`	*Type:* integer. Key. Generated ID A unique identifier for each `NotificationTemplate`. The default templates provided are: • -1 = Contract expiry notification template • -2 = Contract renewal notification template • -3 = Task due notification template • -4 = Task reminder notification template • -5 = Task escalation notification template.
`FileName`	*Type:* text (max 255 characters). Key The template's file name.
`Content`	*Type:* text. Nullable The template content.

NotificationType Table

`NotificationType` stores a list of notification types that can be sent to end-users.

Table 225: Database columns for `NotificationType` table

Database Column	Details
`NotificationTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `NotificationType`. Possible values and the corresponding default strings are: • 1 = Contract Expiry (a notification sent to end-users responsible for a contract when it is due to expire) • 2 = Contract Renewal (a notification sent to end-users responsible for a contract when it is due for renewal) • 3 = Task Due (a notification sent to the end-user assigned to a task when it is due for completion) • 4 = Task Reminder (a notification sent to the end-user assigned to a task as a reminder that the task is nearing completion) • 5 = Task Escalation (a notification sent to the end-user assigned to receive escalations, typically when a task is not completed on time).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a notification type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`NotificationTemplateID`	*Type:* integer. Key. Nullable The template to use when sending notifications of this type. Foreign key to the `NotificationTemplate` table.

OperatorManageState Table

The `OperatorManageState` table lists the possible states for managing who has responsibility for maintaining certain business data. This is for internal use.

Table 226: Database columns for `OperatorManageState` table

Database Column	Details
`OperatorManageStateID`	*Type:* integer. Key. Generated ID A unique identifier for the operator management state of business data.
`Name`	*Type:* text (max 64 characters). Key A unique name for the state

Database Column	Details
DescriptionResourceName	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing the description of the state. Foreign key to the ComplianceResourceString table.
DescriptionDefaultValue	*Type:* text (max 256 characters) A string representing the default name of the state. Foreign key to the ComplianceResourceString table.
IsLocked	*Type:* boolean Is the data locked from edits by an operator.
IsModified	*Type:* boolean Is the data modified by an operator.
IsFactory	*Type:* boolean Is the data from the Reference ARL factory.
AutoUpdate	*Type:* boolean Is the data to be updated automatically.
Priority	*Type:* integer Is the data locked from edits by an operator.

OperatorTaskTypeSetting Table

The OperatorTaskTypeSetting table stores data related to background task type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 227: Database columns for OperatorTaskTypeSetting table

Database Column	Details
OperatorTaskTypeSettingID	*Type:* integer. Key. Generated ID Auto-generated operator task type setting ID
ComplianceOperatorID	*Type:* integer. Key Foreign key to the ComplianceOperator table
ActivityTypeID	*Type:* integer. Key Foreign key to the ActivityType table

Database Column	Details
`Enabled`	*Type:* boolean Enabled flag for a setting

OracleInstance Table

`OracleInstance` stores key characteristics specific to instances of Oracle databases which may impact the cost of licensing.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 228: Database columns for `OracleInstance` table

Database Column	Details
`InstanceID`	*Type:* integer. Key The database instance whose attributes are being stored. Foreign key to the `Instance` table.
`InstanceEnvironmentID`	*Type:* integer The environment of the database instance. Foreign key to the `InstanceEnvironment` table.
`InstanceRoleID`	*Type:* integer The role of the database instance. Foreign key to the `InstanceRole` table.

PaymentSchedule Table

`PaymentSchedule` contains details of the payment schedules managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 229: Database columns for `PaymentSchedule` table

Database Column	Details
`PaymentScheduleID`	*Type:* integer. Key. Generated ID A unique identifier for a payment schedule.

Database Column	Details
`ContractID`	*Type:* integer. Key Identifies a contract to which this payment schedule applies. Foreign key to the `Contract` table.
`PaymentScheduleTypeID`	*Type:* integer Identifies the type of this payment schedule. Foreign key to the `PaymentScheduleType` table.
`PaymentScheduleTermID`	*Type:* integer Identifies the term of payment for this payment schedule. Foreign key to the `PaymentScheduleTerm` table.
`PaymentScheduleCategoryID`	*Type:* integer Identifies the category of this payment schedule. Foreign key to the `PaymentScheduleCategory` table.
`Description`	*Type:* text (max 100 characters) Name of this payment schedule.
`StartDate`	*Type:* datetime The date on which this payment schedule starts.
`EndDate`	*Type:* datetime. Nullable The date on which this payment schedule ends.
`PeriodTypeID`	*Type:* integer Identifies the period type of this payment schedule. Foreign key to the `PeriodType` table.
`IncludeNewAssetsAnd Licenses`	*Type:* boolean If this field is set to `True`, then when a new asset or license is linked to the contract associated with this payment schedule, the item will also be linked to this payment schedule. If `False`, new items linked to the related contract are not automatically linked to the payment schedule (although a manual link can still be made).
`LeaseTerminationDate`	*Type:* datetime. Nullable The termination date of this payment schedule's lease. Only applicable if the payment schedule type is `Lease`.
`LeaseTerminationReason`	*Type:* text (max 100 characters). Nullable The reason this payment schedule's lease was terminated. Only applicable if the payment schedule type is `Lease`.

Database Column	Details
`LeaseNumber`	*Type:* text (max 150 characters). Nullable The number of this payment schedule's lease. Only applicable if the payment schedule type is `Lease`.
`BuyoutCost`	*Type:* currency. Nullable The buyout cost for this payment schedule's lease. Only applicable if the payment schedule type is `Lease`.
`BuyoutCostRateID`	*Type:* integer. Nullable Identifies the currency rate to be applied to this payment schedule's lease buyout cost. Only applicable if the payment schedule type is `Lease`. Foreign key to the `CurrencyRate` table.
`PreviousPurchases`	*Type:* integer. Nullable In the case of a Microsoft Enterprise Agreement renewal, the number of desktops covered by the associated platform license at the end of the previous Microsoft EA.
`Comment`	*Type:* text. Nullable Operator's comments about this payment schedule.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the payment schedule was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator to make the last change to this record.
`UpdatedDate`	*Type:* datetime The date the last change was made to this payment schedule record.

PaymentScheduleCategory Table

`PaymentScheduleCategory` is a static table listing categories that can be assigned to a payment schedule.

Table 230: Database columns for `PaymentScheduleCategory` table

Database Column	Details
`PaymentScheduleCategoryID`	*Type:* integer. Key. Generated ID A unique identifier for each `PaymentScheduleCategory`. Possible values and the corresponding default strings are: • 1 = Fixed • 2 = License true up • 3 = Per hardware item • 4 = Per license quantity.
`ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a payment schedule category. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the category resource string has no translation.

PaymentScheduleDetail Table

`PaymentScheduleDetail` lists all individual periods of a payment schedule.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 231: Database columns for `PaymentScheduleDetail` table

Database Column	Details
`PaymentScheduleDetailID`	*Type:* integer. Key. Generated ID Uniquely identifies this payment schedule period.
`PaymentScheduleID`	*Type:* integer. Key Identifies the payment schedule to which this period applies. Foreign key to the `PaymentSchedule` table.
`PeriodCovered`	*Type:* text (max 50 characters) A string describing the period to which this payment schedule period is applicable. This is a calculated field.
`PeriodStartDate`	*Type:* datetime. Key The date on which this payment schedule period starts.

Database Column	Details
`PeriodEndDate`	*Type:* datetime The date on which this payment schedule period ends.
`DueDate`	*Type:* datetime. Key. Nullable The date on which this payment is due.
`PaymentScheduleDetail PaymentStatusID`	*Type:* integer. Key Identifies the state type of this payment schedule.The default value 2 corresponds to an `Incomplete` status. Foreign key to the `PaymentScheduleDetailPaymentStatus` table.
`PaymentDate`	*Type:* datetime. Nullable Records the date the payment was made.
`ActualAmount`	*Type:* currency. Nullable The actual amount paid in this payment schedule period.
`ActualAmountRateID`	*Type:* integer. Nullable Identifies the currency rate to be applied to the amount paid in this payment schedule period. Foreign key to the `CurrencyRate` table.
`EstimatedAmount`	*Type:* currency. Nullable The estimated amount for this payment schedule period.
`EstimatedAmountRateID`	*Type:* integer. Nullable Identifies the currency rate to be applied to the estimated amount for this payment schedule period. Foreign key to the `CurrencyRate` table.
`BudgetedAmount`	*Type:* currency. Nullable The budgeted amount for this payment schedule period.
`BudgetedAmountRateID`	*Type:* integer. Nullable Identifies the currency rate to be applied to the budgeted amount for this payment schedule period. Foreign key to the `CurrencyRate` table.
`Obligated`	*Type:* boolean If this field is set to `True`, the payee is obligated to pay during this payment schedule period. If this bit is `False` (the default), payment can presumably be deferred.
`Quantity`	*Type:* integer. Nullable The quantity for this payment schedule period.
`UnitPrice`	*Type:* currency. Nullable The unit price for this payment schedule period.

Database Column	Details
UnitPriceRateID	*Type:* integer. Nullable Identifies the currency rate to be applied to the unit price for this payment schedule period. Foreign key to the CurrencyRate table.
SoftwareAssuranceUnit Price	*Type:* currency. Nullable The unit price for support (Software Assurance) for this payment schedule period.
SoftwareAssuranceUnit PriceRateID	*Type:* integer. Nullable Identifies the currency rate to be applied to the unit price for support in this payment schedule period. Foreign key to the CurrencyRate table.
Notes	*Type:* text. Nullable The notes field.
PeriodCoveredResourceName	*Type:* text (max 256 characters). Nullable The resource name used to describe the period to which this payment schedule period is applicable.
PeriodCoveredResource Parameters	*Type:* text (max 256 characters). Nullable The parameters used by the resource name used to describe the period to which this payment schedule period is applicable.

PaymentScheduleDetailPaymentStatus Table

PaymentScheduleDetailPaymentStatus is a static table listing the possible status values for payment schedules.

Table 232: Database columns for PaymentScheduleDetailPaymentStatus table

Database Column	Details
PaymentScheduleDetail PaymentStatusID	*Type:* integer. Key. Generated ID A unique identifier for each PaymentScheduleDetailPaymentStatus. Possible values and the corresponding default strings are: • 1 = Complete • 2 = Incomplete • 3 = Not going to pay.
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a payment schedule status. Foreign key to the ComplianceResourceString table.

Database Column	Details
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the status resource string has no translation.

PaymentScheduleTerm Table

`PaymentScheduleTerm` is a static table listing possible payment schedule terms (the timing of payments in relation to each payment period).

Table 233: Database columns for `PaymentScheduleTerm` table

Database Column	Details
`PaymentScheduleTermID`	*Type:* integer. Key. Generated ID A unique identifier for each `PaymentScheduleTerm`. Possible values and the corresponding default strings are: • 1 = Pre-paid • 2 = At the end of each period • 3 = At the beginning of each period.
`ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a payment schedule term. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the term resource string has no translation.

PaymentScheduleType Table

`PaymentScheduleType` is a static table listing possible payment schedule types.

Table 234: Database columns for `PaymentScheduleType` table

Database Column	Details
`PaymentScheduleTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `PaymentScheduleType`. Possible values and the corresponding default strings are: • 1 = General • 2 = Lease • 3 = Hardware maintenance and support • 4 = Software license • 5 = Software maintenance and support • 6 = Consulting services • 7 = Insurance • 8 = Rent • 9 = Subscription • 10 = EA professional platform • 11 = EA other application.
`ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a payment schedule type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of payment schedule, stored in XML format.

Project Table

Details about each `Project`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 235: Database columns for `Project` table

Database Column	Details
`ProjectID`	*Type:* integer. Key. Generated ID A unique identifier for the project.
`ProjectName`	*Type:* text (max 100 characters). Key The name of the project.
`Comments`	*Type:* text. Nullable Comments recorded about the project.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

PurchaseOrder Table

The `PurchaseOrder` table contains a list of all the purchase orders in the system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 236: Database columns for `PurchaseOrder` table

Database Column	Details
`PurchaseOrderID`	*Type:* integer. Key. Generated ID A unique identifier for the purchase order.
`PurchaseOrderNo`	*Type:* text (max 50 characters). Key The purchase order number.
`ShortDescription`	*Type:* text (max 250 characters). Nullable A short description of the purchase order.

Database Column	Details
PurchaseOrderDate	*Type:* datetime The date recorded for the purchase order.
PurchaseOrderStatusID	*Type:* integer. Nullable The current state of the purchase order. Foreign key to the PurchaseOrderStatus table. The default value of 1 links to a “New” status.
PurchaseOrderTypeID	*Type:* integer. Nullable The type of the purchase order. Foreign key to the PurchaseOrderType table.
InvoiceNo	*Type:* text (max 50 characters). Nullable The invoice number that relates to the purchase order.
InvoiceDate	*Type:* datetime. Nullable The date on the invoice that relates to the purchase order.
TotalPrice	*Type:* currency. Nullable The total price of the purchase order.
TotalPriceRateID	*Type:* integer. Nullable The currency rate to be applied to this purchase order. Foreign key to the CurrencyRate table.
ShippingAndHandling	*Type:* currency. Nullable The amount of money spent on shipping and handling.
ShippingAndHandlingRateID	*Type:* integer. Nullable The currency rate to be applied to the shipping and handling costs related to this purchase order. Foreign key to the CurrencyRate table.
SalesTax	*Type:* currency. Nullable The amount of sales tax paid as part of this purchase order.
SalesTaxRateID	*Type:* integer. Nullable The currency rate to be applied to the sales tax related to this purchase order. Foreign key to the CurrencyRate table.
AutoCalculateCostFrom Children	*Type:* boolean The default value of True indicates that the total price, shipping, and sales tax values should be calculated from the purchase order lines that are children of this purchase order. A value of False means that these values are manually inserted into this purchase order header.
ShippingMethodID	*Type:* integer. Nullable The type shipping used to deliver the product. Foreign key to the ShippingMethod table.

Database Column	Details
`ShippingLocationID`	*Type:* text (max 128 characters). Key. Nullable The location to which the ordered material is shipped. Foreign key to the `GroupEx` table.
`ShippingDate`	*Type:* datetime. Nullable The date the ordered material was shipped.
`RequestNo`	*Type:* text (max 60 characters). Nullable The request number for the purchase order.
`RequestDate`	*Type:* datetime. Nullable The date the purchase order was requested.
`RequestedByID`	*Type:* integer. Key. Nullable The person who requested the purchase order. Foreign key to the `ComplianceUser` table.
`AuthorizedByID`	*Type:* integer. Key. Nullable The person who authorized the purchase order. Foreign key to the `ComplianceUser` table.
`ProcessedByID`	*Type:* integer. Key. Nullable The person who processed the purchase order. Foreign key to the `ComplianceUser` table.
`Comments`	*Type:* text. Nullable Comments recorded about the purchase order.
`VendorID`	*Type:* integer. Key. Nullable The vendor fulfilling this purchase order. Foreign key to the `Vendor` table.
`ContractID`	*Type:* integer. Key. Nullable Foreign key to the `Contract` table, identifying any existing contract related to this purchase order.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this purchase order. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any corporate unit in the enterprise associated with this purchase order. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any cost center in the enterprise associated with this purchase order. Foreign key to the `GroupEx` table.

Database Column	Details
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise category associated with this purchase order. Foreign key to the `GroupEx` table.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

PurchaseOrderDetail Table

The `PurchaseOrderDetail` table contains a list of all the individual purchase order lines in the system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 237: Database columns for `PurchaseOrderDetail` table

Database Column	Details
`PurchaseOrderDetailID`	*Type:* integer. Key. Generated ID A unique identifier for the purchase order line.
`PurchaseOrderID`	*Type:* integer. Key The parent purchase order to which this line belongs. Foreign key to the `PurchaseOrder` table.
`PurchaseOrderDetail ParentID`	*Type:* integer. Nullable When a purchase order line is nested as a child of another, this link identifies the parent. Foreign key to another purchase order line in this `PurchaseOrderDetail` table.
`ItemDescription`	*Type:* text (max 250 characters) A description of the item ordered in this PO line.
`SequenceNumber`	*Type:* integer. Key The sequence number of the PO line in the overall purchase order.

Database Column	Details
`PartNo`	*Type:* text (max 100 characters). Nullable Deprecated, use LicensePartNo.
`Quantity`	*Type:* integer. Nullable The quantity of items purchased in this PO line.
`QuantityPerUnit`	*Type:* integer. Nullable Where the purchase order refers to software licenses, this is the quantity of license included in per unit of this purchase order.
`EffectiveQuantity`	*Type:* integer. Nullable The license entitlemets brought in by this purchase. If the total for this column would exceed the maximum allowable for int, then the total will be reduced to this number.
`LicenseQuantity`	*Type:* integer. Nullable Where the purchase order refers to software licenses, this is the number of license entitlements conferred by the item ordered in this line. This is distinct from the purchase quantity on the line item. For example, it would be possible to order "Qty 50 of XYZ license 10-pack", which would mean a `Quantity` field of 50 and a `LicenseQuantity` of 500.
`LicensePartNo`	*Type:* text (max 100 characters). Key. Nullable The part number or SKU of the item ordered in this PO line.
`UnitPrice`	*Type:* currency. Nullable The unit price of items ordered on this PO line.
`UnitPriceRateID`	*Type:* integer. Nullable The currency rate to be applied to the above unit price. Foreign key to the `CurrencyRate` table.
`SalesTax`	*Type:* currency. Nullable The amount of sales tax paid on this PO line item. May be left null if sales tax is only entered on the purchase order header.
`SalesTaxRateID`	*Type:* integer. Nullable The currency rate to be applied to the above sales tax. Foreign key to the `CurrencyRate` table.
`TotalPrice`	*Type:* currency. Nullable The total price of items in this PO line.
`TotalPriceRateID`	*Type:* integer. Nullable The currency rate to be applied to the above total price. Foreign key to the `CurrencyRate` table.

Database Column	Details
AutoCalculateTotal	*Type:* boolean Set this field to True (the default) for the total price to be caclulated automatically as (UnitPrice * Quantity) + ShippingAndHandling + SalesTax. If False, the operator must enter the total manually.
ShippingAndHandling	*Type:* currency. Nullable The amount of money spent on shipping and handling.
ShippingAndHandlingRateID	*Type:* integer. Nullable The currency rate to be applied to the above shipping and handling costs. Foreign key to the CurrencyRate table.
InheritPOContractID	*Type:* boolean. Key A bit which, if set to 1 (the default), means that the following contract ID is inherited from the parent purchase order.
ContractID	*Type:* integer. Key. Nullable A link to a contract related to this PO line. Foreign key to the Contract table.
InheritPOShippingDetails	*Type:* boolean Set this field to True (the default) for the following shipping details to be inherited from the parent purchase order. If False, an operator has to complete the following details manually.
ShippingDate	*Type:* datetime. Nullable The date the product was shipped.
ShippingMethodID	*Type:* integer. Nullable The delivery method used to deliver the item ordered in this PO line. Foreign key to the ShippingMethod table.
ShippingLocationID	*Type:* text (max 128 characters). Key. Nullable The location to which the item is shipped. Foreign key to the GroupEx table.
MaintenanceOrService Agreement	*Type:* boolean Set this field to True when this PO line includes maintenance or another type of service agreement. If False (the default), there is no maintenance or ofther service agreement associated with this PO line.
EffectiveDate	*Type:* datetime. Nullable The effective date for the Purchase Order Line.
ExpiryDate	*Type:* datetime. Nullable The expiry date for the Purchase Order Line.

Database Column	Details
`InheritPOEnterpriseGroups`	*Type:* boolean Set this field to `True` (the default) for the following enterprise groups to be inherited from the parent purchase order. If `False`, an operator has to complete the following details manually.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this PO line. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any corporate unit within the enterprise associated with this PO line. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise cost center associated with this PO line. Foreign key to the `GroupEx` table.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any category used within the enterprise associated with this PO line. Foreign key to the `GroupEx` table.
`InheritPOProcessDetails`	*Type:* boolean Set this field to `True` (the default) for the following process details to be inherited from the parent purchase order. If `False`, an operator has to complete the following details manually.
`RequestNo`	*Type:* text (max 60 characters). Nullable The request number for the PO line.
`RequestDate`	*Type:* datetime. Nullable The date the related product was requested.
`RequestedByID`	*Type:* integer. Key. Nullable The person who requested the purchase order line. Foreign key to the `ComplianceUser` table.
`AuthorizedByID`	*Type:* integer. Key. Nullable The person who authorized the purchase order line. Foreign key to the `ComplianceUser` table.
`ProcessedByID`	*Type:* integer. Key. Nullable The person who processed the purchase order line. Foreign key to the `ComplianceUser` table.
`Comments`	*Type:* text. Nullable Comments recorded about the purchase order line.

Database Column	Details
`InheritPOInvoiceDetails`	*Type:* boolean Set this field to `True` (the default) for the following invoicing details to be inherited from the parent purchase order. If `False`, an operator has to complete the following details manually.
`InvoiceNo`	*Type:* text (max 50 characters). Nullable The invoice number relating to this PO line.
`InvoiceDate`	*Type:* datetime. Nullable The invoice date for the purchase order line.
`OrderedProduct`	*Type:* text (max 256 characters). Nullable A description of the item ordered in this PO line.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`ExternalID`	*Type:* text (max 32 characters). Nullable A text field where an operator may record the ID of the PO line in any external system it was imported from.
`PurchaseOrderDetailTypeID`	*Type:* integer. Key The type of the PO line. Foreign key to the `PurchaseOrderDetailType` table.
`MSSelectPoolID`	*Type:* integer. Nullable Identifies the Microsoft Select pool. Foreign key to the `MSSelectPool` table.
`MSSelectPoints`	*Type:* decimal. Nullable The number of points consumed by this purchase.
`AutoAcceptRecommendation`	*Type:* boolean Set this field to `True` to automatically accept recommendation calculated for this purchase order line in Link Licenses node.
`SoftwareSkuID`	*Type:* integer. Key. Nullable The SKU that was recognized. This value is optional. Foreign key to the `SoftwareSku` table.

Database Column	Details
PurchaseOrderDetail StatusID	*Type:* integer The current state of the purchase order details. Foreign key to the PurchaseOrderDetailStatus table. The default value of 1 links to a “New” status.
PublisherID	*Type:* integer. Nullable The publisher of this line item. This value is optional. Foreign key to the Vendor table.

PurchaseOrderDetailProperty Table

PurchaseOrderDetailProperty defines extra custom properties for all purchase order lines.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 238: Database columns for PurchaseOrderDetailProperty table

Database Column	Details
PurchaseOrderDetail PropertyID	*Type:* integer. Key. Generated ID Unique identifier for a purchase order line property.
PropertyName	*Type:* text (max 256 characters). Key The name of the custom property. Foreign key to the ComplianceResourceString table.
CustomPropertyDisplayX MLID	*Type:* integer. Nullable Reference to a record in the CustomPropertyDisplayXML table, describing how to show the property on a property dialog.

PurchaseOrderDetailPropertyValue Table

For each purchase order line, PurchaseOrderDetailPropertyValue stores the values for the custom properties defined in PurchaseOrderDetailProperty.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 239: Database columns for `PurchaseOrderDetailPropertyValue` table

Database Column	Details
`PurchaseOrderDetail PropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`PurchaseOrderDetailID`	*Type:* integer. Key The purchase order line associated with the property. Foreign key to the `PurchaseOrderDetail` table
`PurchaseOrderDetail PropertyID`	*Type:* integer. Key the property whose value is being stored. Foreign key to the `PurchaseOrderDetailProperty` table
`PropertyValue`	*Type:* text (max 4000 characters) The property value.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

PurchaseOrderDetailStatus Table

`PurchaseOrderDetailStatus` is a static table listing the possible states for purchase order details, broadly tracking the associated business processes.

Table 240: Database columns for `PurchaseOrderDetailStatus` table

Database Column	Details
`PurchaseOrderDetailStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `PurchaseOrderDetailStatus` Possible values and the corresponding default strings are: • 1 = New • 2 = Pending • 3 = Completed • 4 = Cancelled
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a purchase order status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

PurchaseOrderDetailType Table

`PurchaseOrderDetailType` is a static table listing the possible types of purchase order line item.

Table 241: Database columns for `PurchaseOrderDetailType` table

Database Column	Details
`PurchaseOrderDetailTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `PurchaseOrderDetailType`. Possible values and the corresponding default strings are: • 1 = Not set • 2 = Software • 3 = Hardware • 4 = Service • 5 = Other • 6 = Software upgrade • 7 = Software maintenance • 8 = Disk kit • 9 = Hardware maintenance • 10 = Software Baseline • 11 = Software subscription.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a purchase order line item type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

PurchaseOrderProperty Table

`PurchaseOrderProperty` defines extra custom properties for all purchase orders.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 242: Database columns for `PurchaseOrderProperty` table

Database Column	Details
`PurchaseOrderPropertyID`	*Type:* integer. Key. Generated ID Unique identifier for a purchase order property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

PurchaseOrderPropertyValue Table

For each purchase order, `PurchaseOrderPropertyValue` stores the values for the custom properties defined in `PurchaseOrderProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 243: Database columns for `PurchaseOrderPropertyValue` table

Database Column	Details
`PurchaseOrderPropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`PurchaseOrderID`	*Type:* integer. Key The purchase order associated with this property. Foreign key to the `PurchaseOrder` table.
`PurchaseOrderPropertyID`	*Type:* integer. Key The property whose value is being stored. Foreign key to the `PurchaseOrderProperty` table.
`PropertyValue`	*Type:* text (max 4000 characters) The property value.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.

Database Column	Details
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

PurchaseOrderStatus Table

`PurchaseOrderStatus` is a static table listing the possible states for purchase orders, broadly tracking the associated business processes.

Table 244: Database columns for `PurchaseOrderStatus` table

Database Column	Details
`PurchaseOrderStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `PurchaseOrderStatus`. Possible values and the corresponding default strings are: • 1 = New • 2 = Completed • 3 = Cancelled • 4 = Sent to approver • 5 = Sent to vendor • 6 = Item received.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a purchase order status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

PurchaseOrderType Table

`PurchaseOrderType` is a static table listing the possible types of purchase order. Reserved for future expansion.

Table 245: Database columns for `PurchaseOrderType` table

Database Column	Details
`PurchaseOrderTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `PurchaseOrderType`. Possible values and the corresponding default strings are: • 1 = None.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a purchase order type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

PurchaseProgram Table

`PurchaseProgram` is a static table listing all known contract purchase programs.

Table 246: Database columns for `PurchaseProgram` table

Database Column	Details
`PurchaseProgramID`	*Type:* integer. Key. Generated ID A unique identifier for each `PurchaseProgram`. Possible values and the corresponding default strings are: • 1 = Microsoft Select Agreement • 2 = Microsoft Enterprise Agreement • 3 = Microsoft Open Agreement • 4 = Adobe Cumulative Licensing Program • 5 = Adobe Transactional Licensing Program • 6 = Adobe Site License Program • 7 = Acronis Licensing Program • 8 = Attachmate Volume Purchase Account • 9 = Business Objects Open Licensing Program • 10 = CA Master License Program • 11 = CA Open License Program • 12 = Citrix Easy Licensing Program • 13 = Citrix Enterprise License Program • 14 = Citrix Open Licensing Program • 15 = Citrix Premium Licensing Program • 16 = Corel Contractual License • 17 = Corel Transactional Licensing • 18 = IBM Passport Advantage • 19 = McAfee TSP Licensing Program • 20 = Novell Corporate License Agreement • 21 = Novell Master License Agreement • 22 = Novell Volume License Agreement • 23 = Symantec Elite • 24 = Symantec Express • 25 = Symantec Open Licensing Program • 26 = Symantec Rewards

Database Column	Details
	• 27 = Symantec Volume Licensing Program • 28 = Vmware Purchasing Program • 29 = Macromedia Volume License Program • 30 = Symantec Enterprise Option • 31 = Symantec Enterprise VPA. • 32 = Oracle Master Agreement • 33 = Oracle Unlimited Agreement • 34 = Oracle License and Services Agreement • 35 = Adobe Enterprise Term Licensing Agreement • 36 = Microsoft Products and Services Agreement • 37 = IBM Passport Advantage Express • 38 = IBM Enterprise License Agreement • 39 = IBM Enterprise Software and Services Option
`Name`	*Type:* text (max 100 characters). Key The display name of the purchase program.
`PublisherName`	*Type:* text (max 64 characters). Key The name of publisher under which this purchase program applies.
`Code`	*Type:* text (max 16 characters). Key A short code used to represent this purchase program.

QuerySnapshot Table

`QuerySnapshot` holds the snapshot of data for a report

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 247: Database columns for `QuerySnapshot` table

Database Column	Details
`QuerySnapshotID`	*Type:* integer. Key. Generated ID A unique identifier for a query snapshot.

Database Column	Details
QueryContext	*Type:* text (max 200 characters). Key The query context to partition different queries.
ComplianceSavedSearchID	*Type:* integer. Key. Nullable The query definition this snapshot is for. Foreign key to the ComplianceSavedSearch table.
ComplianceOperatorID	*Type:* integer. Key The operator who ran the report. Foreign key to the ComplianceOperator table.
SnapshotName	*Type:* text (max 200 characters) Name of snapshot.
SnapshotSchema	*Type:* XML Schema of snapshot.
SnapshotDate	*Type:* datetime Date and time of snapshot (UTC)
SnapshotBuildTime	*Type:* big integer Number of milliseconds taken to build the snapshot.
SnapshotRows	*Type:* big integer Number of rows in the snapshot.

RelationType Table

RelationType is a static table containing types of relationship between objects

Table 248: Database columns for RelationType table

Database Column	Details
RelationTypeID	*Type:* integer. Key. Generated ID A unique identifier for each RelationType. Possible values and the corresponding default strings are: • 1 = VMware ESX host managed by vCenter
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a relation type. Foreign key to the ComplianceResourceString table.

Database Column	Details
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the type resource string has no translation.
`ImporterString`	*Type:* text (max 100 characters). Key The text value provided by adapters when importing relation type.

ResponsibilityType Table

`ResponsibilityType` is a static table listing possible end-user responsibilities.

Table 249: Database columns for `ResponsibilityType` table

Database Column	Details
`ResponsibilityTypeID`	*Type:* integer. Key. Generated ID A unique identifier for an end-user's title or responsibility. Possible values and the corresponding default strings are: • 1 = Blank • 2 = Owner • 3 = Signatory • 4 = Contract Manager • 5 = Point of Contact • 6 = Negotiator • 7 = Interested Party.
`ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a user responsibility. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the responsibility resource string has no translation.

RestrictedAccessType Table

`RestrictedAccessType` is a static table holding access types

Table 250: Database columns for `RestrictedAccessType` table

Database Column	Details
`RestrictedAccessTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a type of access. Values are: • 1 = All users • 2 = Accessible only to creator
`RestrictedAccessTypeName`	*Type:* text (max 512 characters). Key Access type name.

RulesEngineRuleDefinition Table

This table stores rule definitions used for consolidating users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 251: Database columns for `RulesEngineRuleDefinition` table

Database Column	Details
`RuleDefinitionID`	*Type:* integer. Key. Generated ID A unique identifier for the rule definition.
`RuleDefinitionName`	*Type:* text (max 128 characters) Name of the rule.
`RuleTypeID`	*Type:* integer Foreign key to the rule type.
`RuleDefinition`	*Type:* text The rule definition XML used to build the rule statement used by the rules engine.
`IsActive`	*Type:* boolean Whether or not this rule is active for execution.
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.

Database Column	Details
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.
`UpdatedDate`	*Type:* datetime The date and time the system landscape was last updated.

RulesEngineRuleType Table

This table stores the available rule types used for rulesengine.

Table 252: Database columns for `RulesEngineRuleType` table

Database Column	Details
`RuleTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the rule type.
`TypeName`	*Type:* text (max 100 characters). Key A unique name for the rule type.
`TitleResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing a rule type. Foreign key to the `ComplianceResourceString` table.
`TitleDefaultValue`	*Type:* text (max 100 characters) The text to display if the rule type resource string has no translation.
`RuleTemplate`	*Type:* text The template used to build a rule for the rules engine.
`DefaultRuleDefinition`	*Type:* text. Nullable Default rule definition for newly created rule

SAMLConfiguration Table

The `SAMLConfiguration` table holds all the saml configurations for the tenants.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 253: Database columns for `SAMLConfiguration` table

Database Column	Details
`SAMLConfigurationID`	*Type:* integer. Key. Generated ID Unique identifier of saml configuration, this is a primary key.
`State`	*Type:* text (max 20 characters) Indicates the state of SAML configuration for the tenant.
`MetadataFileName`	*Type:* text (max 256 characters). Nullable File name of the SAML configuration metadata File.
`MetadataContent`	*Type:* text. Nullable Content from SAML configuration metadata File.
`MetadataURL`	*Type:* text. Nullable URL to download SAML configuration.
`EntityID`	*Type:* text (max 200 characters). Nullable An entity ID is a globally unique name for a SAML entity.
`Created`	*Type:* datetime The date the record was created.
`Updated`	*Type:* datetime. Nullable The date the record was last updated.

SecurityType Table

`SecurityType` lists the types of security model that can be used to determine access to a contract or document.

Table 254: Database columns for `SecurityType` table

Database Column	Details
`SecurityTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SecurityType`. Possible values and the corresponding default strings are: • 1 = Public (security is controlled by the operator's roles) • 2 = Restricted (security is controlled by an access control list of account names).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a security type. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

SerialNumberBlackList Table

`SerialNumberBlackList` stores a blacklist of invalid serial numbers.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 255: Database columns for `SerialNumberBlackList` table

Database Column	Details
`SerialNumberBlackListID`	*Type:* integer. Key. Generated ID The unique identifier for a blacklisted serial number.
`SerialNo`	*Type:* text (max 100 characters). Key The blacklisted serial number.

SessionUIDBeacon Table

The `SessionUIDBeacon` table stores the task's SessionUID and the beacon where the task is running .

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 256: Database columns for `SessionUIDBeacon` table

Database Column	Details
`SessionUID`	*Type:* unique identifier. Key Unique task run identifier
`BeaconID`	*Type:* integer. Key Beacon where the task's session ran

ShippingMethod Table

`ShippingMethod` is a static table listing possible delivery methods. Reserved for future expansion.

Table 257: Database columns for `ShippingMethod` table

Database Column	Details
`ShippingMethodID`	*Type:* integer. Key. Generated ID A unique identifier for each `ShippingMethod`. Possible values and the corresponding default strings are: • 1 = None.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a shipping method. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the shipping method resource string has no translation.

SoftwareLicenseContractPaymentSchedule Table

`SoftwareLicenseContractPaymentSchedule` links a payment schedule to a software license, via a link from that software license to a contract.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 258: Database columns for `SoftwareLicenseContractPaymentSchedule` table

Database Column	Details
`SoftwareLicenseContractID`	*Type:* integer. Key Identifies a link between a software license and a contract. Foreign key to the `SoftwareLicenseContract` table.
`PaymentScheduleID`	*Type:* integer. Key Identifies a payment schedule. Foreign key to the `PaymentSchedule` table.

SystemShutdown Table

A row in this table indicates that the system is being taken down, and is used to show a warning to users.

Table 259: Database columns for `SystemShutdown` table

Database Column	Details
`SystemShutdownID`	*Type:* integer. Key. Generated ID Synthetic key for this table.
`MessageResourceName`	*Type:* text (max 256 characters). Nullable A resource name used to look up a message to show to the operator
`StartTime`	*Type:* datetime The time the shutdown is scheduled to begin
`EndTime`	*Type:* datetime The estimated time that the shutdown will end

TDSComputer Table

`TDSComputer` stores computer properties from Technopedia.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 260: Database columns for `TDSComputer` table

Database Column	Details
`ResourceID`	*Type:* text (max 64 characters). Key Identifier of the computer in TDS.
`ComplianceComputerID`	*Type:* integer. Key The computer associated with this TDSComputer. Foreign key to the `ComplianceComputer` table
`ProductTechnopediaID`	*Type:* text (max 64 characters). Nullable Computer product ID in Technopedia.
`ModelTechnopediaID`	*Type:* text (max 64 characters). Nullable Computer model ID in Technopedia.

TaskExecutionStatus Table

The `TaskExecutionStatus` table stores progress data for rules and background tasks.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 261: Database columns for `TaskExecutionStatus` table

Database Column	Details
`TaskExecutionStatusID`	*Type:* integer. Key. Generated ID Auto-generated task execution status ID
`SessionUID`	*Type:* unique identifier. Key Unique task run identifier
`TaskName`	*Type:* text (max 255 characters). Key The name of task.
`ActivityTypeID`	*Type:* integer. Key Foreign key to the `ActivityType` table
`DateStarted`	*Type:* datetime. Nullable Start date and time for a task.
`DateCompleted`	*Type:* datetime. Nullable Completion date and time for a task.
`EventTypeStatusID`	*Type:* integer. Key Foreign key to the `EventTypeStatus` table
`BeaconRuleID`	*Type:* integer. Key. Nullable Foreign key to the `BeaconRule` table
`ScheduledTriggerDateTick`	*Type:* big integer. Key. Nullable Executed date time in Tick.
`BeaconID`	*Type:* integer. Key. Nullable Beacon where the task is executing.
`BeaconPolicyRevision Number`	*Type:* integer. Nullable Beacon policy revision number
`OperatorLogin`	*Type:* text (max 255 characters). Nullable Login of the operator who started task.

TaskExecutionStatusStep Table

The `TaskExecutionStatusStep` table stores progress data for rule or background task steps.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 262: Database columns for `TaskExecutionStatusStep` table

Database Column	Details
`TaskExecutionStatusStepID`	*Type:* integer. Key. Generated ID Auto-generated task step execution status ID
`TaskExecutionStatusID`	*Type:* integer. Key Foreign key to `TaskExecutionStatus` table.
`TaskStepID`	*Type:* integer. Key The ID of task step.
`BeaconRuleAction PropertyID`	*Type:* integer. Key. Nullable The ID of rule action subtask.
`DateStarted`	*Type:* datetime. Nullable Start date and time for a step.
`DateCompleted`	*Type:* datetime. Nullable Completion date and time for a step.
`EventTypeStatusID`	*Type:* integer. Key Foreign key to the `EventTypeStatus` table
`BeaconUID`	*Type:* unique identifier. Key. Nullable Beacon ID.
`EventTypeID`	*Type:* integer. Key. Nullable Foreign key to the `EventType` table
`EventID`	*Type:* integer. Key. Nullable Foreign key to the `Event` table
`Location`	*Type:* text (max 255 characters). Nullable Server name where operation was performed.
`TaskParameters`	*Type:* XML. Nullable parameters for the task step.

TaskStep Table

The `TaskStep` table stores task steps.

Table 263: Database columns for `TaskStep` table

Database Column	Details
`TaskStepID`	*Type:* integer. Key. Generated ID Auto-generated task step ID
`ActivityTypeID`	*Type:* integer. Key Foreign key to the `ActivityType` table
`TaskStepResourceName`	*Type:* text (max 255 characters). Key Task step name resource name
`TaskStepDefaultValue`	*Type:* text (max 255 characters) Task step name default value
`TaskStepOrder`	*Type:* integer Task step order index

TaskStepEventType Table

The `TaskStepEventType` table stores eventType realted to the taskStep.

Table 264: Database columns for `TaskStepEventType` table

Database Column	Details
`TaskStepID`	*Type:* integer. Key Foreign key to the `TaskStep` table
`EventTypeID`	*Type:* integer. Key Foreign key to the `EventType` table

TermAndCondition Table

`TermAndCondition` stores a list of terms and conditions related to a contract.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 265: Database columns for `TermAndCondition` table

Database Column	Details
`TermAndConditionID`	*Type:* integer. Key. Generated ID A unique identifier for the term/condition.
`TermAndConditionTypeID`	*Type:* integer. Key The type of term/condition. Foreign key to `TermAndConditionType` table.
`Description`	*Type:* text (max 100 characters). Key A description assigned by the operator.
`DocReference`	*Type:* text (max 100 characters). Nullable A text reference to a document for this term/condition.
`Comments`	*Type:* text. Nullable Comments about this term/condition.
`BeginDate`	*Type:* datetime. Nullable The start date for this term or condition.
`EndDate`	*Type:* datetime. Nullable The end date for this term or condition.
`ContractID`	*Type:* integer. Key The contract to which this term/condition applies. Foreign key to the `Contract` table.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the term/condition was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The name of the operator who last updated the term/condition.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`EmailComplianceUserID`	*Type:* integer. Key. Nullable A user who may be emailed according to conditions on this term/condition. Foreign key to the `ComplianceUser` table.
`EmailIntervalTypeID`	*Type:* integer. Key. Nullable The interval type for `EmailInterval`. Foreign key to the `IntervalType` table.

Database Column	Details
`EmailInterval`	*Type:* integer. Nullable The interval used when sending emails.
`ReminderIntervalTypeID`	*Type:* integer. Key. Nullable The interval type for `ReminderInterval`. Foreign key to the `IntervalType` table.
`ReminderInterval`	*Type:* integer. Nullable The interval used when sending reminders.
`EscalationComplianceUserID`	*Type:* integer. Key. Nullable A user who may be emailed if the term/condition needs to be escalated. Foreign key to the `ComplianceUser` table.
`EscalationIntervalTypeID`	*Type:* integer. Key. Nullable The interval type for `EscalationInterval`. Foreign key to the `IntervalType` table.
`EscalationInterval`	*Type:* integer. Nullable The interval used when sending escalation messages.
`Auditable`	*Type:* boolean Boolean to indicate whether the term/condition is auditable.

TermAndConditionTask Table

`TermAndConditionTask` holds extra information about a task.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 266: Database columns for `TermAndConditionTask` table

Database Column	Details
`ComplianceTaskID`	*Type:* integer. Key The task this extra information applies to. Foreign key to the `ComplianceTask` table.
`Completed`	*Type:* boolean Set this field to `True` if this task has been completed.

Database Column	Details
`CompletionDate`	*Type:* datetime. Nullable The date of completion of the task.
`ComplianceUserID`	*Type:* integer. Key. Nullable The end-user this task is assigned to. Foreign key to the `ComplianceUser` table.
`Notes`	*Type:* text. Nullable Notes or comments related to the task.

TermAndConditionType Table

`TermAndConditionType` stores a list of types of different terms/conditions that may be associated with contracts.

Table 267: Database columns for `TermAndConditionType` table

Database Column	Details
`TermAndConditionTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `TermAndConditionType`. The default values and corresponding default strings are: • 1 = Acceptance Period • 2 = Price Change • 3 = Cancellation • 4 = Renewal • 5 = Expiry • 6 = Review • 7 = Limitation.
`TermAndConditionType ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a term/condition type. Foreign key to the `ComplianceResourceString` table.
`TermAndConditionType DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`ManageSoftType`	*Type:* boolean If set to `True`, this field indicates that this term and condition type was created by FlexNet Manager Suite and should not be deleted or edited. If `False`, the type has been created by an operator, and may be modified.

UserNameBlacklist Table

`UserNameBlacklist` stores a list of excluded accounts that will not be imported into FlexNet Manager Suite. If an end-user with account name matching a record in `UserNameBlacklist` already exists in FlexNet Manager Suite, that end-user will not be included in compliance calculations and will not appear in many of the end-user lists.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 268: Database columns for `UserNameBlacklist` table

Database Column	Details
`UserNameBlacklistID`	*Type:* integer. Key. Generated ID A unique identifier for the blacklisted account.
`UserName`	*Type:* text (max 64 characters). Key A blacklisted account name. May contain wildcards (%, _). End-users whose domain\SAM account name match this value will be excluded from compliance calculations.

VMEnabledState Table

`VMEnabledState` is a static table listing the possible operational states of a virtual machine.

Table 269: Database columns for `VMEnabledState` table

Database Column	Details
`VMEnabledStateID`	*Type:* integer. Key. Generated ID A unique identifier for each `VMEnabledState`. Possible values and the corresponding default strings are: • 1 = Started • 2 = Stopped • 3 = Suspended • 4 = Unknown • 5 = Terminated.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a virtual machine operational state. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the operational state resource string has no translation.
`ImporterString`	*Type:* text (max 100 characters). Nullable This is the string which is coming from the data source.

VMHostDatastore Table

`VMHostDatastore` stores host and datastore relationship.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 270: Database columns for `VMHostDatastore` table

Database Column	Details
`HostComplianceComputerID`	*Type:* integer. Key Host computer which has the datastore. Foreign key to the `ComplianceComputer` table.
`Datastore`	*Type:* text (max 64 characters). Key. Nullable The datastore available on the VM host.

VMHostManagedBySoftware Table

`VMHostManagedBySoftware` stores relationships between management software and VM hosts it manages. The RelationTypeID specifies the context of these relationships

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 271: Database columns for `VMHostManagedBySoftware` table

Database Column	Details
`VMHostManagedBySoftwareID`	*Type:* integer. Key. Generated ID The primary key of `VMHostManagedBySoftware`.

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key A unique identifier of an `InstalledSoftware`.
`RelationTypeID`	*Type:* integer. Key The type of relationship between management software and the VM hosts. Foreign key to the `RelationType` table.
`ComplianceComputerID`	*Type:* integer. Key A unique identifier of a `ComplianceComputer`.

VMHostProperty Table

`VMHostProperty` stores properties of VM host.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 272: Database columns for `VMHostProperty` table

Database Column	Details
`HostComplianceComputerID`	*Type:* integer. Key Host computer which has the datastore. Foreign key to the `ComplianceComputer` table.
`VMTypeID`	*Type:* integer. Nullable The VMHost technology type. Foreign key to the `VMType` table.
`HypervisorVersion`	*Type:* text (max 32 characters). Nullable The hypervisor version of the VM host.
`HyperThreadingEnabled`	*Type:* boolean. Nullable Set this to `True` if this VM host has hyper threading enabled.
`PowerState`	*Type:* text (max 32 characters). Nullable The power state of the VM host.
`ManagingSoftwareVersion`	*Type:* text (max 32 characters). Nullable The version of the managing software for the VM host.
`ConnectionState`	*Type:* text (max 32 characters). Nullable The connection state of the VM host to the managing software envrionment.

Database Column	Details
`CloudServiceInstanceTypeID`	*Type:* integer. Nullable Cloud instance type defined by provider. Foreign key to the `CloudServiceInstanceType` table
`CloudServiceRegionID`	*Type:* integer. Nullable Region of the instance. Foreign key to the `CloudServiceRegion` table.
`AvailabilityZoneID`	*Type:* integer. Nullable Location of the instance. Foreign key to the `AvailabilityZone` table
`AllocationTime`	*Type:* datetime. Nullable The time that the Dedicated Host was allocated.
`ReleaseTime`	*Type:* datetime. Nullable The time that the Dedicated Host was released.
`Autoplacement`	*Type:* boolean. Nullable Whether auto-placement is on or off.

VMPool Table

`VMPool` contains information about virtual machine pools (logical groups of VMs or partitions).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 273: Database columns for `VMPool` table

Database Column	Details
`VMPoolID`	*Type:* integer. Key. Generated ID A unique identifier for a virtual machine pool.
`PoolName`	*Type:* text (max 100 characters). Key The name of the pool.
`PoolFriendlyName`	*Type:* text (max 256 characters) The friendly name of the pool.
`Path`	*Type:* text (max 1000 characters) The full path of the pool (including parent pool names).
`VCObjectID`	*Type:* text (max 256 characters). Nullable The ID of the virtual machine folder (pool) in Virtual Center.

Database Column	Details
`NextChild`	*Type:* integer One more than the number of children this pool has.
`PoolPathID`	*Type:* text (max 128 characters) A numerical representation of the path of this pool, constructed from `VMPoolID` values (something like: “1.2.”).
`HostComplianceComputerID`	*Type:* integer. Key. Nullable A link to the host computer that this pool exists on. This is a foreign key to the `ComplianceComputer` table.
`VMPoolTypeID`	*Type:* integer. Key The type of pool. Foreign key to the `VMPoolType` table.
`VirtualMachineID`	*Type:* integer. Nullable If this pool is a virtual machine or partition itself, this is a link to that virtual machine or partition. Foreign key to the `VirtualMachine` table.
`NumberOfProcessors`	*Type:* decimal. Nullable The number of processors in this pool.
`NumberOfLogicalProcessors`	*Type:* integer. Nullable The active number of threads in this pool.
`NumberOfCores`	*Type:* decimal. Nullable The number of cores in this pool.
`MaxNumberOfLogical Processors`	*Type:* integer. Nullable The maximum number of threads assigned for this pool of type processor set.

VMPoolType Table

`VMPoolType` is a static table listing the possible types of a virtual machine pool.

Table 274: Database columns for `VMPoolType` table

Database Column	Details
`VMPoolTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a `VMPoolType`. Possible values and the corresponding default names are: • 1 = Folder • 2 = Data Center • 3 = Compute Resource • 4 = Host System • 5 = Resource Pool • 6 = Virtual Machine • 7 = Physical Shared Pool • 8 = Virtual Shared Pool • 9 = LPAR • 10 = RSET • 11 = Cluster Compute Resource. • 12 = PSET
`VCTypeID`	*Type:* text (max 32 characters) The type of the virtual machine folder in Virtual Center.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a pool type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the pool type resource string has no translation.

VMSourceType Table

`VMSourceType` is a static table used to define possible virtual machine inventory source values (that is, whether the properties were created manually or reported by the compliance importer).

Table 275: Database columns for `VMSourceType` table

Database Column	Details
`VMSourceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `VMSourceType`. Possible values and the corresponding default strings are: • 1 = Manual (the virtual machine properties were manually created and have not been updated by the compliance importer) • 2 = VM Host (the virtual machine's host recently reported inventory and updated these virtual machine properties).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a virtual machine source type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the source type resource string has no translation.

VMState Table

`VMState` is a static table listing the possible relationships between a virtual machine and a physical (inventoried) computer.

Table 276: Database columns for `VMState` table

Database Column	Details
`VMStateID`	*Type:* integer. Key. Generated ID A unique identifier for each `VMState`. Possible values and the corresponding default strings are: • 1 = Linked (the virtual machine is linked to an inventoried or manually created computer) • 2 = Unlinked (the virtual machine is only linked to a "light" computer, automatically created from the host computer's inventory) • 3 = Duplicated (the virtual machine has a duplicate UUID and is not linked to an inventoried or manually created computer).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a virtual machine state. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
DefaultValue	*Type:* text (max 100 characters) The text to display if the state resource string has no translation.

VMType Table

VMType is a static table listing the possible types of virtual machine or partition.

Table 277: Database columns for VMType table

Database Column	Details
VMTypeID	*Type:* integer. Key. Generated ID A unique identifier for a VMType. Possible values and the corresponding default names are: • 1 = VMware • 2 = Hyper-V • 3 = LPAR • 4 = WPAR • 5 = nPar • 6 = vPar • 7 = SRP • 8 = Zone • 9 = Unknown. • 10 = Oracle VM • 11 = AWS EC2 • 12 = Linux KVM
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a virtual machine or partition type. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

Vendor Table

The `Vendor` table contains a list of all the vendors in the system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 278: Database columns for `Vendor` table

Database Column	Details
`VendorID`	*Type:* integer. Key. Generated ID A unique identifier for the vendor.
`VendorName`	*Type:* text (max 64 characters). Key The name of the vendor.
`VendorPreviousName`	*Type:* text (max 64 characters). Nullable Any earlier name that the vendor was previously known as.
`BusinessPhoneNumber`	*Type:* text (max 30 characters). Nullable The business phone number of the vendor.
`FaxPhoneNumber`	*Type:* text (max 30 characters). Nullable The fax number of the vendor.
`Address_Street`	*Type:* text (max 200 characters). Nullable The street address of the vendor.
`Address_City`	*Type:* text (max 200 characters). Nullable The city of the vendor.
`Address_State`	*Type:* text (max 200 characters). Nullable The state or province of the vendor.
`Address_ZIP`	*Type:* text (max 20 characters). Nullable The ZIP or postal code of the vendor.
`Address_Country`	*Type:* text (max 100 characters). Nullable The country of the vendor.
`Address2_Street`	*Type:* text (max 200 characters). Nullable The second street address of the vendor, if applicable.
`Address2_City`	*Type:* text (max 200 characters). Nullable The second city of the vendor.

Database Column	Details
`Address2_State`	*Type:* text (max 200 characters). Nullable The second state or province of the vendor.
`Address2_ZIP`	*Type:* text (max 20 characters). Nullable The second ZIP or postal code of the vendor.
`Address2_Country`	*Type:* text (max 100 characters). Nullable The second country of the vendor.
`WebSite`	*Type:* text (max 200 characters). Nullable The web site of the vendor.
`Email`	*Type:* text (max 200 characters). Nullable The email address of the vendor.
`ParentVendorID`	*Type:* integer. Nullable A link to a vendor's parent vendor. Foreign key to another vendor record in this `Vendor` table. Vendor hierarchies are not currently implemented.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`AutomaticallyAccept Purchases`	*Type:* boolean Whether purchases from this vendor should have their license linking recommendations in the `EntitlementRecommendation` table automatically accepted.

VendorContact Table

`VendorContact` contains a list of all the vendor contacts, or individuals employed by the vendor with whom this enterprise has contact.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 279: Database columns for `VendorContact` table

Database Column	Details
`VendorContactID`	*Type:* integer. Key. Generated ID A unique identifier for the contact.
`UserTitleID`	*Type:* integer. Nullable The title of the contact's name. Foreign key to the `UserTitle` table.
`FirstName`	*Type:* text (max 128 characters) The first name of the contact.
`MiddleName`	*Type:* text (max 128 characters). Nullable The middle name(s) of the contact.
`LastName`	*Type:* text (max 128 characters). Nullable The last name name of the contact.
`UserSuffixID`	*Type:* integer. Nullable The suffix to the name of the contact.
`JobTitle`	*Type:* text (max 128 characters). Nullable The job title of the contact.
`VendorID`	*Type:* integer. Key A link to the contact's parent vendor. Foreign key to the `Vendor` table.
`BusinessPhoneNumber`	*Type:* text (max 30 characters). Nullable The business phone number of the contact.
`MobilePhoneNumber`	*Type:* text (max 30 characters). Nullable The mobile phone number of the contact.
`FaxPhoneNumber`	*Type:* text (max 30 characters). Nullable The fax number of the contact.
`Address_Street`	*Type:* text (max 200 characters). Nullable The street address of the contact.
`Address_City`	*Type:* text (max 200 characters). Nullable The city of the contact.
`Address_State`	*Type:* text (max 200 characters). Nullable The state or province of the contact.
`Address_ZIP`	*Type:* text (max 20 characters). Nullable The ZIP or postal code of the contact.

Database Column	Details
`Address_Country`	*Type:* text (max 100 characters). Nullable The country of the contact.
`Address2_Street`	*Type:* text (max 200 characters). Nullable The second street address of the contact, if applicable.
`Address2_City`	*Type:* text (max 200 characters). Nullable The second city of the contact.
`Address2_State`	*Type:* text (max 200 characters). Nullable The second state or province of the contact.
`Address2_ZIP`	*Type:* text (max 20 characters). Nullable The second ZIP or postal code of the contact.
`Address2_Country`	*Type:* text (max 100 characters). Nullable The second country of the contact.
`Email`	*Type:* text (max 200 characters). Nullable The email address of the contact.
`Messenger`	*Type:* text (max 200 characters). Nullable The instant messenger address of the contact.
`Comments`	*Type:* text. Nullable Comments recorded about the contact.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

VendorProperty Table

`VendorProperty` defines extra custom properties for all vendors.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 280: Database columns for `VendorProperty` table

Database Column	Details
`VendorPropertyID`	*Type:* integer. Key. Generated ID Unique identifier for a vendor property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the custom property. Foreign key to the `ComplianceResourceString` table.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Reference to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

VendorPropertyValue Table

For each vendor, `VendorPropertyValue` stores the values for the custom properties defined in `VendorProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 281: Database columns for `VendorPropertyValue` table

Database Column	Details
`VendorPropertyValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`VendorID`	*Type:* integer. Key The vendor associated with this property. Foreign key to the `Vendor` table.
`VendorPropertyID`	*Type:* integer. Key The property whose value is being stored. Foreign key to the `VendorProperty` table.
`PropertyValue`	*Type:* text (max 4000 characters) The property value.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.

Database Column	Details
UpdatedUser	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
UpdatedDate	*Type:* datetime. Nullable The date the record was last updated.

VirtualMachine Table

VirtualMachine stores extra information for computers identified as virtual machines or hardware partitions.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 282: Database columns for VirtualMachine table

Database Column	Details
VirtualMachineID	*Type:* integer. Key. Generated ID A unique identifier for virtual machine or partition properties associated with a computer.
HostComplianceComputerID	*Type:* integer. Key. Nullable The virtual machines or partition's host computer. Foreign key to the ComplianceComputer table.
ComplianceComputerID	*Type:* integer. Key The computer associated with these virtual machine or partition properties. Computer in the ComplianceComputer table.
VMTypeID	*Type:* integer The type of virtual machine or partition. Foreign key to the VMType table.
UUID	*Type:* text (max 256 characters). Nullable The UUID (Universally Unique Identifier) of the virtual machine. Used to match virtual machine properties to their associated ComplianceComputer.
VMName	*Type:* text (max 256 characters). Nullable The name of the virtual machine or partition.
VMLocation	*Type:* text (max 256 characters). Nullable The location of the virtual machine on the file system.

Database Column	Details
GuestFullName	*Type:* text (max 256 characters). Nullable The configured operating system for the guest.
FriendlyName	*Type:* text (max 256 characters). Nullable The friendly name of the virtual machine or partition.
VCObjectID	*Type:* text (max 256 characters). Nullable The ID of the virtual machine in Virtual Center.
TotalMemory	*Type:* big integer. Nullable The total memory of the virtual machine (in bytes).
VMStateID	*Type:* integer The state of the virtual machine, related to whether it is linked to a computer or not. Foreign key to the VMState table.
VMPoolID	*Type:* integer. Nullable The resource pool that the virtual machine belongs to. Foreign key to the VMPool table.
ZoneResourceManagement MethodTypeID	*Type:* integer. Nullable The resource management method used for this Solaris Zone VM. Foreign key to the ZoneResourceManagementMethodType table.
CPUUsage	*Type:* integer. Nullable The maximum CPU usage of the Virtual Machine (measured in MHz).
MemoryUsage	*Type:* big integer. Nullable The maximum memory usage of the Virrtual Machine (in bytes).
MaxNumberOfLogical Processors	*Type:* decimal. Nullable The maximum number of threads this VM is allowed to access.
VMEnabledStateID	*Type:* integer The operational state of the virtual machine (powered on, off, and so on). Foreign key to the VMEnabledState table.
VMSourceTypeID	*Type:* integer Whether the virtual machine properties are manually entered or created from inventory. Foreign key to the VMSourceType table.
CreationUser	*Type:* text (max 256 characters) The operator who created this record.
CreationDate	*Type:* datetime The date/time when this record was created.

Database Column	Details
`UpdatedUser`	*Type:* text (max 256 characters). Nullable The operator who last updated this record.
`UpdatedDate`	*Type:* datetime The date/time when this record was last updated.
`AffinityEnabled`	*Type:* boolean Set this to `True` if this VM is unable to move to different host computers.
`CPUAffinity`	*Type:* text (max 256 characters). Nullable Contains the CPU Affinity value for virtual machines (Host Logical processors)
`CoreAffinity`	*Type:* text (max 256 characters). Nullable Contains the Core Affinity value for virtual machine
`PartitionID`	*Type:* text (max 100 characters). Nullable Partition ID generated and used by the managing virtualization platform
`PartitionNumber`	*Type:* integer. Nullable Number of this partition
`IsHostAssignedManually`	*Type:* boolean Was the virtual machine assigned to its host manually? This prevents unlinking of the virtual machine.

XMLInsertType Table

`XMLInsertType` is a static table storing how custom property XML snippets will be inserted into the default property display layout XML file.

Table 283: Database columns for `XMLInsertType` table

Database Column	Details
`XMLInsertTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `XMLInsertType`. Possible values are: • 1 = Before (the new snippet needs to go before the existing XML element) • 2 = After (the new snippet needs to go after the existing XML element) • 3 = Replace (the new snippet needs to replace the existing XML element) • 4 = First child (the new snippet needs to be added as the first child of the existing XML element) • 5 = Last child (the new snippet needs to be added as the last child of the existing XML element).
`TypeDescription`	*Type:* text (max 50 characters). Key A description of the insert type.

ZoneResourceManagementMethodType Table

`ZoneResourceManagementMethodType` is a static table listing the possible resource management methods which can be used for Solaris Zones.

Table 284: Database columns for `ZoneResourceManagementMethodType` table

Database Column	Details
`ZoneResourceManagement MethodTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a `ZoneResourceManagementMethodType`. Possible values and the corresponding default names are: • 1 = resource-pool • 2 = capped-cpu • 3 = dedicated-cpu
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a Solaris Zone resource management method. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the resource management method resource string has no translation.

Database Column	Details
`ImporterString`	*Type:* text (max 100 characters) This is the string which is coming from the data source.

Compliance.Logic.Discovery Tables

The complete set of database tables documented here includes:

- ASN1Object table (see ASN1Object Table)
- DeviceRole table (see DeviceRole Table)
- DiscoveredDevice table (see DiscoveredDevice Table)
- DiscoveredDeviceCalculatedMember table (see DiscoveredDeviceCalculatedMember Table)
- DiscoveredDeviceParent table (see DiscoveredDeviceParent Table)
- DiscoveredDeviceSNMPInfo table (see DiscoveredDeviceSNMPInfo Table)
- DiscoveredDeviceVDIBrokerInfo table (see DiscoveredDeviceVDIBrokerInfo Table)
- DiscoveredDeviceVDIInfo table (see DiscoveredDeviceVDIInfo Table)
- DiscoveredDeviceVirtualizationInfo table (see DiscoveredDeviceVirtualizationInfo Table)
- KnownOracleListener table (see KnownOracleListener Table)
- KnownOracleService table (see KnownOracleService Table)
- Site table (see Site Table)
- SiteSubnet table (see SiteSubnet Table)
- VirtualizationProductName table (see VirtualizationProductName Table)

ASN1Object Table

Stores a mapping from an ASN ObjectID (OID) to a type of device.

Table 285: Database columns for `ASN1Object` table

Database Column	Details
`OID`	*Type:* text (max 128 characters). Key ASN object identifier.
`Description`	*Type:* text (max 512 characters) The fully expanded text version of the object identifier.

Database Column	Details
`ObjectRole`	*Type:* integer. Nullable What role does the device perform?

DeviceRole Table

A lookup table of possible roles for network devices.

Table 286: Database columns for `DeviceRole` table

Database Column	Details
`DeviceRoleID`	*Type:* integer. Key. Generated ID The id of the device role.
`Description`	*Type:* text (max 64 characters). Key The name of the device role. Possible id and name pairs are: • 0 = Computer • 1 = Workstation • 2 = Server • 3 = Printer • 4 = Switch • 5 = Router • 6 = Hub • 7 = NetworkDevice • 8 = Vendor.

DiscoveredDevice Table

A `DiscoveredDevice` is a loose record of the discovery of a device on a network, using any of a number of discovery methods. As such, the same device may be found in more than one way (see `DuplicateID` which may be able to track this fact if known), or by more than one distinguishing feature. Accordingly this table has a somewhat unsatisfactory primary key!

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 287: Database columns for `DiscoveredDevice` table

Database Column	Details
`DeviceID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`DeviceUID`	*Type:* unique identifier. Key A unique external identifier for the device.
`DuplicateID`	*Type:* integer. Key. Nullable Reference to another discovery record for this device, if known.
`ComputerID`	*Type:* integer. Key. Nullable FlexNet Manager Suite computer id, if known
`DeviceName`	*Type:* text (max 64 characters). Key. Nullable NetBIOS name for computers or any name for other devices, if known.
`DNSFullName`	*Type:* text (max 256 characters). Key. Nullable Fully qualified DNS name, if known.
`NTDomainName`	*Type:* text (max 256 characters). Key. Nullable NT domain name, if known.
`IPAddress`	*Type:* text (max 64 characters). Key. Nullable IP address of the device.
`IPSubnet`	*Type:* text (max 64 characters). Nullable IP subnet that contains the node.
`IPSubnetMask`	*Type:* text (max 64 characters). Nullable IP subnet mask for the subnet contains the device.
`PhysicalAddress`	*Type:* text (max 64 characters). Key. Nullable Network adapter physical address of the node. Can be a MAC address or token ring address.

Database Column	Details
`DeviceRole`	*Type:* integer. Nullable What role does the device perform? • NULL = unknown • 0 = Computer (don't know if server or workstation) • 1 = Workstation • 2 = Server • 3 = Printer • 4 = Switch • 5 = Router • 6 = Hub
`OperatingSystem`	*Type:* text (max 128 characters). Nullable Operating system of the node, if it is a computer.
`IsManaged`	*Type:* integer. Key. Nullable Is the device to be managed by FlexNet Manager Suite? 0 = no, 1 = yes, NULL = unknown.
`Description`	*Type:* text (max 256 characters). Nullable Operator-entered description of the device.
`SystemDescription`	*Type:* text (max 256 characters). Nullable This field is currently unused.
`SystemLocation`	*Type:* text (max 256 characters). Nullable This field is currently unused.
`SystemContact`	*Type:* text (max 256 characters). Nullable This field is currently unused.
`FirstDiscovered`	*Type:* datetime The date and time that the node was first discovered.
`LastUpdate`	*Type:* datetime The last time the node was checked or updated.
`LastDataSourceName`	*Type:* text (max 128 characters). Key. Nullable A name that identifies where the discovery information came from (for example: physical location, server, and so on).
`LastDataSourceType`	*Type:* text (max 32 characters). Key. Nullable The type of data source (for example: Excel, Fluke, NM, Text).

Database Column	Details
`OpenPortsTCP`	*Type:* text (max 512 characters). Nullable The comma-delimited list of TCP ports which were found to be open on scan.
`OpenPortsUDP`	*Type:* text (max 512 characters). Nullable The comma-delimited list of UDP ports which were found to be open on scan.
`ScannedOperatingSystem`	*Type:* text (max 512 characters). Nullable The IP scan tool's best guess at the operating system. This is based on corner cases in the behavior of the network protocol stack.
`ScannedOsType`	*Type:* text (max 512 characters). Nullable OS Type, as reported by scan tool.
`ScannedOsVendor`	*Type:* text (max 512 characters). Nullable OS Vendor, as reported by scan tool.
`ScannedOsFamily`	*Type:* text (max 512 characters). Nullable OS family, as reported by scan tool.
`ScannedOsGen`	*Type:* text (max 512 characters). Nullable OS Generation (Versions), as reported by scan tool.
`ScannedMacAddress`	*Type:* text (max 64 characters). Nullable MAC Address, as reported by scan tool.
`ScannedMacVendor`	*Type:* text (max 512 characters). Nullable MAC Vendor, as reported by scan tool.
`SQLDiscoveredBy`	*Type:* text (max 128 characters). Nullable The discovery tool used to discover SQL Server.
`SQLPorts`	*Type:* text (max 128 characters). Nullable The ports where SQL Server has been discovered.
`IPAddressInt`	*Type:* big integer. Key. Nullable Integer representation of IPAddress column.

DiscoveredDeviceCalculatedMember Table

Stores summary strings of `DiscoveredDevice` details that are expensive to calculate on demand.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 288: Database columns for `DiscoveredDeviceCalculatedMember` table

Database Column	Details
`DeviceID`	*Type:* integer. Key Device identity number.
`IsOracle`	*Type:* boolean. Nullable Have we discovered Oracle on this machine?
`OracleListeners`	*Type:* text (max 512 characters). Nullable A summary string representing any known Oracle Listeners, and the port they can be contacted on.
`OracleServices`	*Type:* text (max 512 characters). Nullable A summary string representing any known Oracle Services.
`IsSQL`	*Type:* boolean. Nullable Have we discovered SQL Server on this machine?
`IsVDI`	*Type:* boolean. Nullable Is this machine a virtual desktop?
`IsVDIBroker`	*Type:* boolean. Nullable Have we discovered a VDI broker on this machine?

DiscoveredDeviceParent Table

Records any parent-child relationships between DiscoveredDevice records.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 289: Database columns for `DiscoveredDeviceParent` table

Database Column	Details
`DeviceID`	*Type:* integer. Key The child DiscoveredDevice ID
`ParentDeviceID`	*Type:* integer. Key The parent DiscoveredDevice ID

DiscoveredDeviceSNMPInfo Table

Records any SNMP information discovered for a `DiscoveredDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 290: Database columns for `DiscoveredDeviceSNMPInfo` table

Database Column	Details
`DeviceID`	*Type:* integer. Key Device identity number.
`SNMP_snmpCommunityString`	*Type:* text (max 256 characters). Nullable The SNMP Community String that was used for obtaining SNMP MIBs. This and all following SNMP attributes are defined in RFC1907 and others available from the IETF websites.
`SNMP_sysDescr`	*Type:* text (max 256 characters). Nullable A textual description of the device. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software.
`SNMP_sysObjectID`	*Type:* text (max 256 characters). Nullable The vendor's authoritative identification of the network management subsystem contained in the entity. This value is allocated within the SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining 'what kind of device' is being managed. For example, if vendor 'Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its 'Fred Router'.
`SNMP_sysObjectIDSymbolic`	*Type:* text (max 256 characters). Nullable The symbolic representation of the same value as sysObjectID.
`SNMP_sysUpTime`	*Type:* big integer. Nullable The time (in hundredths of a second) since the network management portion of the system was last re-initialized.
`SNMP_sysContact`	*Type:* text (max 256 characters). Nullable The textual identification of the contact person for this managed node, together with information on how to contact this person.
`SNMP_sysName`	*Type:* text (max 256 characters). Nullable An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name.

Database Column	Details
`SNMP_sysLocation`	*Type:* text (max 256 characters). Nullable The physical location of this node (for example, 'telephone closet, 3rd floor').
`SNMP_sysServices`	*Type:* integer. Nullable A bitmask indicating at which of the seven OSI protocol levels the system provides services (physical=1, TCP = 8, applications = 64, etc).
`SNMP_ipForwarding`	*Type:* integer. Nullable Set to 1 if the device forwards IP packets, 2 otherwise.

DiscoveredDeviceVDIBrokerInfo Table

Maps a `DiscoveredDevice` to a VDI site and broker type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 291: Database columns for `DiscoveredDeviceVDIBrokerInfo` table

Database Column	Details
`DeviceID`	*Type:* integer. Key Device identity number. Foreign key to the `DiscoveredDevice` table.
`VDISiteName`	*Type:* text (max 256 characters). Key. Nullable The site to which this VDI broker belongs.
`BrokerType`	*Type:* text (max 256 characters). Key The type of VDI broker found.

DiscoveredDeviceVDIInfo Table

Records any VDI information discovered for a `DiscoveredDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 292: Database columns for `DiscoveredDeviceVDIInfo` table

Database Column	Details
`DeviceID`	*Type:* integer. Key Device identity number.
`VDIGroupName`	*Type:* text (max 256 characters). Nullable The Desktop Group to which this VDI belongs.
`VDITemplateName`	*Type:* text (max 256 characters). Nullable The template from which this VDI device was cloned.
`VDISiteName`	*Type:* text (max 256 characters). Key. Nullable The site to which this VDI belongs.
`BrokerType`	*Type:* text (max 256 characters). Key. Nullable The type of broker that serves up this VDI.
`BrokerMachineName`	*Type:* text (max 64 characters). Nullable NetBIOS name for the VDI broker.
`BrokerDomainName`	*Type:* text (max 256 characters). Nullable NT domain name of the broker.
`BrokerIPAddress`	*Type:* text (max 256 characters). Nullable The IP of the broker.
`IsPersistent`	*Type:* boolean Whether or not the VDI device is a persistent one.

DiscoveredDeviceVirtualizationInfo Table

Records any virtualization server information discovered for a `DiscoveredDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 293: Database columns for `DiscoveredDeviceVirtualizationInfo` table

Database Column	Details
`DeviceID`	*Type:* integer. Key Device identity number.

Database Column	Details
`Protocol`	*Type:* text (max 16 characters). Nullable The protocol by which the virtualization API is accessed on the device.
`Port`	*Type:* integer The TCP port used by the protocol.
`APIType`	*Type:* text (max 32 characters). Nullable The reported API type.
`APIVersion`	*Type:* text (max 16 characters). Nullable The supported version of the API.
`ProductNameID`	*Type:* integer The reported product name.
`ProductVersion`	*Type:* text (max 16 characters). Nullable The reported product version.

KnownOracleListener Table

Records any discovered Oracle listeners that a `DiscoveredDevice` is providing.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 294: Database columns for `KnownOracleListener` table

Database Column	Details
`KnownOracleListenerID`	*Type:* integer. Key. Generated ID Unique id for the known listener.
`DeviceID`	*Type:* integer. Key Device identity number. Foreign key to the `DiscoveredDevice` table.
`Port`	*Type:* integer. Key Port for this listener.
`Name`	*Type:* text (max 128 characters) The name of the service provided by the device.
`Version`	*Type:* text (max 32 characters) The version of the service provided by the device.

Database Column	Details
`ManuallyAdded`	*Type:* boolean Boolean field specifying whether the KnownService record has been manually added by the user.
`DiscoveredRemotely`	*Type:* boolean True means this listener is discovered using remote discovery, false otherwise.
`DiscoveredLocally`	*Type:* boolean True means this listener is discovered using local discovery, false otherwise.
`DiscoveredViaTNSNames`	*Type:* boolean True means this listener is discovered from a TNSNames file on an inventory beacon, false otherwise.

KnownOracleService Table

Records any discovered Oracle services (databases) on a `DiscoveredDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 295: Database columns for `KnownOracleService` table

Database Column	Details
`KnownOracleServiceID`	*Type:* integer. Key. Generated ID Unique id for the known Oracle service.
`KnownOracleListenerID`	*Type:* integer. Key. Nullable Listener identity number.
`DeviceID`	*Type:* integer. Key Network device identity number. Foreign key to the `DiscoveredDevice` table.
`Name`	*Type:* text (max 128 characters). Key The name of the service provided by the device.
`ManuallyAdded`	*Type:* boolean Boolean field specifying whether the KnownService record has been manually added by the user.
`DiscoveredRemotely`	*Type:* boolean True means this service is discovered using remote discovery, false otherwise.

Database Column	Details
`DiscoveredLocally`	*Type:* boolean True means this service is discovered using local discovery, false otherwise.
`DiscoveredViaTNSNames`	*Type:* boolean True means this service is discovered from a TNSNames file on an inventory beacon, false otherwise.

Site Table

The `Site` table contains data about network locations (sites).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 296: Database columns for `Site` table

Database Column	Details
`SiteID`	*Type:* integer. Key. Generated ID The ID for the site.
`Name`	*Type:* text (max 256 characters). Key The name of the site.
`AutoPopulated`	*Type:* boolean Specifies whether the row was populated automatically (1) or manually (0).
`Enabled`	*Type:* boolean Specifies whether the row will be used when mapping domains and devices to sites.

SiteSubnet Table

The `SiteSubnet` table contains data about subnets in a network location (or site).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 297: Database columns for `SiteSubnet` table

Database Column	Details
`SubnetID`	*Type:* integer. Key. Generated ID The ID for the subnet.
`IPSubnet`	*Type:* text (max 64 characters). Key The IP address of the subnet, in human-readable dotted decimal notation (example: 172.16.254.1).
`IPSubnetBits`	*Type:* tiny integer. Key The number of bits in the CIDR routing prefix (in IPv4 terms, the subnet mask), expressed as a decimal number.
`SiteID`	*Type:* integer. Key SiteID of the site in which the Subnet resides. Foreign key to the `Site` table.
`AutoPopulated`	*Type:* boolean Specifies whether the row was populated automatically (1) or manually (0).
`Enabled`	*Type:* boolean Specifies whether the row will be used when mapping domains and devices to sites.
`IPAddressRangeFrom`	*Type:* big integer. Key The first IP address within the subnet, expressed as an integer. This value is automatically calculated when the record is created.
`IPAddressRangeTo`	*Type:* big integer. Key The last IP address within the subnet, expressed as an integer. This value is automatically calculated. (For an IPv4 address with a 32-bit subnet mask, this value is the same as the first IP address within the subnet, since this IP address identifies exactly one device.)

VirtualizationProductName Table

Stores unique virtualization server software names for a `DiscoveredDevice`.

Table 298: Database columns for `VirtualizationProductName` table

Database Column	Details
`VirtualizationProductNameID`	*Type:* integer. Key. Generated ID Device identity number.

Database Column	Details
`ProductName`	*Type:* text (max 256 characters). Key The reported product name.

Compliance.Logic.Licensing Tables

The complete set of database tables documented here includes:

- AccessEvidence table (see AccessEvidence Table)
- AccessEvidenceEx table (see AccessEvidenceEx Table)
- AccessEvidenceMatchCount table (see AccessEvidenceMatchCount Table)
- AccessMode table (see AccessMode Table)
- AccessedSoftware table (see AccessedSoftware Table)
- AccessedSoftwareDeviceAssignmentReasons table (see AccessedSoftwareDeviceAssignmentReasons Table)
- AccessedSoftwareOccurrence table (see AccessedSoftwareOccurrence Table)
- AccessedSoftwareUserAssignmentReasons table (see AccessedSoftwareUserAssignmentReasons Table)
- AccessingDevice table (see AccessingDevice Table)
- AccessingDeviceSnapshot table (see AccessingDeviceSnapshot Table)
- AccessingUser table (see AccessingUser Table)
- AccessingUserSnapshot table (see AccessingUserSnapshot Table)
- AssignmentStatus table (see AssignmentStatus Table)
- AssignmentStatusFailureReasonMapping table (see AssignmentStatusFailureReasonMapping Table)
- ClientAccessSourceType table (see ClientAccessSourceType Table)
- ClientAccessedAccessEvidence table (see ClientAccessedAccessEvidence Table)
- ClientAccessedAccessOccurrence table (see ClientAccessedAccessOccurrence Table)
- Cluster table (see Cluster Table)
- ClusterComputer table (see ClusterComputer Table)
- ClusterHostAffinityRule table (see ClusterHostAffinityRule Table)
- ClusterHostAffinityRuleType table (see ClusterHostAffinityRuleType Table)
- ClusterNodeType table (see ClusterNodeType Table)
- ClusterType table (see ClusterType Table)
- ComplianceComputerSnapshot table (see ComplianceComputerSnapshot Table)

- ComplianceComputerTag table (see ComplianceComputerTag Table)
- ComplianceUserSnapshot table (see ComplianceUserSnapshot Table)
- ComplianceUserTag table (see ComplianceUserTag Table)
- DatabaseMutex table (see DatabaseMutex Table)
- EndOfSupportLife table (see EndOfSupportLife Table)
- EndOfSupportLifeName table (see EndOfSupportLifeName Table)
- EntitlementRecommendation table (see EntitlementRecommendation Table)
- EntitlementRecommendationState table (see EntitlementRecommendationState Table)
- EntitlementTransaction table (see EntitlementTransaction Table)
- EntitlementTransactionOtherCandidate table (see EntitlementTransactionOtherCandidate Table)
- EntitlementTransactionState table (see EntitlementTransactionState Table)
- EntitlementTransactionType table (see EntitlementTransactionType Table)
- EvidenceExistenceRule table (see EvidenceExistenceRule Table)
- EvidenceStatus table (see EvidenceStatus Table)
- FNMEAFeature table (see FNMEAFeature Table)
- FNMEALicensedFeature table (see FNMEALicensedFeature Table)
- FileEvidenceCompany table (see FileEvidenceCompany Table)
- FileEvidenceEx table (see FileEvidenceEx Table)
- FileEvidenceFile table (see FileEvidenceFile Table)
- FileEvidenceLanguage table (see FileEvidenceLanguage Table)
- FileEvidenceMatchCount table (see FileEvidenceMatchCount Table)
- FileEvidencePath table (see FileEvidencePath Table)
- GroupSnapshot table (see GroupSnapshot Table)
- ImporterRun table (see ImporterRun Table)
- ImporterStepValidationIssue table (see ImporterStepValidationIssue Table)
- ImporterStepValidationIssueType table (see ImporterStepValidationIssueType Table)
- InstalledFileEvidence table (see InstalledFileEvidence Table)
- InstalledInstallerAttribute table (see InstalledInstallerAttribute Table)
- InstalledInstallerEvidence table (see InstalledInstallerEvidence Table)
- InstalledInstanceReplacement table (see InstalledInstanceReplacement Table)

- InstalledSoftwareData table (see InstalledSoftwareData Table)
- InstalledSoftwareRemoval table (see InstalledSoftwareRemoval Table)
- InstalledSoftwareReplacement table (see InstalledSoftwareReplacement Table)
- InstalledSoftwareUsageData table (see InstalledSoftwareUsageData Table)
- InstalledWMIEvidence table (see InstalledWMIEvidence Table)
- InstallerEvidence table (see InstallerEvidence Table)
- InstallerEvidenceEx table (see InstallerEvidenceEx Table)
- InstallerEvidenceMatchCount table (see InstallerEvidenceMatchCount Table)
- InstallerEvidenceType table (see InstallerEvidenceType Table)
- InstdSWAssignmentReasons table (see InstdSWAssignmentReasons Table)
- LicenseAssignmentConsumptionReason table (see LicenseAssignmentConsumptionReason Table)
- LicenseAssignmentFailureReason table (see LicenseAssignmentFailureReason Table)
- LicenseBreachReason table (see LicenseBreachReason Table)
- LicenseDefinitionTitle table (see LicenseDefinitionTitle Table)
- LicenseDefinitionType table (see LicenseDefinitionType Table)
- LicenseDefinitionUsageRight table (see LicenseDefinitionUsageRight Table)
- LicenseMeasurement table (see LicenseMeasurement Table)
- LicenseSimulation table (see LicenseSimulation Table)
- LicenseSimulationBreachStatus table (see LicenseSimulationBreachStatus Table)
- LicenseSimulationChangeType table (see LicenseSimulationChangeType Table)
- LicenseSimulationHWDetails table (see LicenseSimulationHWDetails Table)
- LicenseSimulationLicenseDetails table (see LicenseSimulationLicenseDetails Table)
- LicenseSimulationResults table (see LicenseSimulationResults Table)
- LicenseSimulationRowType table (see LicenseSimulationRowType Table)
- LicenseSimulationSWDetails table (see LicenseSimulationSWDetails Table)
- LicenseSimulationScenario table (see LicenseSimulationScenario Table)
- LicenseStatus table (see LicenseStatus Table)
- NewFileEvidence table (see NewFileEvidence Table)
- OracleLegacyLicenseType table (see OracleLegacyLicenseType Table)
- PODetailProcess table (see PODetailProcess Table)

- PVUSoftwareLicenseProcessorData table (see PVUSoftwareLicenseProcessorData Table)
- PVUVirtualMachineLayer table (see PVUVirtualMachineLayer Table)
- PeriodType table (see PeriodType Table)
- ProcessAction table (see ProcessAction Table)
- ProcessState table (see ProcessState Table)
- ReconcileAccdSWDevAsgnReasons table (see ReconcileAccdSWDevAsgnReasons Table)
- ReconcileAccdSWUsrAsgnReasons table (see ReconcileAccdSWUsrAsgnReasons Table)
- ReconcileAccessedSoftwareData table (see ReconcileAccessedSoftwareData Table)
- ReconcileInstalledSoftwareData table (see ReconcileInstalledSoftwareData Table)
- ReconcileInstalledSoftwareUsageData table (see ReconcileInstalledSoftwareUsageData Table)
- ReconcileInstdSWAssignmentReasons table (see ReconcileInstdSWAssignmentReasons Table)
- ReconcileInterestingBundleAccessComputer table (see ReconcileInterestingBundleAccessComputer Table)
- ReconcileInterestingBundleInstallComputer table (see ReconcileInterestingBundleInstallComputer Table)
- ReconcileInterestingLicenses table (see ReconcileInterestingLicenses Table)
- ReconcileInterestingTitles table (see ReconcileInterestingTitles Table)
- ReconcileSoftwareAccessDeviceLicensePointsConsumedData table (see ReconcileSoftwareAccessDeviceLicensePointsConsumedData Table)
- ReconcileSoftwareAccessUserLicensePointsConsumedData table (see ReconcileSoftwareAccessUserLicensePointsConsumedData Table)
- ReconcileSoftwareLicenseComputerProblem table (see ReconcileSoftwareLicenseComputerProblem Table)
- ReconcileSoftwareLicenseCoresConsumedData table (see ReconcileSoftwareLicenseCoresConsumedData Table)
- ReconcileSoftwareLicenseGroupPointsConsumedData table (see ReconcileSoftwareLicenseGroupPointsConsumedData Table)
- ReconcileSoftwareLicenseILMTPointsConsumedData table (see ReconcileSoftwareLicenseILMTPointsConsumedData Table)
- ReconcileSoftwareLicensePointsConsumedData table (see ReconcileSoftwareLicensePointsConsumedData Table)
- ReconcileSoftwareLicensePointsConsumedReason table (see ReconcileSoftwareLicensePointsConsumedReason Table)
- ReconcileSoftwareLicenseProcessorData table (see ReconcileSoftwareLicenseProcessorData Table)
- ReconcileSoftwareLicenseSecondUseMappingData table (see ReconcileSoftwareLicenseSecondUseMappingData Table)
- ReconcileSoftwareUserLicensePointsConsumedData table (see ReconcileSoftwareUserLicensePointsConsumedData Table)

- ReconcileVirtualMachineLayer table (see ReconcileVirtualMachineLayer Table)
- RegistryEvidence table (see RegistryEvidence Table)
- RegistryEvidenceHive table (see RegistryEvidenceHive Table)
- RegistryEvidenceKey table (see RegistryEvidenceKey Table)
- RegistryEvidenceValue table (see RegistryEvidenceValue Table)
- RelatedInstalledInstallerEvidence table (see RelatedInstalledInstallerEvidence Table)
- RelatedInstalledInstallerEvidenceSourceMap table (see RelatedInstalledInstallerEvidenceSourceMap Table)
- RelatedInstalledSoftwareData table (see RelatedInstalledSoftwareData Table)
- SAPSoftwareLicense table (see SAPSoftwareLicense Table)
- SAPSoftwareLicenseType table (see SAPSoftwareLicenseType Table)
- SAPSpecialVersion table (see SAPSpecialVersion Table)
- ServicePack table (see ServicePack Table)
- ServicePackName table (see ServicePackName Table)
- SoftwareAccessDeviceLicensePointsConsumedData table (see SoftwareAccessDeviceLicensePointsConsumedData Table)
- SoftwareAccessMode table (see SoftwareAccessMode Table)
- SoftwareAccessUserLicensePointsConsumedData table (see SoftwareAccessUserLicensePointsConsumedData Table)
- SoftwareLicense table (see SoftwareLicense Table)
- SoftwareLicenseAllocation table (see SoftwareLicenseAllocation Table)
- SoftwareLicenseAllocationStatus table (see SoftwareLicenseAllocationStatus Table)
- SoftwareLicenseAllocationUserType table (see SoftwareLicenseAllocationUserType Table)
- SoftwareLicenseBreachReasonData table (see SoftwareLicenseBreachReasonData Table)
- SoftwareLicenseChangeEvent table (see SoftwareLicenseChangeEvent Table)
- SoftwareLicenseChangeEventReason table (see SoftwareLicenseChangeEventReason Table)
- SoftwareLicenseChangeEventSource table (see SoftwareLicenseChangeEventSource Table)
- SoftwareLicenseCloudServiceProvider table (see SoftwareLicenseCloudServiceProvider Table)
- SoftwareLicenseComplianceStatus table (see SoftwareLicenseComplianceStatus Table)
- SoftwareLicenseComputerProblemData table (see SoftwareLicenseComputerProblemData Table)
- SoftwareLicenseComputerProblemType table (see SoftwareLicenseComputerProblemType Table)
- SoftwareLicenseConnection table (see SoftwareLicenseConnection Table)
- SoftwareLicenseContract table (see SoftwareLicenseContract Table)

- SoftwareLicenseCoresConsumedData table (see SoftwareLicenseCoresConsumedData Table)
- SoftwareLicenseCreation table (see SoftwareLicenseCreation Table)
- SoftwareLicenseDefinition table (see SoftwareLicenseDefinition Table)
- SoftwareLicenseDuration table (see SoftwareLicenseDuration Table)
- SoftwareLicenseExemptionReason table (see SoftwareLicenseExemptionReason Table)
- SoftwareLicenseExemptionRole table (see SoftwareLicenseExemptionRole Table)
- SoftwareLicenseGroupAllocationReportingType table (see SoftwareLicenseGroupAllocationReportingType Table)
- SoftwareLicenseGroupAssignmentHistory table (see SoftwareLicenseGroupAssignmentHistory Table)
- SoftwareLicenseGroupAssignmentHistoryType table (see SoftwareLicenseGroupAssignmentHistoryType Table)
- SoftwareLicenseGroupBreachStatus table (see SoftwareLicenseGroupBreachStatus Table)
- SoftwareLicenseGroupPointsConsumedData table (see SoftwareLicenseGroupPointsConsumedData Table)
- SoftwareLicenseILMTPointsConsumedData table (see SoftwareLicenseILMTPointsConsumedData Table)
- SoftwareLicenseKey table (see SoftwareLicenseKey Table)
- SoftwareLicenseKeyType table (see SoftwareLicenseKeyType Table)
- SoftwareLicenseMetric table (see SoftwareLicenseMetric Table)
- SoftwareLicensePVUPointsConsumedComputersSCD table (see SoftwareLicensePVUPointsConsumedComputersSCD Table)
- SoftwareLicensePVUPointsConsumedPerHostSCD table (see SoftwareLicensePVUPointsConsumedPerHostSCD Table)
- SoftwareLicensePVUPointsConsumedPerRegionSCD table (see SoftwareLicensePVUPointsConsumedPerRegionSCD Table)
- SoftwareLicensePVUPointsPeakConsumedData table (see SoftwareLicensePVUPointsPeakConsumedData Table)
- SoftwareLicensePVURegionPeakCalculatedData table (see SoftwareLicensePVURegionPeakCalculatedData Table)
- SoftwareLicensePartitioningDefault table (see SoftwareLicensePartitioningDefault Table)
- SoftwareLicensePoints table (see SoftwareLicensePoints Table)
- SoftwareLicensePointsConsumedData table (see SoftwareLicensePointsConsumedData Table)
- SoftwareLicensePointsConsumedReasonData table (see SoftwareLicensePointsConsumedReasonData Table)
- SoftwareLicensePointsConsumedReasonType table (see SoftwareLicensePointsConsumedReasonType Table)
- SoftwareLicensePointsDefault table (see SoftwareLicensePointsDefault Table)
- SoftwareLicensePointsRule table (see SoftwareLicensePointsRule Table)
- SoftwareLicensePointsRuleCloudServiceProvider table (see SoftwareLicensePointsRuleCloudServiceProvider Table)
- SoftwareLicensePointsRuleSet table (see SoftwareLicensePointsRuleSet Table)

- SoftwareLicenseProcessorPointsData table (see SoftwareLicenseProcessorPointsData Table)
- SoftwareLicenseProduct table (see SoftwareLicenseProduct Table)
- SoftwareLicensePropertyValue table (see SoftwareLicensePropertyValue Table)
- SoftwareLicenseProposalStatus table (see SoftwareLicenseProposalStatus Table)
- SoftwareLicensePurchaseType table (see SoftwareLicensePurchaseType Table)
- SoftwareLicenseReservation table (see SoftwareLicenseReservation Table)
- SoftwareLicenseReservationNecessityCheckResult table (see SoftwareLicenseReservationNecessityCheckResult Table)
- SoftwareLicenseReservationStatus table (see SoftwareLicenseReservationStatus Table)
- SoftwareLicenseReservationType table (see SoftwareLicenseReservationType Table)
- SoftwareLicenseScopeTag table (see SoftwareLicenseScopeTag Table)
- SoftwareLicenseScopeTagType table (see SoftwareLicenseScopeTagType Table)
- SoftwareLicenseScoping table (see SoftwareLicenseScoping Table)
- SoftwareLicenseSecondUseMappingData table (see SoftwareLicenseSecondUseMappingData Table)
- SoftwareLicenseSnapshot table (see SoftwareLicenseSnapshot Table)
- SoftwareLicenseTierType table (see SoftwareLicenseTierType Table)
- SoftwareLicenseType table (see SoftwareLicenseType Table)
- SoftwareLicenseTypeChangeProposal table (see SoftwareLicenseTypeChangeProposal Table)
- SoftwareLicenseTypeCloudServiceProviderDefault table (see SoftwareLicenseTypeCloudServiceProviderDefault Table)
- SoftwareLicenseTypePriority table (see SoftwareLicenseTypePriority Table)
- SoftwareLicenseTypeProperty table (see SoftwareLicenseTypeProperty Table)
- SoftwareLicenseUseRight table (see SoftwareLicenseUseRight Table)
- SoftwareLicenseUseRightIBM table (see SoftwareLicenseUseRightIBM Table)
- SoftwareLicenseUseRightName table (see SoftwareLicenseUseRightName Table)
- SoftwareLicenseUseRightProposal table (see SoftwareLicenseUseRightProposal Table)
- SoftwareLifeCycle table (see SoftwareLifeCycle Table)
- SoftwareRecognition table (see SoftwareRecognition Table)
- SoftwareSKULookup table (see SoftwareSKULookup Table)
- SoftwareSku table (see SoftwareSku Table)
- SoftwareTitle table (see SoftwareTitle Table)

- SoftwareTitleAccessEvidence table (see SoftwareTitleAccessEvidence Table)
- SoftwareTitleAction table (see SoftwareTitleAction Table)
- SoftwareTitleClassification table (see SoftwareTitleClassification Table)
- SoftwareTitleEOSL table (see SoftwareTitleEOSL Table)
- SoftwareTitleEdition table (see SoftwareTitleEdition Table)
- SoftwareTitleEx table (see SoftwareTitleEx Table)
- SoftwareTitleFileEvidence table (see SoftwareTitleFileEvidence Table)
- SoftwareTitleHierarchy table (see SoftwareTitleHierarchy Table)
- SoftwareTitleHierarchyEx table (see SoftwareTitleHierarchyEx Table)
- SoftwareTitleInstallerEvidence table (see SoftwareTitleInstallerEvidence Table)
- SoftwareTitleLicense table (see SoftwareTitleLicense Table)
- SoftwareTitleLicenseProposal table (see SoftwareTitleLicenseProposal Table)
- SoftwareTitleLicenseProposalAction table (see SoftwareTitleLicenseProposalAction Table)
- SoftwareTitleLicenseReason table (see SoftwareTitleLicenseReason Table)
- SoftwareTitleOracle table (see SoftwareTitleOracle Table)
- SoftwareTitleProduct table (see SoftwareTitleProduct Table)
- SoftwareTitleProperty table (see SoftwareTitleProperty Table)
- SoftwareTitlePropertyValue table (see SoftwareTitlePropertyValue Table)
- SoftwareTitlePublisher table (see SoftwareTitlePublisher Table)
- SoftwareTitleRegistryEvidence table (see SoftwareTitleRegistryEvidence Table)
- SoftwareTitleSuite table (see SoftwareTitleSuite Table)
- SoftwareTitleSuiteEx table (see SoftwareTitleSuiteEx Table)
- SoftwareTitleType table (see SoftwareTitleType Table)
- SoftwareTitleVersion table (see SoftwareTitleVersion Table)
- SoftwareTitleVersionServicePack table (see SoftwareTitleVersionServicePack Table)
- SoftwareTitleWMIEvidence table (see SoftwareTitleWMIEvidence Table)
- SoftwareUserLicensePointsConsumedData table (see SoftwareUserLicensePointsConsumedData Table)
- SoftwareUserLicensePointsConsumedSuggested table (see SoftwareUserLicensePointsConsumedSuggested Table)
- SoftwareUserLicensePointsConsumedSuggestedHistory table (see SoftwareUserLicensePointsConsumedSuggestedHistory Table)
- SoftwareUserLicensePointsHistory table (see SoftwareUserLicensePointsHistory Table)

- Tag table (see Tag Table)
- TargetOperatingSystemType table (see TargetOperatingSystemType Table)
- TemporalInstalledSoftwareSCD table (see TemporalInstalledSoftwareSCD Table)
- VDI table (see VDI Table)
- VDIEndPointAccess table (see VDIEndPointAccess Table)
- VDIGroup table (see VDIGroup Table)
- VDISite table (see VDISite Table)
- VDITemplate table (see VDITemplate Table)
- VDIUser table (see VDIUser Table)
- WMIEvidence table (see WMIEvidence Table)
- WMIEvidenceMatchCount table (see WMIEvidenceMatchCount Table)

AccessEvidence Table

`AccessEvidence` lists software access evidence that is used to identify that a particular item of software (defined in the `SoftwareTitle` table) has been accessed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 299: Database columns for `AccessEvidence` table

Database Column	Details
`AccessEvidenceID`	*Type:* integer. Key. Generated ID A unique identifier for an software access evidence record.
`DisplayName`	*Type:* text (max 256 characters). Key The display name of the software as reported by the software access evidence.
`Version`	*Type:* text (max 72 characters). Key The version of the software as reported by the software access evidence.
`Edition`	*Type:* text (max 50 characters). Key The edition of the software as reported by the software access evidence.
`Publisher`	*Type:* text (max 200 characters). Key The publisher of the software as reported by the installer evidence.

Database Column	Details
OperatorManageStateID	*Type:* integer. Key The management responsibility for this information. Foreign key to the OperatorManageState table.
Ignored	*Type:* boolean Set this field to True if the access evidence is not used for application recognition.
IsShared	*Type:* boolean

AccessEvidenceEx Table

The AccessEvidenceEx table contains additional information on the access evidence managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 300: Database columns for AccessEvidenceEx table

Database Column	Details
AccessEvidenceID	*Type:* integer. Key A unique identifier for an access evidence record.
OperatorManageStateID	*Type:* integer. Nullable The management responsibility for this information. Foreign key to the OperatorManageState table.
Ignored	*Type:* boolean. Nullable Set this field to True if the access evidence is not used for application recognition.

AccessEvidenceMatchCount Table

AccessEvidenceMatchCount tracks the number of times that each access evidence (rule) has been detected as installed and recorded in the data source. A separate count is kept for each access evidence rule, and for each data source.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 301: Database columns for `AccessEvidenceMatchCount` table

Database Column	Details
`AccessEvidenceMatchCountID`	*Type:* integer. Key. Generated ID A synthetic unique identifier is required, since `ComplianceConnectionID`, being nullable, cannot be included in the primary key.
`AccessEvidenceID`	*Type:* integer. Key The access evidence which is being matched. Foreign key to the `AccessEvidence` table.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The data source where the match is occurring. Foreign key to the `ComplianceConnection` table.
`MatchedCount`	*Type:* integer The number of installed access evidence records in this data source matching this access evidence rule.
`InstallCount`	*Type:* integer The number of physical application installations recognized in this data source using this access evidence rule.

AccessMode Table

The `AccessMode` table holds the available states an application can be considered accessed.

Table 302: Database columns for `AccessMode` table

Database Column	Details
`AccessModeID`	*Type:* integer. Key. Generated ID A unique identifier for each `AccessMode`. Possible values and the corresponding default strings are: • 1 = Local • 2 = App-V • 3 = XenApp • 4 = XenDesktop • 5 = VMware View • 6 = Office 365

Database Column	Details
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an access mode. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the access mode resource string has no translation.

AccessedSoftware Table

AccessedSoftware lists all the access records of an application from a device.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 303: Database columns for AccessedSoftware table

Database Column	Details
AccessedSoftwareID	*Type:* big integer. Key. Generated ID A unique identifier for an accessed software record.
AccessingUserID	*Type:* integer. Key. Nullable An identifier for a accessing user record. Foreign key to the AccessingUser table.
AccessingDeviceID	*Type:* integer. Key. Nullable An identifier for a accessing device record. Foreign key to the AccessingDevice table.
SoftwareTitleID	*Type:* integer. Key The software that is being accessed. Foreign key to the SoftwareTitle table.
SoftwareLicenseID	*Type:* integer. Key. Nullable The link to the license this access has been counted against. Foreign key to the SoftwareLicense table.
SoftwareLicense AllocationID	*Type:* integer. Key. Nullable The link to the license allocation this access has consumed. Foreign key to the SoftwareLicenseAllocation table.
ConsumedCount	*Type:* integer. Nullable The number of this installation consumed on the license.

Database Column	Details
IsLicensed	*Type:* boolean Set this field to True when this access is licensed.
LastAccessDate	*Type:* datetime. Nullable Last access date recorded for this software access.
LastInventoryDate	*Type:* datetime. Nullable Last time access inventory was collected for this software access.
PointsCalculated	*Type:* integer The number of calculated points this installation consumes.

AccessedSoftwareDeviceAssignmentReasons Table

AccessedSoftwareDeviceAssignmentReasons lists all license assignments attempted for client access of an application after license reconcile for the device.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 304: Database columns for AccessedSoftwareDeviceAssignmentReasons table

Database Column	Details
AccessingDeviceID	*Type:* integer. Key. Nullable The accessing device under examination. Foreign key to the AccessingDevice table.
ComplianceComputerID	*Type:* integer. Key The compliance computer under examination. Foreign key to the ComplianceComputer table.
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
MultiProductLicensePhase	*Type:* boolean This field is set to True when this installation is licensed during the multi-product license phase of license reconcile.
Order	*Type:* integer The order this license was attempted to be assigned to this installation.

Database Column	Details
LAConsReasonID	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
LicenseAssignment FailureReasonID	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
AddedBySQLPhase	*Type:* boolean Specifies whether the licence allocation was done by SQL or C# code. This is an internal field that can be used when troubleshooting license assignments.

AccessedSoftwareOccurrence Table

AccessedSoftwareOccurrence lists access occurrences for accessed software.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 305: Database columns for AccessedSoftwareOccurrence table

Database Column	Details
AccessedSoftwareID	*Type:* big integer. Key An identifier for an accessed software. Foreign key to the AccessedSoftware
ServerComputerID	*Type:* integer. Key An identifier for a server record. Foreign key to the ComplianceComputer table.
AccessDate	*Type:* datetime. Nullable Date on which access has occurred.
LicenseDate	*Type:* datetime. Key Date which will be used for licensing purposes.
InventoryDate	*Type:* datetime. Key Date on which access occurrence was recorded.
AccessCount	*Type:* integer Number of access occurrences on this date.

AccessedSoftwareUserAssignmentReasons Table

AccessedSoftwareUserAssignmentReasons lists all license assignments attempted for client access of an application after the license reconcile is over.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 306: Database columns for `AccessedSoftwareUserAssignmentReasons` table

Database Column	Details
`AccessingUserID`	*Type:* integer. Key. Nullable The accessing device under examination. Foreign key to the `AccessingDevice` table.
`ComplianceUserID`	*Type:* integer. Key The compliance computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`MultiProductLicensePhase`	*Type:* boolean This field is set to `True` when this installation is licensed during the multi-product license phase of license reconcile.
`Order`	*Type:* integer The order this license was attempted to be assigned to this installation.
`LAConsReasonID`	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
`LicenseAssignment FailureReasonID`	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
`AddedBySQLPhase`	*Type:* boolean Specifies whether the licence allocation was done by SQL or C# code. This is an internal field that can be used when troubleshooting license assignments.

AccessingDevice Table

`AccessingDevice` stores information about devices which are accessing a software on the server.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 307: Database columns for `AccessingDevice` table

Database Column	Details
`AccessingDeviceID`	*Type:* integer. Key. Generated ID A unique identifier for a `AccessingDevice`
`ComplianceComputerID`	*Type:* integer. Key. Nullable An identifier for a compliance computer record. Foreign key to the `ComplianceComputer` table.
`IPAddress`	*Type:* text (max 256 characters). Key. Nullable IP address of the of the device.
`ComputerName`	*Type:* text (max 256 characters). Key. Nullable Computer name.
`SerialNo`	*Type:* text (max 100 characters). Nullable The serial number of the computer.
`Domain`	*Type:* text (max 100 characters). Nullable The domain name of the computer.

AccessingDeviceSnapshot Table

The `AccessingDeviceSnapshot` table lists all the snapshotted accessing devices.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 308: Database columns for `AccessingDeviceSnapshot` table

Database Column	Details
`AccessingDeviceID`	*Type:* integer. Key A unique identifier for a `AccessingDevice`.
`ComplianceComputerID`	*Type:* integer. Nullable An identifier for a compliance computer record. Foreign key to the `ComplianceComputerSnapshot` table.
`IPAddress`	*Type:* text (max 256 characters). Nullable IP address of the of the device.
`ComputerName`	*Type:* text (max 256 characters). Nullable Computer name.

Database Column	Details
LicenseMeasurementID	*Type:* integer. Key The snapshot ID. Foreign key to the LicenseMeasurement table.

AccessingUser Table

AccessingUser stores information about users which are accessing a software on the server.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 309: Database columns for AccessingUser table

Database Column	Details
AccessingUserID	*Type:* integer. Key. Generated ID A unique identifier for a AccessingUser
ComplianceUserID	*Type:* integer. Key. Nullable An identifier for a compliance user record. Foreign key to the ComplianceUser table.
UserName	*Type:* text (max 256 characters). Key Username of the end user.
DomainName	*Type:* text (max 100 characters). Key. Nullable Domain name of the end user.

AccessingUserSnapshot Table

The AccessingUserSnapshot table lists all the snapshotted accessing users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 310: Database columns for AccessingUserSnapshot table

Database Column	Details
AccessingUserID	*Type:* integer. Key A unique identifier for a AccessingUser.

Database Column	Details
`ComplianceUserID`	*Type:* integer. Nullable An identifier for a compliance user record. Foreign key to the `ComplianceUserSnapshot` table.
`UserName`	*Type:* text (max 256 characters) Username of the end user.
`DomainName`	*Type:* text (max 100 characters). Nullable Domain name of the end user.
`LicenseMeasurementID`	*Type:* integer. Key The snapshot ID. Foreign key to the `LicenseMeasurement` table.

AssignmentStatus Table

`AssignmentStatus` lists all license assignment status .

Table 311: Database columns for `AssignmentStatus` table

Database Column	Details
`AssignmentStatusID`	*Type:* integer. Key A unique identifier for each `AssignmentStatus`. Possible values and the corresponding default strings are: • 1 = Consumed • 2 = Rejected • 3 = Not assessed
`ResourceName`	*Type:* text (max 256 characters) The unique name of the localizable resource string representing a license assignment status.
`DefaultValue`	*Type:* text (max 100 characters) The link to the license this install has been counted against. Foreign key to the `SoftwareLicense` table.

AssignmentStatusFailureReasonMapping Table

`AssignmentStatusFailureReasonMapping` mapping of the failures to the respective status .

Table 312: Database columns for `AssignmentStatusFailureReasonMapping` table

Database Column	Details
`AssignmentStatusID`	*Type:* integer. Key Foreign key from the `AssignmentStatus` table.
`LicenseAssignment FailureReasonID`	*Type:* integer. Key Foreign key from the `LicenseAssignmentFailureReason` table.

ClientAccessSourceType Table

`ClientAccessSourceType` is a static table listing the types of client access source type that can be used to determine whether the evidence is collection from which source.

Table 313: Database columns for `ClientAccessSourceType` table

Database Column	Details
`ClientAccessSourceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ClientAccessSourceType`. Possible values and the corresponding default strings are: • 1 = Unknown • 2 = UAL • 3 = Exchange • 4 = Lync • 5 = SCCM • 6 = Manual • 7 = SharePoint • 8 = SaaS
`TypeResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a clinet access source type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`ImporterString`	*Type:* text (max 100 characters). Key The text value provided by adapters when importing client access source type.

ClientAccessedAccessEvidence Table

`ClientAccessedAccessEvidence` lists access evidence from user and device that occurred on a server computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 314: Database columns for `ClientAccessedAccessEvidence` table

Database Column	Details
`ClientAccessedAccessEvidenceID`	*Type:* big integer. Key. Generated ID A unique identifier for a `ClientAccessedAccessEvidence`
`AccessEvidenceID`	*Type:* integer. Key An identifier for an access evidence record. Foreign key to the `AccessEvidence` table.
`AccessingUserID`	*Type:* integer. Key. Nullable An identifier for a accessing user record. Foreign key to the `AccessingUser` table.
`AccessingDeviceID`	*Type:* integer. Key. Nullable An identifier for a accessing device record. Foreign key to the `AccessingDevice` table.
`ServerComputerID`	*Type:* integer. Key An identifier for a server record. Foreign key to the `ComplianceComputer` table.
`MaxAccessCount`	*Type:* integer. Nullable Maximum access count recorded for this evidence.
`LastAccessCount`	*Type:* integer. Nullable Last access count recorded for this evidence.
`LastAccessDate`	*Type:* datetime. Nullable Last access date recorded for this evidence.
`LastInventoryDate`	*Type:* datetime. Nullable Last time access inventory was collected for this evidence.
`ClientAccessSourceTypeID`	*Type:* integer. Key Referencing to the client access source type.

ClientAccessedAccessOccurrence Table

`ClientAccessedAccessOccurrence` lists access occurrences for access evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 315: Database columns for `ClientAccessedAccessOccurrence` table

Database Column	Details
`ClientAccessedAccess EvidenceID`	*Type:* big integer. Key An identifier for an accessed access evidence. Foreign key to the `ClientAccessedAccessEvidence`
`AccessDate`	*Type:* datetime. Nullable Date on which access has occurred.
`InventoryDate`	*Type:* datetime. Key Date on which access occurrence was recorded.
`LicenseDate`	*Type:* datetime. Key Date which will be used for licensing purposes.
`AccessCount`	*Type:* integer Number of access occurrences on this date.

Cluster Table

The `Cluster` table stores information about a logical group of computers which form a cluster.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 316: Database columns for `Cluster` table

Database Column	Details
`ClusterID`	*Type:* integer. Key. Generated ID A unique identifier for the cluster.
`ParentClusterID`	*Type:* integer. Key. Nullable An optional link back to a parent cluster.

Database Column	Details
`ExternalName`	*Type:* text (max 256 characters). Key. Nullable The identifier of the cluster in the external cluster management system.
`Name`	*Type:* text (max 256 characters). Key The user-visible name of the cluster.
`Namespace`	*Type:* text (max 256 characters). Key. Nullable The name of the domain or datacenter containing the cluster.
`ClusterTypeID`	*Type:* integer. Key Foreign key to the `ClusterType` table.
`ComplianceComputer InventorySourceTypeID`	*Type:* integer Whether this cluster has ever been reported in inventory, or has been manually created and maintained. Foreign key to the `ComplianceComputerInventorySourceType` table.
`InventoryDate`	*Type:* datetime. Nullable The date the computer last had inventory reported.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The name of the operator who last updated the computer details.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the cluster was created.
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory.
`DRS`	*Type:* boolean. Nullable Whether Distributed Resource Scheduler (DRS) is enabled
`DPM`	*Type:* boolean. Nullable Whether Distributed Power Management (DPM) is enabled

ClusterComputer Table

The `ClusterComputer` table stores information about the relationship of computers to a cluster.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 317: Database columns for `ClusterComputer` table

Database Column	Details
`ClusterComputerID`	*Type:* integer. Key. Generated ID A unique identifier for the cluster computer.
`ClusterID`	*Type:* integer. Key Foreign key to the `Cluster` table.
`ComplianceComputerID`	*Type:* integer. Key Foreign key to the `ComplianceComputer` table.
`ClusterNodeTypeID`	*Type:* integer Foreign key to the `ClusterNodeType` table.
`ComplianceComputer InventorySourceTypeID`	*Type:* integer Whether this cluster computer relationship has ever been reported in inventory, or has been manually created and maintained. Foreign key to the `ComplianceComputerInventorySourceType` table.

ClusterHostAffinityRule Table

The `ClusterHostAffinityRule` table stores rules that define whether there is affinity between different VM groups and host groups within a cluster.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 318: Database columns for `ClusterHostAffinityRule` table

Database Column	Details
`ClusterHostAffinityRuleID`	*Type:* integer. Key. Generated ID A unique identifier for each `ClusterHostAffinityRule`.
`ClusterHostAffinity RuleTypeID`	*Type:* integer A unique identifier indicating a type of Cluster Host Affinity Rule.
`Name`	*Type:* text (max 256 characters). Key The name assigned to an affinity rule.

Database Column	Details
HostGroupClusterID	*Type:* integer The unique identifier of the host group to which the affinity rule applies. Foreign key to the Cluster table.
VMGroupClusterID	*Type:* integer The unique identifier of the VM group to which the affinity rule applies. Foreign key to the Cluster table.
ClusterID	*Type:* integer. Key Foreign key to the Cluster table.
ComplianceComputer InventorySourceTypeID	*Type:* integer Whether this cluster host affinity rule has ever been reported in inventory, or has been manually created and maintained. Foreign key to the ComplianceComputerInventorySourceType table.

ClusterHostAffinityRuleType Table

ClusterHostAffinityRuleType is a static table listing all of the types of cluster host affinity rules.

Table 319: Database columns for ClusterHostAffinityRuleType table

Database Column	Details
ClusterHostAffinity RuleTypeID	*Type:* integer. Key. Generated ID A unique identifier for each ClusterHostAffinityRuleType. Possible values and the corresponding default strings are: • 1 = must run on (VMs in the LHS group MUST run on hosts specified in the RHS group) • 2 = must not run on (VMs in the LHS group MUST NOT run on any of the hosts specified in the RHS group)
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the type of a cluster host affinity rule. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ClusterNodeType Table

ClusterNodeType is a static table listing all of the roles a computer can have in a cluster.

Table 320: Database columns for `ClusterNodeType` table

Database Column	Details
`ClusterNodeTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ClusterNodeType`. Possible values and the corresponding default strings are: • 1 = Active (a node that is powered on and in use.) • 2 = Passive (a node that is powered on but not in use unless an active node fails over to it) • 3 = Hot (an active node–IBM nomenclature) • 4 = Warm (a passive node–IBM nomenclature) • 5 = Cold (a node that is powered off–IBM nomenclature)
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a cluster node type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ClusterType Table

`ClusterType` is a static table listing all of the types of a cluster.

Table 321: Database columns for `ClusterType` table

Database Column	Details
`ClusterTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ClusterType`. Possible values and the corresponding default strings are: • 1 = vMotion (a mobility cluster based on VMWare ESX technology) • 2 = Hyper-V (a mobility cluster based on Microsoft's Hyper-V virtualization technology) • 5 = Oracle VM (a cluster based on Oracle VM virtualization technology)
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a cluster type. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of cluster, stored in XML format.

ComplianceComputerSnapshot Table

The `ComplianceComputerSnapshot` table lists all the snapshotted computers.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 322: Database columns for `ComplianceComputerSnapshot` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The snapshotted ComplianceComputerID.
`ComputerName`	*Type:* text (max 256 characters). Nullable The snapshotted computer name.
`Domain`	*Type:* text (max 256 characters). Nullable The snapshotted computer domain name.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted LocationID.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted BusinessUnitID.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted CostCenterID.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted CategoryID.
`LicenseMeasurementID`	*Type:* integer. Key The snapshot ID. Foreign key to the `LicenseMeasurement` table.

ComplianceComputerTag Table

Reserved for future development.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 323: Database columns for `ComplianceComputerTag` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key Foreign key to the `ComplianceComputer` table
`TagID`	*Type:* integer. Key Foreign key to the `Tag` table.

ComplianceUserSnapshot Table

The `ComplianceUserSnapshot` table lists all the users for each snapshot.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 324: Database columns for `ComplianceUserSnapshot` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key The snapshotted ComplianceUserID.
`UserName`	*Type:* text (max 256 characters). Nullable The snapshotted user name.
`Domain`	*Type:* text (max 256 characters). Nullable The snapshotted user domain name.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted LocationID.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted BusinessUnitID.

Database Column	Details
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted CostCenterID.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable The snapshotted CategoryID.
`LicenseMeasurementID`	*Type:* integer. Key The snapshot ID. Foreign key to the `LicenseMeasurement` table.

ComplianceUserTag Table

Reserved for future use.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 325: Database columns for `ComplianceUserTag` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key Foreign key to the `ComplianceUser` table.
`TagID`	*Type:* integer. Key Foreign key to the `Tag` table.

DatabaseMutex Table

The `DatabaseMutex` table lists all current database mutexes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 326: Database columns for `DatabaseMutex` table

Database Column	Details
`DatabaseMutexID`	*Type:* integer. Key. Generated ID A unique identifier for the database mutex.

Database Column	Details
Name	*Type:* text (max 256 characters). Key The name of the mutex.

EndOfSupportLife Table

Table 327: Database columns for `EndOfSupportLife` table

Database Column	Details
`EndOfSupportLifeID`	*Type:* integer. Key. Generated ID A unique identifier.
`SoftwareLifeCycleID`	*Type:* integer. Key The software life cycle this EOSL belongs to. Foreign key to the `SoftwareLifeCycle` table.
`EndOfSupportLifeNameID`	*Type:* integer. Key The name of the EOSL. Foreign key to the `EndOfSupportLifeName` table.
`EndDate`	*Type:* datetime. Nullable The support end date.
`Notes`	*Type:* text. Nullable Notes for this end of support life

EndOfSupportLifeName Table

Table 328: Database columns for `EndOfSupportLifeName` table

Database Column	Details
`EndOfSupportLifeNameID`	*Type:* integer. Key. Generated ID A unique identifier for EOSL name.
Name	*Type:* text (max 256 characters). Key The EOSL's name

EntitlementRecommendation Table

`EntitlementRecommendation` is a table listing all of the recommendations that have been made to link entitlements to licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 329: Database columns for `EntitlementRecommendation` table

Database Column	Details
`EntitlementRecommendationID`	*Type:* integer. Key. Generated ID A unique identifier for this recommendation.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable The license affected by this recommendation, null if a new license is being created. Foreign key to the `SoftwareLicense` table.
`SoftwareLicenseDefinitionID`	*Type:* integer. Key. Nullable The license defintion of the new license being created. Foreign key to the `SoftwareLicenseDefinition` table.
`SoftwareLicenseDefinition`	*Type:* text. Nullable Encrypted XML definition of the customised license being created if any.
`MaintenanceDefinition`	*Type:* text. Nullable Encrypted XML definition of the maintenance being applied to the license associated with this recommendation.
`ContractID`	*Type:* integer. Key. Nullable The contract affected by this recommendation, if any. Foreign key to the `Contract` table.
`MaintenanceContractID`	*Type:* integer. Nullable The contract providing maintenance for this recommendation, if any. Foreign key to the `Contract` table.
`ProcessActionID`	*Type:* integer. Key. Nullable The action that is recommended by this recommendation. Foreign key to the `ProcessAction` table.
`EntitlementRecommendationStateID`	*Type:* integer. Nullable The state that the recommendation is in. Foreign key to the `EntitlementRecommendationState` table.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.

Database Column	Details
UpdatedUser	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
UpdatedDate	*Type:* datetime. Nullable The date the record was last updated.
DoTransferSoftwareLicenseAllocations	*Type:* boolean. Nullable Indicates whether to transfer Group Assignments and Allocations when performing an upgrade and all the entitlements are transferred to the new license.

EntitlementRecommendationState Table

EntitlementRecommendationState is a static table listing all of the states a entitlement recommendation or transaction can be in.

Table 330: Database columns for EntitlementRecommendationState table

Database Column	Details
EntitlementRecommendationStateID	*Type:* integer. Key. Generated ID A unique identifier for each EntitlementRecommendationState. Possible values and the corresponding default strings are: • 1 = Automatically recommended • 2 = Manually created • 3 = Edited by an operator • 4 = Accepted by an operator or automatically • 5 = Rolled back by an operator • 6 = Deferred by an operator • 7 = Failed to be accepted.
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the entitlement recommendation's state. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

EntitlementTransaction Table

`EntitlementTransaction` is a table listing all of the recommendations that have been made to link entitlements to licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 331: Database columns for `EntitlementTransaction` table

Database Column	Details
`EntitlementTransactionID`	*Type:* integer. Key. Generated ID A unique identifier for this transaction.
`Entitlement RecommendationID`	*Type:* integer. Key. Nullable The recommendation this transaction is related to if any. Foreign key to the `EntitlementRecommendation` table.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable The license affected by this recommendation. If a new license is being created from a recommendation but the recommendation is pending, the value of this field is null. The license identified depends on the EntitlementTransactionType. For a recommendation, this could be the license being updated (the “from” license) or it could be the new license (the “to” license). Foreign key to the `SoftwareLicense` table.
`PurchaseOrderDetailID`	*Type:* integer. Key. Nullable The purchase order line associated with this transaction. Foreign key to the `PurchaseOrderDetail` table.
`Adjustment`	*Type:* integer. Nullable The (potentially partial) amount of the purchased license quantity that is being applied to the license.
`OtherCandidates`	*Type:* boolean. Nullable Whether there were other licenses which could have been recommended.
`EntitlementTransaction TypeID`	*Type:* integer. Nullable The type of the transaction. Foreign key to the `EntitlementTransactionType` table.
`Entitlement RecommendationStateID`	*Type:* integer. Key. Nullable The state that the transaction is in. Foreign key to the `EntitlementRecommendationState` table.

Database Column	Details
`IsDeferred`	*Type:* boolean Flags the entitlement transaction whether it is deferred for later processing.
`TransactionUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`TransactionDate`	*Type:* datetime. Nullable The date the record was last updated.
`PreviousMaintenance Definition`	*Type:* text. Nullable Encrypted XML definition of the maintenance previously applied to the license associated with this transaction.
`PreviousMaintenance ContractID`	*Type:* integer. Nullable The ID of the contract previously giving maintenance to the license associated with this transaction. Foreign key to the `Contract` table.
`LicenseNameMatched`	*Type:* boolean Indicates whether or not there was a license name match.
`PrimaryApplicationMatched`	*Type:* boolean Indicates whether or not there was a primary application match.
`AnyApplicationMatched`	*Type:* boolean Indicates whether or not there was a match on any application.
`MaintenanceSettings Matched`	*Type:* boolean Indicates whether or not there was a match based on maintenance settings.
`EnterpriseGroupMatched`	*Type:* boolean Indicates whether or not there was a match based on enterprise groups.
`NumberOfVersionsDifferent`	*Type:* integer Indicated the number of versions between the version being upgraded to from the version being upgraded from.
`EntitlementTransaction StateID`	*Type:* integer The state of the transaction. Foreign key to the `EntitlementTransactionState` table.
`AdjustmentDefault`	*Type:* integer. Nullable The default amount of the purchased license quantity that is being applied to the license.

Database Column	Details
AllowMaintenanceGap	*Type:* boolean Will determine if the end users will be alerted about a gap in maintenance for this purchase. If this is set to 0, then an alert will be generated if a gap is detected. if it is set to 1, then no alert will be generated.

EntitlementTransactionOtherCandidate Table

EntitlementTransactionOtherCandidate is a table listing all of the other possible license recommendations that have been made to for entitlements.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 332: Database columns for EntitlementTransactionOtherCandidate table

Database Column	Details
EntitlementTransaction OtherCandidateID	*Type:* integer. Key. Generated ID A unique identifier for this possible candidate.
EntitlementTransactionID	*Type:* integer. Key The entitlement the recommendation belongs to.
SoftwareLicenseID	*Type:* integer. Key The license affected by this recommendation.
UpgradeFrom	*Type:* boolean Indicates whether this license was a candidate to upgrade from or not.
LicenseNameMatched	*Type:* boolean Indicates whether or not there was a license name match.
PrimaryApplicationMatched	*Type:* boolean Indicates whether or not there was a primary application match.
AnyApplicationMatched	*Type:* boolean Indicates whether or not there was a match on any application.
MaintenanceSettings Matched	*Type:* boolean Indicates whether or not there was a match based on maintenance settings.
EnterpriseGroupMatched	*Type:* boolean Indicates whether or not there was a match based on enterprise groups.

Database Column	Details
`NumberOfVersionsDifferent`	*Type:* integer Indicated the number of versions between the version being upgraded to from the version being upgraded from.

EntitlementTransactionState Table

`EntitlementTransactionState` is a static table listing all of the states that can be associated with purchased entitlements.

Table 333: Database columns for `EntitlementTransactionState` table

Database Column	Details
`EntitlementTransaction StateID`	*Type:* integer. Key. Generated ID A unique identifier for each `EntitlementTransactionState`. Possible values and the corresponding default strings are: • 1 = Enabled • 2 = Disabled • 3 = Always enabled • 4 = Not contributing.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the enabled state of the transaction. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the type resource string has no translation.

EntitlementTransactionType Table

`EntitlementTransactionType` is a static table listing all of the types of transactions that can be performed associating purchased entitlements to a license.

Table 334: Database columns for `EntitlementTransactionType` table

Database Column	Details
`EntitlementTransaction TypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `EntitlementTransactionType`. Possible values and the corresponding default strings are: • 1 = Purchased entitlements added to license • 2 = Purchased entitlements removed from license • 3 = Purchased entitlements taken from this license for upgrade purposes • 4 = Entitlements adjusted manually on the license by an operator • 5 = Maintenance entitlements adjusted on the license. • 6 = Maintenance entitlements adjusted manually on the license. • 7 = Upgrade entitlements adjusted manually on the license.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the type of transaction. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the type resource string has no translation.

EvidenceExistenceRule Table

`EvidenceExistenceRule` is a static table listing the rules to be applied to file evidence and its relationship to a software (application) title.

Table 335: Database columns for `EvidenceExistenceRule` table

Database Column	Details
`EvidenceExistenceRuleID`	*Type:* integer. Key. Generated ID Unique identifier for each `EvidenceExistenceRule`. Possible values and the corresponding default strings are: • 1 = Required (the file evidence must be present for the title to be considered installed) • 2 = Not for recognition (not used for recognizing application installations - the presence of this file evidence does not guarantee installation of the title) • 3 = Not allowed (if the file evidence is present, the title is not installed). • 4 = At least one (the presence of any of the file evidence identified this way is enough for the title to be considered installed).
`RuleResourceString`	*Type:* text (max 50 characters). Key The unique name of the localizable resource string representing an evidence rule. Foreign key to the `ComplianceResourceString` table.
`RuleDefaultString`	*Type:* text (max 100 characters) The text to display if the rule resource string has no translation.

EvidenceStatus Table

The collection of status values for installation evidence.

Table 336: Database columns for `EvidenceStatus` table

Database Column	Details
`EvidenceStatusID`	*Type:* integer. Key. Generated ID A unique identifier for an evidence status. Possible values (and associated default names) are: • 1 = Active • 2 = Inactive • 3 = Unassigned • 4 = Ignored • 5 = Assigned.

Database Column	Details
`StatusResourceString`	*Type:* text (max 50 characters). Key The name of the resource string containing the text to display on the user interface.
`StatusDefaultString`	*Type:* text (max 100 characters) The value to display if there is no resource string available for this status.

FNMEAFeature Table

`FNMEAFeature` records additional license features, associated with a specific license, that have been imported from FlexNet Manager for Engineering Applications.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 337: Database columns for `FNMEAFeature` table

Database Column	Details
`FNMEAFeatureID`	*Type:* integer. Key. Generated ID A unique identifier for the FNM-EA feature record.
`Name`	*Type:* text (max 256 characters) Name of the feature.
`Version`	*Type:* text (max 60 characters). Nullable Version of the feature.
`PublisherID`	*Type:* integer. Nullable The publisher of the license associated with this feature. Foreign key to the `Vendor` table.
`NumberPurchased`	*Type:* integer The quantity of purchased feature entities.
`NumberInstalled`	*Type:* integer The quantity of software installations accounted for by this feature.
`SoftwareLicenseComplianceStatusID`	*Type:* integer The compliance status of the license associated with this feature. Defaults to `Compliant`. Foreign key to the `SoftwareLicenseComplianceStatus` table.

FNMEALicensedFeature Table

`FNMEALicensedFeature` associated imported FlexNet Manager for Engineering Applications features with software licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 338: Database columns for `FNMEALicensedFeature` table

Database Column	Details
`FNMEAFeatureID`	*Type:* integer. Key The feature associated with a license. Foreign key to the `FNMEAFeature` table.
`SoftwareLicenseID`	*Type:* integer. Key The license associated with a feature. Foreign key to the `SoftwareLicense` table.
`QuantityPerLicense`	*Type:* integer The quantity of feature entitlements per associated license purchased.
`ProductID`	*Type:* text (max 256 characters). Key The external identifier of the product the linked feature is a part of.
`ComplianceConnectionID`	*Type:* integer. Key An identifier for the data source the product has been imported from.

FileEvidenceCompany Table

`FileEvidenceCompany` contains the company names appearing in the headers of files used as evidence that an application is installed.

Table 339: Database columns for `FileEvidenceCompany` table

Database Column	Details
`FileEvidenceCompanyID`	*Type:* integer. Key. Generated ID A unique identifier for this company.
`Company`	*Type:* text (max 100 characters). Key The name of the company.

FileEvidenceEx Table

The `FileEvidenceEx` table contains additional information on the file evidence managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 340: Database columns for `FileEvidenceEx` table

Database Column	Details
`FileEvidenceID`	*Type:* integer. Key A unique identifier for an file evidence record.
`OperatorManageStateID`	*Type:* integer. Nullable The management responsibility for this information. Foreign key to the `OperatorManageState` table.
`Ignored`	*Type:* boolean. Nullable Set this field to `True` if the file evidence is not used for application recognition.

FileEvidenceFile Table

`FileEvidenceFile` contains the names of the files used as evidence that an application is installed.

Table 341: Database columns for `FileEvidenceFile` table

Database Column	Details
`FileEvidenceFileID`	*Type:* integer. Key. Generated ID A unique identifier for the file.
`FileName`	*Type:* text (max 256 characters). Key The name of the file.

FileEvidenceLanguage Table

`FileEvidenceLanguage` contains the language names appearing in headers of files used as evidence that an application is installed.

Table 342: Database columns for `FileEvidenceLanguage` table

Database Column	Details
`FileEvidenceLanguageID`	*Type:* integer. Key. Generated ID A unique identifier for this language.
`Language`	*Type:* text (max 200 characters). Key The name of the language.

FileEvidenceMatchCount Table

`FileEvidenceMatchCount` tracks the number of times that each file evidence (rule) has been detected as installed and recorded in the data source. A separate count is kept for each file evidence rule, and for each data source.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 343: Database columns for `FileEvidenceMatchCount` table

Database Column	Details
`FileEvidenceMatchCountID`	*Type:* integer. Key. Generated ID A synthetic unique identifier is required, since `ComplianceConnectionID`, being nullable, cannot be included in the primary key.
`FileEvidenceID`	*Type:* integer. Key The file evidence rule being matched. Foreign key to the `NewFileEvidence` table.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The data source where the match is occurring. Foreign key to the `ComplianceConnection` table.
`MatchedCount`	*Type:* integer The number of installed files in this data source matching this file evidence rule.
`InstallCount`	*Type:* integer The number of physical application installations recognized in this data source using this file evidence rule.

FileEvidencePath Table

`FileEvidencePath` contains the file paths to files used as evidence that an application is installed.

Table 344: Database columns for `FileEvidencePath` table

Database Column	Details
`FileEvidencePathID`	*Type:* integer. Key. Generated ID A unique identifier for this path.
`FilePath`	*Type:* text (max 400 characters). Key The content of the file path.

GroupSnapshot Table

The `GroupSnapshot` table lists all the snapshotted groups.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 345: Database columns for `GroupSnapshot` table

Database Column	Details
`GroupID`	*Type:* integer. Key The snapshotted GroupID.
`GroupExID`	*Type:* text (max 128 characters). Key The snapshotted GroupExID.
`Path`	*Type:* text (max 500 characters) The snapshotted Path.
`LicenseMeasurementID`	*Type:* integer. Key The snapshot ID. Foreign key to the `LicenseMeasurement` table.

ImporterRun Table

The `ImporterRun` table lists all previously run imports.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 346: Database columns for `ImporterRun` table

Database Column	Details
`ImporterRunID`	*Type:* integer. Key. Generated ID A unique identifier for the import run.
`LicenseMeasurementID`	*Type:* integer. Key. Nullable The LicenseMeasurementID if a license reconcile was performed. Foreign key to the `LicenseMeasurement` table.
`StartDate`	*Type:* datetime. Nullable The time the import was started.
`EndDate`	*Type:* datetime. Nullable The time the import was completed.
`ImportSourcesAppliedDate`	*Type:* datetime. Nullable If non-licensing writers ran and completed successfully, this field will be set to the date/time of their completion. In effect, it records the application of data from the importer staging tables in to the core tables. This is the case even if the record as a whole is marked as a failure, as the writers processing will have already completed.
`Arguments`	*Type:* text (max 1024 characters) The command line arguments to the import.
`RunAs`	*Type:* text (max 1024 characters) The user who performed the import.
`Comment`	*Type:* text (max 1024 characters). Nullable Comments related to the import.
`EventLogSummaryID`	*Type:* integer. Key. Nullable The EventLogSummaryID for the import. Foreign key to the `EventLogSummary` table.
`Success`	*Type:* boolean. Key. Nullable Determines whether the import completed successfully.

ImporterStepValidationIssue Table

The `ImporterStepValidationIssue` table lists any validation issues that occurred during an import, that the user may need to review.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 347: Database columns for `ImporterStepValidationIssue` table

Database Column	Details
`ImporterStepValidationIssueID`	*Type:* integer. Key. Generated ID A unique identifier for the import validation.
`ImporterRunID`	*Type:* integer. Key Foreign key to the `ImporterRun` table.
`ComplianceConnectionID`	*Type:* integer. Key Foreign key to the `ComplianceConnection` table.
`ProcedureName`	*Type:* text (max 256 characters). Nullable The procedure that contains the issue.
`StepName`	*Type:* text (max 512 characters). Nullable The step that contains the issue.
`RowSkipped`	*Type:* boolean Source to object validatation issue specifing if row skipped.
`ColErrorReason`	*Type:* integer. Nullable Source to object validatation issue specifing reason for error on particular row.
`ColumnName`	*Type:* text (max 128 characters). Nullable Column name of the failed source to object validatation issue.
`RowNumber`	*Type:* big integer. Nullable Row number of the failed source to object validatation issue.
`AffectedItem`	*Type:* text (max 512 characters). Nullable An optional description for any further related item.
`ImporterStepValidationIssueTypeID`	*Type:* integer. Nullable Foreign key to the `ImporterStepValidationIssueType` table.
`OccurrenceDate`	*Type:* datetime. Nullable The time the issue was raised.

ImporterStepValidationIssueType Table

`ImporterStepValidationIssueType` is a static table listing all of the validation issues that can occur on a `ComplianceConnection`.

Table 348: Database columns for `ImporterStepValidationIssueType` table

Database Column	Details
`ImporterStepValidationIssueTypeID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the ImporterStepValidationIssueType record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

InstalledFileEvidence Table

`InstalledFileEvidence` lists file evidence that has been installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 349: Database columns for `InstalledFileEvidence` table

Database Column	Details
`FileEvidenceID`	*Type:* integer. Key An identifier for a file evidence record. Foreign key to the `NewFileEvidence` table.
`ComplianceComputerID`	*Type:* integer. Key The managed computer on which this evidence was found. Foreign key to the `ComplianceComputer` table.
`AccessModeID`	*Type:* integer. Key The state an application was considered accessed. Foreign key to the `AccessMode` table.

InstalledInstallerAttribute Table

`InstalledInstallerAttribute` installer evidence attributes that exist on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 350: Database columns for `InstalledInstallerAttribute` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key An identifier for an installer evidence record. Foreign key to the `InstallerEvidence` table.
`ComplianceComputerID`	*Type:* integer. Key An identifier for a computer record. Foreign key to the `ComplianceComputer` table.
`InstanceName`	*Type:* text (max 256 characters). Key. Nullable The name of the instance on the computer where this installer evidence was found.
`AttributeID`	*Type:* integer. Key The installer evidence attribute. Foreign key to the `Attribute` table.
`Value`	*Type:* text The value of the attribute.

InstalledInstallerEvidence Table

`InstalledInstallerEvidence` lists installer evidence that has been installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 351: Database columns for `InstalledInstallerEvidence` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key An identifier for an installer evidence record. Foreign key to the `InstallerEvidence` table.
`ComplianceComputerID`	*Type:* integer. Key An identifier for a computer record. Foreign key to the `ComplianceComputer` table.

Database Column	Details
`InstanceName`	*Type:* text (max 256 characters). Key. Nullable The name of the instance on the computer where this installer evidence was found.
`InstallDate`	*Type:* datetime. Nullable The install date of the installer evidence.
`DiscoveryDate`	*Type:* datetime. Nullable The date that the installer evidence was first seen.
`AccessModeID`	*Type:* integer. Key The state an application was considered accessed. Foreign key to the `AccessMode` table.

InstalledInstanceReplacement Table

`InstalledInstanceReplacement` tracks the particular installations instances where a software suite replaced the installation record of its member application.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 352: Database columns for `InstalledInstanceReplacement` table

Database Column	Details
`InstanceID`	*Type:* integer. Key The installation instance of the software suite. Foreign key to the `Instance` table.
`ReplacedSoftwareTitleID`	*Type:* integer. Key Software title that has been replaced by its parent suite. Foreign key to the `SoftwareTitle` table.

InstalledSoftwareData Table

`InstalledSoftware` lists all the installations of an application (as defined in the `SoftwareTitle` table).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 353: Database columns for `InstalledSoftwareData` table

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key. Generated ID A unique identifier for an installed software record.
`ComplianceComputerID`	*Type:* integer. Key The computer on which the software is installed. Foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key The software that is installed. Foreign key to the `SoftwareTitle` table.
`IsUsed`	*Type:* boolean Set this field to `True` if the software title is installed according to usage thresholds in the `SoftwareTitle` table.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable The link to the license this install has been counted against. Foreign key to the `SoftwareLicense` table.
`SoftwareLicense AllocationID`	*Type:* integer. Key. Nullable The link to the license allocation this installation has consumed. Foreign key to the `SoftwareLicenseAllocation` table.
`IsLicensed`	*Type:* boolean Set this field to `True` when this installation is licensed.
`PointsUsed`	*Type:* integer. Nullable The number of points this installation consumes on a points-based license.
`RawPointsUsed`	*Type:* integer. Nullable The number of points this installation consumes on a points-based license before exemptions are considered.
`InstallDate`	*Type:* datetime. Nullable The install date of the software.
`DiscoveryDate`	*Type:* datetime. Key. Nullable The date that the software was first seen.
`LastUsedDate`	*Type:* datetime. Nullable The date that the software was last used.
`PointsCalculated`	*Type:* integer The number of calculated points this installation consumes.

InstalledSoftwareRemoval Table

`InstalledSoftwareRemoval` table keeps track of software titles that have been recognised, but then removed due to precedence. This is typically because a higher quality (more specific) title has been found.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 354: Database columns for `InstalledSoftwareRemoval` table

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key Installation record for lower quality title. Foreign key to the `InstalledSoftware` table.
`RemovedSoftwareTitleID`	*Type:* integer. Key Software title whose installation is now being ignored due to the presence of a higher quality title. Foreign key to the `SoftwareTitle` table.

InstalledSoftwareReplacement Table

`InstalledSoftwareReplacement` tracks which individual application installation records have (ever) been subsumed by recognition of their parent software suite installed on the same computer. Only the suite and its member application are linked here.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 355: Database columns for `InstalledSoftwareReplacement` table

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key The suite's installation record. Foreign key to the `InstalledSoftware` table.
`ReplacedSoftwareTitleID`	*Type:* integer. Key The software title that has been replaced by its parent suite. Foreign key to the `SoftwareTitle` table.

InstalledSoftwareUsageData Table

`InstalledSoftwareUsage` records the end-users who are using a piece of software installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 356: Database columns for `InstalledSoftwareUsageData` table

Database Column	Details
`InstalledSoftwareUsageID`	*Type:* integer. Key. Generated ID The unique identifier for this record.
`ComplianceUserID`	*Type:* integer. Key. Nullable The end-user using the application. Foreign key to the `ComplianceUser` table.
`SoftwareLicenseID`	*Type:* integer. Nullable The license that covers this installation. Foreign key to the `SoftwareLicense` table.
`SoftwareLicense AllocationID`	*Type:* integer. Key. Nullable A link to any individual allocation that this installation consumes. Foreign key to the `SoftwareLicenseAllocation` table.
`IsLicensed`	*Type:* boolean Set this field to `True` if this usage is licensed.
`UsageSessions`	*Type:* integer The number of sessions for (or times that the application was used by) this end-user on this computer.
`UsageActiveTime`	*Type:* integer The amount of time this application was in active use (in the foreground) for this end-user on this computer.
`ComplianceComputerID`	*Type:* integer. Key The application. Foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key The application. Foreign key to the `SoftwareTitle` table.
`LastUsedDate`	*Type:* datetime. Nullable The date that the installed software was last used.
`AccessModeID`	*Type:* integer. Key The date that the installed software was last used.

InstalledWMIEvidence Table

`InstalledWMIEvidence` lists WMI evidence that has been installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 357: Database columns for `InstalledWMIEvidence` table

Database Column	Details
`WMIEvidenceID`	*Type:* integer. Key An identifier for a WMI evidence record. Foreign key to the `WMIEvidence` table.
`ComplianceComputerID`	*Type:* integer. Key An identifier for a computer record. Foreign key to the `ComplianceComputer` table.
`AccessModeID`	*Type:* integer. Key The state an application was considered accessed. Foreign key to the `AccessMode` table.
`InstanceName`	*Type:* text (max 256 characters). Key The name of the WMI class instance used in the source connection for the WMI evidence

InstallerEvidence Table

`InstallerEvidence` lists installer evidence that is used to identify that a particular item of software (defined in the `SoftwareTitle` table) has been installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 358: Database columns for `InstallerEvidence` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key. Generated ID A unique identifier for an installer evidence record.

Database Column	Details
`InstallerEvidenceTypeID`	*Type:* integer. Key Identifies the type of installer evidence. Defaults to `MSI`. Foreign key to the `InstallerEvidenceType` table.
`DisplayName`	*Type:* text (max 256 characters). Key The display name of the software as reported by the installer evidence.
`Version`	*Type:* text (max 72 characters). Key The version of the software as reported by the installer evidence.
`Publisher`	*Type:* text (max 200 characters). Key The publisher of the software as reported by the installer evidence.
`OperatorManageStateID`	*Type:* integer. Key The management responsibility for this information. Foreign key to the `OperatorManageState` table.
`Ignored`	*Type:* boolean Set this field to `True` if the installer evidence is not used for application recognition.
`IsShared`	*Type:* boolean

InstallerEvidenceEx Table

The `InstallerEvidenceEx` table contains additional information on the installer evidence managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 359: Database columns for `InstallerEvidenceEx` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key A unique identifier for an installer evidence record.
`OperatorManageStateID`	*Type:* integer. Nullable The management responsibility for this information. Foreign key to the `OperatorManageState` table.

Database Column	Details
`Ignored`	*Type:* boolean. Nullable Set this field to `True` if the installer evidence is not used for application recognition.

InstallerEvidenceMatchCount Table

`InstallerEvidenceMatchCount` tracks the number of times that each installer evidence (rule) has been detected as installed and recorded in the data source. A separate count is kept for each installer evidence rule, and for each data source.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 360: Database columns for `InstallerEvidenceMatchCount` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key The installer evidence which is being matched. Foreign key to the `InstallerEvidence` table.
`ComplianceConnectionID`	*Type:* integer. Key The data source where the match is occurring. Foreign key to the `ComplianceConnection` table.
`MatchedCount`	*Type:* integer The number of installed installer evidence records in this data source matching this installer evidence rule.
`InstallCount`	*Type:* integer The number of physical application installations recognized in this data source using this installer evidence rule.

InstallerEvidenceType Table

`InstallerEvidenceType` is a static table listing the types of installer evidence that can be used to determine whether an item of software has been installed.

Table 361: Database columns for `InstallerEvidenceType` table

Database Column	Details
`InstallerEvidenceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `InstallerEvidenceType`. Possible values and the corresponding default strings are: • 1 = Any • 2 = Add/Remove Programs • 3 = Software ID Tag • 4 = MSI • 5 = Unknown • 6 = ILMT • 7 = RPM • 8 = OS X App • 9 = LPP • 10 = SDUX • 11 = SUNPKG • 12 = IA • 13 = BEA • 14 = ISMP • 15 = IPS • 16 = ADDM • 17 = OracleEBSModule • 18 = BDNA • 19 = FlexeraID • 20 = DPKG • 21 = App-V • 22 = OUI • 23 = IIM • 24 = DSPMQ • 25 = VMware • 26 = HPUD

Database Column	Details
	• 27 = SaaS • 28 = UniversalApplication
`TypeResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an installer evidence type. Foreign key to the `ComplianceResourceString` table.
`TypeDefaultString`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`ImporterString`	*Type:* text (max 100 characters) The text value provided by adapters when importing installer evidence.

InstdSWAssignmentReasons Table

`InstdSWAssignmentReasons` lists all license assignments attempted for an installation of an application.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 362: Database columns for `InstdSWAssignmentReasons` table

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key The link to the installed software record a license assignment was attempted for. Foreign key to the `InstalledSoftware` table.
`SoftwareLicenseID`	*Type:* integer. Key The link to the license this install has been counted against. Foreign key to the `SoftwareLicense` table.
`ComplianceUserID`	*Type:* integer. Key. Nullable The link to the user this install has been counted against. Foreign key to the `ComplianceUser` table.
`MultiProductLicensePhase`	*Type:* boolean This field is set to `True` when this installation is licensed during the multi-product license phase of license reconcile.
`Order`	*Type:* integer The order this license was attempted to be assigned to this installation.

Database Column	Details
`LicenseAssignment FailureReasonID`	*Type:* integer. Nullable The reason this installation could not be assigned to this license. Foreign key to the `LicenseAssignmentFailureReason` table.
`LAConsReasonID`	*Type:* integer. Nullable The reason this installation consumed entitlements from this license. Foreign key to the `LicenseAssignmentConsumptionReason` table.
`FirstAvailable`	*Type:* boolean Whether this license was the first available to be assigned to this installation regardless of purchases.
`RequestedValue`	*Type:* text (max 256 characters). Nullable The requested value for this installation on this license.
`AvailableValue`	*Type:* text (max 256 characters). Nullable The available value for this installation on this license.
`AddedBySQLPhase`	*Type:* boolean Specifies whether the licence allocation was done by SQL or C# code. This is an internal field that can be used when troubleshooting license assignments.

LicenseAssignmentConsumptionReason Table

`LicenseAssignmentConsumptionReason` holds all the reasons why a license assignment for an installation of an application consumed the entitlements it did.

Table 363: Database columns for `LicenseAssignmentConsumptionReason` table

Database Column	Details
`LAConsReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseAssignmentConsumptionReason`. Possible values and the corresponding default strings are: • 1 = License entitlement or points consumed as expected • 2 = Covered by right of second use • 3 = Allocation of a license entitlement to this device triggers automatic consumption • 4 = Access to the application is counted as consumption for this license type

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a license assignment consumption reason. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the license assignment consumption reason resource string has no translation.

LicenseAssignmentFailureReason Table

`LicenseAssignmentFailureReason` holds all the reasons why a license assignment for an installation of an application could not be made.

Table 364: Database columns for `LicenseAssignmentFailureReason` table

Database Column	Details
`LicenseAssignment FailureReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseAssignmentFailureReason`. Possible values and the corresponding default strings are: • 1 = There are not enough entitlements available on this license to cover this device • 2 = The installation was outside the restricted enterprise group • 3 = Custom tags on the device do not match the custom tags set on the license • 4 = This license type requires allocation, but no license entitlement was allocated to the device or user • 5 = The number of sockets, required to calculate consumption, is missing for this device • 6 = The number of sockets for this device is more than the maximum allowed for this license • 7 = The number of processors, required for calculating consumption from this license type, is missing for this device • 8 = The number of processors for this device is fewer than the minimum allowed for this license • 9 = The number of processors for this device is more than the maximum allowed for this license • 10 = The number of cores, required for calculating consumption for this license type,is missing for this device • 11 = The number of cores (or processors) for this device is more than the maximum allowed for this license • 12 = No licensable Oracle database instance is known for this device • 13 = No consumption calculated, possibly because processor, core, or thread counts are missing from device inventory • 14 = Application is supplementary on this license, and consumption needs either the missing primary product installation, or an allocation • 15 = The license was not assessed, because the device has consumed from a higher priority license • 16 = Access logs are missing the usage date/period needed for a CAL

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a license assignment failure reason. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the license assignment failure reason resource string has no translation.

LicenseBreachReason Table

`LicenseBreachReason` is a static table holding the collection of reasons why a license can be at risk.

Table 365: Database columns for `LicenseBreachReason` table

Database Column	Details
`LicenseBreachReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseBreachReason`. Possible values and the corresponding default strings are: • 1 = Installed Greater Than Purchased • 2 = Child License At Risk • 3 = Install Linked to License has Invalid Sockets • 4 = Software License Does Not Meet Minimums • 5 = Software License Has Expired • 6 = Unlicensed Component Installed • 7 = Peak Consumed Quantity Greater Than Purchased • 8 = Nested License At Risk • 9 = Supplementary Product Exceeds Ratio.
`BreachResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a risk reason. Foreign key to the `ComplianceResourceString` table.
`BreachDefaultValue`	*Type:* text (max 512 characters) The text to display if the reason resource string has no translation.

LicenseDefinitionTitle Table

`LicenseDefinitionTitle` associates software license definitions with their related applications.

Table 366: Database columns for `LicenseDefinitionTitle` table

Database Column	Details
`SoftwareLicenseDefinitionID`	*Type:* integer. Key The license definition. Foreign key to the `SoftwareLicenseDefinition` table.
`SoftwareRecognitionID`	*Type:* text (max 100 characters). Key The encrypted FlexNet Manager Suite factory unique ID for the linked application in the Application Recognition Library.

LicenseDefinitionType Table

`LicenseDefinitionType` is a static table listing supported software license definition types, which are used to distinguish records downloaded from the Product Use Rights Library.

Table 367: Database columns for `LicenseDefinitionType` table

Database Column	Details
`LicenseDefinitionTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a definition type. Possible values (and associated default names) are: • 1 = License • 2 = Product • 3 = Usage Right.
`TypeName`	*Type:* text (max 100 characters). Key Unique internal name for this definition type.

LicenseDefinitionUsageRight Table

`LicenseDefinitionUsageRight` associates software license definitions and Application Recognition Library software applications to recommended usage rights.

Table 368: Database columns for `LicenseDefinitionUsageRight` table

Database Column	Details
`LicenseDefinition FactoryUID`	*Type:* text (max 100 characters). Key The encrypted factory unique ID for a license definition or ARL application.
`UsageRightFactoryUID`	*Type:* text (max 100 characters). Key The encrypted factory unique ID for a usage right template.
`IsPrimary`	*Type:* boolean Is the software application a primary application to the recommended usage rights?
`IsBundle`	*Type:* boolean Is the recommended usage rights a bundle?
`IsRelatedByEdition`	*Type:* boolean Is the recommended usage rights is related to this primary application by the edition?

LicenseMeasurement Table

The `LicenseMeasurement` table is used to store license measurement snapshots.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 369: Database columns for `LicenseMeasurement` table

Database Column	Details
`LicenseMeasurementID`	*Type:* integer. Key. Generated ID A unique identifier for the license measurement.
`MeasurementCode`	*Type:* text (max 128 characters) The unique code for this measurement.
`MeasurementTime`	*Type:* datetime. Key The date and time this measurement was started.
`MeasurementEndTime`	*Type:* datetime. Nullable The date and time this measurement was completed.
`Success`	*Type:* boolean Determines whether the measurement completed successfully.

Database Column	Details
`Description`	*Type:* text (max 50 characters) The description of this measurement.
`IsPartial`	*Type:* boolean Indicate whether this licence run was a partial run or not.

LicenseSimulation Table

A `LicenseSimulation` is made up of an initial scenario, and a cloned version of this scenario. The user can modify the rows in this cloned scenario.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 370: Database columns for `LicenseSimulation` table

Database Column	Details
`LicenseSimulationID`	*Type:* integer. Key. Generated ID Unique ID for the `LicenseSimulation` table.
`LicenseSimulation ScenarioID`	*Type:* integer. Key Foreign key to the `LicenseSimulationScenario` table.
`LastModified`	*Type:* datetime The last time this simulation was modified.
`ComplianceOperatorID`	*Type:* integer. Key The compliance operator responsible for this scenario
`DisplayName`	*Type:* text (max 256 characters). Nullable The name given to this simulation by the owner/operator.
`DisplayRateID`	*Type:* integer. Nullable The rate to be used to display all price values in this simulation. Foreign key to the `CurrencyRate` table. If null, then the user's default can be used.

LicenseSimulationBreachStatus Table

`LicenseSimulationBreachStatus` is a static table listing all of the risk states a license can be in, once it is modelled in a Simulation.

Table 371: Database columns for `LicenseSimulationBreachStatus` table

Database Column	Details
`LicenseSimulationBreachStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseSimulationBreachStatus`. Possible values and the corresponding default strings are: • 1 = Still compliant • 2 = Still at risk • 3 = Now compliant • 4 = Now at risk.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing risk status in a license simulation. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

LicenseSimulationChangeType Table

`LicenseSimulationChangeType` is a static table listing all the types of operations that can be applied as changes to simulation data

Table 372: Database columns for `LicenseSimulationChangeType` table

Database Column	Details
`LicenseSimulationChangeTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseSimulationChangeType`. Possible values and the corresponding default strings are: • 1 = Unchanged • 2 = Added • 3 = Deleted • 4 = Modified • 5 = Moved.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a computer role. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
DefaultValue	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

LicenseSimulationHWDetails Table

LicenseSimulationHWDetails stores a complete snapshot of hardware data for simulations. The LicenseSimulationScenario associated with each record could be an original snapshot of data, or a user modifiable scenario.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 373: Database columns for LicenseSimulationHWDetails table

Database Column	Details
LicenseSimulationHW DetailsID	*Type:* integer. Key. Generated ID A unique identifier for a hardware item that is part of a simulation scenario.
LicenseSimulation ScenarioID	*Type:* integer. Key The simulation scenario this hardware item is part of. Foreign key to the LicenseSimulationScenario table.
Name	*Type:* text (max 256 characters). Nullable The friendly name for this hardware item.
LicenseSimulationRow TypeID	*Type:* integer The type of hardware for this item.
Manufacturer	*Type:* text (max 128 characters). Nullable The manufacturer of this hardware item. Typically applies to a virtualisation server.
ModelNo	*Type:* text (max 128 characters). Nullable The model number of this hardware item. Typically applies to a virtualisation server.
ChassisNumber	*Type:* text (max 128 characters). Nullable The chassis number of this hardware item. Typically applies to a virtualisation server.
SerialNo	*Type:* text (max 100 characters). Nullable The serial number of this hardware item. Typically applies to a virtualisation server or physical machine.

Database Column	Details
`ProcessorType`	*Type:* text (max 256 characters). Nullable The processor type of this hardware item.
`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed of this hardware item.
`PurchaseDate`	*Type:* datetime. Nullable The date this hardware item was purchased on, if it has an associated `Asset`.
`NumSockets`	*Type:* integer. Nullable The number of physical CPU sockets of this hardware item.
`PoolTypeID`	*Type:* integer. Nullable The type of pool technology of this hardware item. Typically applies to resource pools. Foreign key to the `VMPoolType` table.
`VMTypeID`	*Type:* integer. Nullable The type of virtual machine technology of this hardware item. Typically applies to virtual machines. Foreign key to the `VMType` table.
`OperatingSystem`	*Type:* text (max 128 characters). Nullable The operating system running on this hardware item.
`NumProcessors`	*Type:* decimal. Nullable The number of processors available to this hardware item.
`NumCores`	*Type:* decimal. Nullable The number of cores available to this hardware item.
`NumThreads`	*Type:* integer. Nullable The number of threads available to this hardware item.
`MaxNumberOfLogical Processors`	*Type:* decimal. Nullable The configured maximum number of logical processors(ie, threads) for this hardware item, if applicable.
`ParentLicense SimulationHWDetailsID`	*Type:* integer. Key. Nullable The parent hardware item of this item.
`HostLicenseSimulationH WDetailsID`	*Type:* integer. Nullable The host hardware item of this item.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The actual computer record for this hardware item. Foreign key to the `ComplianceComputer` table.

Database Column	Details
`VMLayerID`	*Type:* integer. Key. Nullable Internal unique identifier used when populating hardware items to create a new simulation.
`LicenseSimulation ChangeTypeID`	*Type:* integer Tracks the state of the hardware item, as it gets modified by the simulation user. Foreign key to the `LicenseSimulationChangeType` table.
`ClusterID`	*Type:* integer. Nullable The hardware cluster to which this computer belongs, if any. Foreign key to the `Cluster` table.
`AffinityEnabled`	*Type:* boolean Whether this VM is locked to its current host computer.
`CoreAffinity`	*Type:* text (max 256 characters). Nullable Contains the Core Affinity value for virtual machine
`CloudServiceProviderID`	*Type:* integer. Key. Nullable The cloud service provider for the virtual machine

LicenseSimulationLicenseDetails Table

`LicenseSimulationLicenseDetails` stores properties associated with each license included in a simulation scenario. The `LicenseSimulationScenario` associated with each record could be an original snapshot of data, or a user modifiable scenario.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 374: Database columns for `LicenseSimulationLicenseDetails` table

Database Column	Details
`LicenseSimulation LicenseDetailsID`	*Type:* integer. Key. Generated ID A unique identifier for a license item that is part of a simulation scenario.
`OriginalLicense SimulationLicense DetailsID`	*Type:* integer. Key. Nullable The original version of this license, that has not been modified by a simulation user.

Database Column	Details
LicenseSimulation ScenarioID	*Type:* integer. Key The simulation scenario this softare license is part of. Foreign key to the LicenseSimulationScenario table.
SoftwareLicenseID	*Type:* integer. Key The software license for this simulation license. Foreign key to the SoftwareLicense table.
UnitPrice	*Type:* currency. Nullable The unit price associated with this license.
UnitPriceRateID	*Type:* integer. Nullable The rate for the total value. Foreign key to the CurrencyRate table.
LicenseSimulation ChangeTypeID	*Type:* integer Tracks the state of the softare license, as it gets modified by the simulation user. Foreign key to the LicenseSimulationChangeType table.

LicenseSimulationResults Table

LicenseSimulationResults stores points consumed by each item in a simulation scenario against each license included in the simulation scenario.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 375: Database columns for LicenseSimulationResults table

Database Column	Details
LicenseSimulationHW DetailsID	*Type:* integer. Key The hardware item for this license simulation result. Foreign key to the LicenseSimulationHWDetails table.
LicenseSimulation ScenarioID	*Type:* integer. Key The scenario for this license simulation result. Foreign key to the LicenseSimulationScenario table.
SoftwareLicenseID	*Type:* integer. Key The software license for this license simulation result. Foreign key to the SoftwareLicense table.

Database Column	Details
`InstalledCount`	*Type:* decimal The number of processors/cores on which a software title licensed by the license is installed.
`UsedCount`	*Type:* decimal The number of processors/cores on which a software title licensed by the license is used.
`CapacityCount`	*Type:* decimal The number of processors/cores that apply to a software license under full capacity counting rules.
`IsCapped`	*Type:* boolean Does this layer implement hard partitioning for this license?
`PointsFactor`	*Type:* decimal The number of points consumed per processor/core on this computer for this license.
`PointsConsumed`	*Type:* decimal. Nullable The number of processor/core points required to cover the above InstalledCount.
`PointsUsed`	*Type:* decimal. Nullable The number of processor/core points required to cover the above UsedCount.
`CapacityPointsConsumed`	*Type:* decimal. Nullable The number of processor/core points required to cover the above CapacityCount.
`PointsCalculated`	*Type:* decimal The number of calculated points this installation consumes.
`Overridden`	*Type:* boolean Is this simulation result derived from an overridden consumption via allocation.

LicenseSimulationRowType Table

`LicenseSimulationRowType` is a static table listing all types of rows that can be displayed in the Simulation UI. Entries in the `LicenseSimulationSWDetails` table are assumed to be type 4 (Software installation)

Table 376: Database columns for `LicenseSimulationRowType` table

Database Column	Details
`LicenseSimulationRowTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseSimulationRowType`. Possible values and the corresponding default strings are: • 1 = Host • 2 = Shared pool • 3 = Virtual Machine • 4 = Software installation • 5 = Physical machine.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the type of a row in a license simulation. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

LicenseSimulationSWDetails Table

`LicenseSimulationSWDetails` stores a complete snapshot of software data for simulations. The `LicenseSimulationHWDetails` record associated with each `LicenseSimulationSWDetails` record could be part of an original snapshot of data, or a user modifiable scenario.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 377: Database columns for `LicenseSimulationSWDetails` table

Database Column	Details
`LicenseSimulationSWDetailsID`	*Type:* integer. Key. Generated ID A unique identifier for a software installation that is part of a simulation scenario.
`LicenseSimulationHWDetailsID`	*Type:* integer. Key The hardware item that this software title is installed on. Foreign key to the `LicenseSimulationHWDetails` table.

Database Column	Details
LicenseSimulation ScenarioID	*Type:* integer. Key The simulation scenario this softare installation is part of. Foreign key to the LicenseSimulationScenario table.
OriginalLicense SimulationSWDetailsID	*Type:* integer. Key. Nullable The original version of this software installation, that has not been modified by a simulation user.
Name	*Type:* text (max 512 characters) The friendly name of this software installation.
SoftwareTitleID	*Type:* integer. Key The software title that is installed here. Foreign key to the SoftwareTitle table.
SoftwareLicenseID	*Type:* integer. Key The software license that this install is assigned to. Foreign key to the SoftwareLicense table.
LicenseSimulation ChangeTypeID	*Type:* integer Tracks the state of the softare installation, as it gets modified by the simulation user. Foreign key to the LicenseSimulationChangeType table.
IsUsed	*Type:* boolean Set this field to True if the software title is installed according to usage thresholds in the SoftwareTitle table.

LicenseSimulationScenario Table

A LicenseSimulationScenario is a set of hardware and software inventory details that are recorded at a particular point in time. A scenario can be modifed by the user for the purposes of simulation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 378: Database columns for LicenseSimulationScenario table

Database Column	Details
LicenseSimulation ScenarioID	*Type:* integer. Key. Generated ID Unique ID for the LicenseSimulationScenario table.
OriginalLicense SimulationScenarioID	*Type:* integer. Key. Nullable The original (unmodified) scenario that a user-modifiable scenario was based on

LicenseStatus Table

`LicenseStatus` is a static table storing the collection of possible license states.

Table 379: Database columns for `LicenseStatus` table

Database Column	Details
`LicenseStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `LicenseStatus`. Possible values and the corresponding default strings are: • 1 = Active • 2 = Retired • 3 = In Stock • 4 = Purchased • 5 = Received.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a license status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

NewFileEvidence Table

`NewFileEvidence` identifies files used as evidence that an application (defined in the `SoftwareTitle` table) has been installed on a computer. File evidence may have wildcards, so each record in this table should be considered a rule, which one or more physical files on a computer may match.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 380: Database columns for `NewFileEvidence` table

Database Column	Details
`FileEvidenceID`	*Type:* integer. Key. Generated ID A unique identifier for a file evidence record.
`FileEvidenceFileID`	*Type:* integer. Key The file name. Foreign key to the `FileEvidenceFile` table.

Database Column	Details
`FileEvidenceCompanyID`	*Type:* integer. Key The company publishing the software. Foreign key to the `FileEvidenceCompany` table.
`FileEvidencePathID`	*Type:* integer. Key. Nullable The file path where the file was located. Foreign key to the `FileEvidencePath` table.
`FileEvidenceLanguageID`	*Type:* integer. Key. Nullable The language identified in the file header. Foreign key to the `FileEvidenceLanguage` table.
`FileVersion`	*Type:* text (max 100 characters). Key The version number of the file used as evidence of software installation.
`ProductName`	*Type:* text (max 200 characters). Nullable The product name in the file header.
`ProductVersion`	*Type:* text (max 200 characters). Nullable The product version number in the file header.
`Description`	*Type:* text (max 200 characters). Key The description in the file header.
`FileSize`	*Type:* integer. Key. Nullable The size of the file.
`OperatorManageStateID`	*Type:* integer. Key The management responsibility for this information. Foreign key to the `OperatorManageState` table.
`Ignored`	*Type:* boolean Set this field to `True` to indicate that this file evidence is ignored for application recognition.
`IsShared`	*Type:* boolean

OracleLegacyLicenseType Table

`OracleLegacyLicenseType` lists some of the legacy Oracle license types.

Table 381: Database columns for `OracleLegacyLicenseType` table

Database Column	Details
`OracleLegacyLicenseTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `OracleLegacyLicenseType`. Possible values and the corresponding default strings are: • 1 = Named User • 2 = Named User Network license • 3 = Named User Single Server • 4 = Named User Multi Server • 5 = Concurrent Device • 6 = Concurrent Device Network License • 7 = UPU • 8 = Developer • 9 = Developer Network License • 10 = Concurrent User • 11 = Concurrent User Network License • 12 = Application Specific Full User Licensing • 13 = Embedded Software License • 14 = Site.
`OracleLegacyLicenseTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an Oracle legacy license type. Foreign key to the `ComplianceResourceString` table.
`OracleLegacyLicenseTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

PODetailProcess Table

`PODetailProcess` records the processing steps taken when applying upgrades to software installations. The newly-purchased upgrade license is linked here to the original license being upgraded.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 382: Database columns for `PODetailProcess` table

Database Column	Details
`PurchaseOrderDetailID`	*Type:* integer. Key The purchase order line that defines this upgrade. Foreign key to the `PurchaseOrderDetail` table.
`FromSoftwareLicenseID`	*Type:* integer. Key. Nullable The original software license to which an upgrade is being applied. Foreign key to the `SoftwareLicense` table.
`ToSoftwareLicenseID`	*Type:* integer. Key. Nullable The upgrade license referenced in the PO line and permitting the installation of the software upgrade. Foreign key to the `SoftwareLicense` table.
`ProcessActionID`	*Type:* integer The processing action taken with respect to this upgrade. Defaults to `Defer`. Foreign key to the `ProcessAction` table.
`ProcessStateID`	*Type:* integer. Key The resulting process state of the upgrade. Foreign key to the `ProcessState` table.
`CreationDate`	*Type:* datetime The date this record was created.

PVUSoftwareLicenseProcessorData Table

This serves as an intermediate table during PVU reconciliation process to store the number of processors (or cores) on which licensed software is installed and used for each computer, and the calculated points.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 383: Database columns for `PVUSoftwareLicenseProcessorData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key. Nullable The host computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.

Database Column	Details
PVUVirtualMachineLayerID	*Type:* integer. Key. Nullable The virtual machine layer under examination. Foreign key to the ReconcileVirtualMachineLayer table.
IsHost	*Type:* boolean. Key Does this refer to the top layer for this host?
IsCapped	*Type:* boolean Does this layer implement hard partitioning for this license?
InstalledCount	*Type:* decimal The number of processors/cores on which a software title licensed by the license is installed.
UsedCount	*Type:* decimal The number of processors/cores on which a software title licensed by the license is used.
CapacityCount	*Type:* decimal The number of processors/cores that apply to a software title licensed by the license under full capacity counting rules.
PointsFactor	*Type:* decimal The number of points consumed per processor/core on this computer.
InstalledPoints	*Type:* integer The number of processor/core points required to cover the above InstalledCount.
UsedPoints	*Type:* integer The number of processor/core points required to cover the above UsedCount.
CapacityPoints	*Type:* integer The number of processor/core points required to cover the above CapacityCount.
CalculatedConsumption	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.
Overridden	*Type:* boolean Whether this consumption value was the result of an override.

PVUVirtualMachineLayer Table

This serves as an intermediate table during PVU reconciliation process to store virtual machines, pools and hosts in a generalized tree structure.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 384: Database columns for `PVUVirtualMachineLayer` table

Database Column	Details
`PVUVirtualMachineLayerID`	*Type:* integer. Key A unique identifier for a hardware item that is part of a simulation scenario.
`FNMPComputerUID`	*Type:* unique identifier. Key. Nullable The unique identifier generated for the computer from the IM database. This property should only be populated by the ManageSoft inventory adapter.
`ParentPVUVirtual MachineLayerID`	*Type:* integer. Key. Nullable The parent hardware item of this item. Foreign key to the `PVUVirtualMachineLayer` table.
`HostPVUVirtualMachine LayerID`	*Type:* integer. Key. Nullable The host hardware item of this item. Foreign key to the `PVUVirtualMachineLayer` table.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The actual computer record for this hardware item. Foreign key to the `ComplianceComputer` table.
`HostComplianceComputerID`	*Type:* integer. Key. Nullable The actual host computer record for this hardware item. Foreign key to the `ComplianceComputer` table.
`ExternalID`	*Type:* integer. Key. Nullable The identifier used in the source connection for the end-user.
`PoolTypeID`	*Type:* integer. Nullable The type of pool technology of this hardware item. Typically applies to resource pools. Foreign key to the `VMPoolType` table.
`VMTypeID`	*Type:* integer. Nullable The type of virtual machine technology of this hardware item. Typically applies to virtual machines. Foreign key to the `VMType` table.
`VMPoolID`	*Type:* integer. Nullable The resource pool that the virtual machine belongs to. Foreign key to the `VMPool` table.
`VirtualMachineID`	*Type:* integer. Nullable The identifier of this virtual machine. Foreign key to the `VirtualMachine` table.

Database Column	Details
`ParentVMPoolID`	*Type:* integer. Nullable The identifier of the parent VM pool of this pool. Foreign key to the `VMPool` table.
`ClusterID`	*Type:* integer. Nullable The hardware cluster to which this computer belongs, if any. Foreign key to the `Cluster` table.
`Name`	*Type:* text (max 256 characters). Nullable The friendly name for this hardware item.
`LicenseSimulationRowTypeID`	*Type:* integer The type of hardware for this item.
`Manufacturer`	*Type:* text (max 128 characters). Nullable The manufacturer of this hardware item. Typically applies to a virtualisation server.
`ModelNo`	*Type:* text (max 128 characters). Nullable The model number of this hardware item. Typically applies to a virtualisation server.
`ChassisNumber`	*Type:* text (max 128 characters). Nullable The chassis number of this hardware item. Typically applies to a virtualisation server.
`SerialNo`	*Type:* text (max 100 characters). Nullable The serial number of this hardware item. Typically applies to a virtualisation server or physical machine.
`ProcessorType`	*Type:* text (max 256 characters). Nullable The processor type of this hardware item.
`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed of this hardware item.
`PartialNumberOfProcessors`	*Type:* decimal. Nullable The fractional processor count available to this layer.
`PurchaseDate`	*Type:* datetime. Nullable The date this hardware item was purchased on, if it has an associated `Asset`.
`NumSockets`	*Type:* integer. Nullable The number of physical CPU sockets of this hardware item.
`OperatingSystem`	*Type:* text (max 128 characters). Nullable The operating system running on this hardware item.

Database Column	Details
NumProcessors	*Type:* decimal. Nullable The number of processors available to this hardware item.
NumCores	*Type:* decimal. Nullable The number of cores available to this hardware item.
NumThreads	*Type:* integer. Nullable The number of threads available to this hardware item.
MaxNumberOfLogical Processors	*Type:* decimal. Nullable The configured maximum number of logical processors(ie, threads) for this hardware item, if applicable.
AffinityEnabled	*Type:* boolean Whether this VM is locked to its current host computer.
CoreAffinity	*Type:* text (max 256 characters). Nullable Contains the Core Affinity value for virtual machine
IsFlexNetInventory	*Type:* boolean. Nullable Whether this VM inventory was obtained from the FlexNet Manager agent.

PeriodType Table

PeriodType is a static table holding a collection of supported time periods to indicate the frequency of license charge-backs.

Table 385: Database columns for PeriodType table

Database Column	Details
PeriodTypeID	*Type:* integer. Key. Generated ID A unique identifier for each PeriodType. Possible values and the corresponding default strings are: • 1 = None • 2 = Weekly • 3 = Monthly • 4 = Quarterly • 5 = Yearly • 6 = Lump Sum.

Database Column	Details
`PeriodTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a period type. Foreign key to the `ComplianceResourceString` table.
`PeriodTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ProcessAction Table

`ProcessAction` is a static table holding a collection of possible actions that can be applied while processing a SKU, with a special focus on processing software license upgrades.

Table 386: Database columns for `ProcessAction` table

Database Column	Details
`ProcessActionID`	*Type:* integer. Key. Generated ID A unique identifier for each `ProcessAction`. Possible values and the corresponding default strings are: • 1 = Link to existing license • 2 = Create new from SKU • 3 = Create new from PO line • 4 = Upgrade license: Link to existing license and upgrade from existing license • 5 = Upgrade license: Link to existing license and select upgrade from license • 6 = Upgrade license: Link to new license created from SKU and select upgrade from license • 7 = Remove from list • 8 = Upgrade license: Link to new license created from PO line and select upgrade from license • 9 = Create new from SKU with fixed maintenance • 10 = Create new from PO line with fixed maintenance • 11 = Create new from SKU with unlimited maintenance • 12 = Create new from PO line with unlimited maintenance • 13 = Create new from SKU with maintenance from contract • 14 = Create new from PO line with maintenance from contract • 15 = Apply contract maintenance to an existing license • 16 = Apply fixed maintenance to an existing license • 17 = Apply unlimited maintenance to an existing license • 18 = Apply contract maintenance to an existing license by SKU • 19 = Apply fixed maintenance to an existing license by SKU • 20 = Apply unlimited maintenance to an existing license by SKU • 21 = Apply contract maintenance to a non-existent license for SKU • 22 = Apply fixed maintenance to a non-existent license for SKU • 23 = Apply unlimited maintenance to a non-existent license for SKU • 24 = Upgrade license: Link to existing license and upgrade from existing license with contract maintenance

Database Column	Details
	• 25 = Upgrade license: Link to existing license and upgrade from existing license with fixed maintenance • 26 = Upgrade license: Link to existing license and upgrade from existing license with unlimited maintenance • 27 = Upgrade license: Link to existing license and select upgrade from license with contract maintenance • 28 = Upgrade license: Link to existing license and select upgrade from license with fixed maintenance • 29 = Upgrade license: Link to existing license and select upgrade from license with unlimited maintenance • 30 = Upgrade license: Link to new license created from SKU and select upgrade from license with contract maintenance • 31 = Upgrade license: Link to new license created from SKU and select upgrade from license with fixed maintenance • 32 = Upgrade license: Link to new license created from SKU and select upgrade from license with unlimited maintenance • 33 = Upgrade license: Link to new license created from PO line and select upgrade from license with contract maintenance • 34 = Upgrade license: Link to new license created from PO line and select upgrade from license with fixed maintenance • 35 = Upgrade license: Link to new license created from PO line and select upgrade from license with unlimited maintenance • 36 = Apply maintenance to a contract • 37 = No recommendation • 38 = Create a new license • 39 = Create a new license with a maintenance contract • 40 = Create a new license with fixed maintenance • 41 = Create a new license with unlimited maintenance • 42 = Add entitlements to a license • 43 = Add entitlements to a license with a maintenance contract • 44 = Add entitlements to a license with fixed maintenance • 45 = Add entitlements to a license with unlimited maintenance

Database Column	Details
	• 46 = Upgrade to a new license • 47 = Upgrade to a new license with a maintenance contract • 48 = Upgrade to a new license with fixed maintenance • 49 = Upgrade to a new license with unlimited maintenance • 50 = Upgrade to an existing license • 51 = Upgrade to an existing license with a maintenance contract • 52 = Upgrade to an existing license with fixed maintenance • 53 = Upgrade to an existing license with unlimited maintenance • 54 = Apply maintenance from a contract to an existing license • 55 = Apply fixed maintenance to an existing license • 56 = Apply unlimited maintenance to an existing license
`ProcessActionResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an action. Foreign key to the `ComplianceResourceString` table.
`ProcessActionDefaultValue`	*Type:* text (max 256 characters) The text to display if the action resource string has no translation.

ProcessState Table

`ProcessState` is a static table holding the collection of processing states that a purchase order line containing a SKU can be left in.

Table 387: Database columns for `ProcessState` table

Database Column	Details
`ProcessStateID`	*Type:* integer. Key. Generated ID A unique identifier for each `ProcessState`. Possible values and the corresponding default strings are: • 1 = Unprocessed • 2 = Processed • 3 = Deferred • 4 = Discarded.

Database Column	Details
ProcessStateResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a processing state. Foreign key to the ComplianceResourceString table.
ProcessStateDefaultValue	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

ReconcileAccdSWDevAsgnReasons Table

ReconcileAccdSWDevAsgnReasons lists all license assignments attempted for client access of an application during an execution of license reconcile for the device.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 388: Database columns for ReconcileAccdSWDevAsgnReasons table

Database Column	Details
AccessingDeviceID	*Type:* integer. Key. Nullable The accessing device under examination. Foreign key to the AccessingDevice table.
ComplianceComputerID	*Type:* integer. Key. Nullable The compliance computer under examination. Foreign key to the ComplianceComputer table.
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
MultiProductLicensePhase	*Type:* boolean This field is set to True when this installation is licensed during the multi-product license phase of license reconcile.
Order	*Type:* integer The order this license was attempted to be assigned to this installation.
LAConsReasonID	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
LicenseAssignment FailureReasonID	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.

Database Column	Details
AddedBySQLPhase	*Type:* boolean Specifies whether the licence allocation was done by SQL or C# code. This is an internal field that can be used when troubleshooting license assignments.

ReconcileAccdSWUsrAsgnReasons Table

ReconcileAccdSWUsrAsgnReasons lists all license assignments attempted for client access of an application during an execution of license reconcile for the user.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 389: Database columns for ReconcileAccdSWUsrAsgnReasons table

Database Column	Details
AccessingUserID	*Type:* integer. Key. Nullable The accessing device under examination. Foreign key to the AccessingDevice table.
ComplianceUserID	*Type:* integer. Key The compliance computer under examination. Foreign key to the ComplianceComputer table.
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
MultiProductLicensePhase	*Type:* boolean This field is set to True when this installation is licensed during the multi-product license phase of license reconcile.
Order	*Type:* integer The order this license was attempted to be assigned to this installation.
LAConsReasonID	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
LicenseAssignment FailureReasonID	*Type:* integer. Nullable How many of the points consumed are for installations actually being used.
AddedBySQLPhase	*Type:* boolean Specifies whether the licence allocation was done by SQL or C# code. This is an internal field that can be used when troubleshooting license assignments.

ReconcileAccessedSoftwareData Table

A list of all the accesses of an application, or item of software (as defined in the `SoftwareTitle` table).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 390: Database columns for `ReconcileAccessedSoftwareData` table

Database Column	Details
`AccessedSoftwareID`	*Type:* big integer. Key A unique identifier for an accessed software record.
`ServerComputerID`	*Type:* integer. Key The server computer on which the software is available. Foreign key to the `ComplianceComputer` table.
`AccessingUserID`	*Type:* integer. Key. Nullable The user who accessed the software. Foreign key to the `AccessingUser` table.
`ComplianceUserID`	*Type:* integer. Nullable The compliance user who accessed the software. Foreign key to the `ComplianceUser` table.
`AccessingDeviceID`	*Type:* integer. Key. Nullable The device from which the software is accessed. Foreign key to the `AccessingDevice` table.
`ComplianceComputerID`	*Type:* integer. Nullable The compliance computer from which the software is accessed. Foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key The software that is accessed. Foreign key to the `SoftwareTitle` table.
`IsUsed`	*Type:* boolean Set this field to `True` if the software title is accessed according to usage thresholds in the `SoftwareTitle` table.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable The link to the license this access has been counted against. Foreign key to the `SoftwareLicense` table.

Database Column	Details
SoftwareLicenseAllocationID	*Type:* integer. Key. Nullable The link to the license allocation this access has consumed. Foreign key to the SoftwareLicenseAllocation table.
IsLicensed	*Type:* boolean Set this field to True when this access is licensed.
PointsUsed	*Type:* integer. Nullable The number of this accesses consumed on license.
LastUsedDate	*Type:* datetime. Nullable The last used date of the application by client.
PointsCalculated	*Type:* integer The number of calculated points this installation consumes.

ReconcileInstalledSoftwareData Table

A list of all the installations of an application, or item of software (as defined in the SoftwareTitle table).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 391: Database columns for ReconcileInstalledSoftwareData table

Database Column	Details
InstalledSoftwareID	*Type:* integer. Key. Nullable A unique identifier for an installed software record. In case of allcation without installation, this would be negetive. For temporal installation, this would be NULL.
ComplianceComputerID	*Type:* integer. Key The computer on which the software is installed. Foreign key to the ComplianceComputer table.
SoftwareTitleID	*Type:* integer. Key The software that is installed. Foreign key to the SoftwareTitle table.
IsUsed	*Type:* boolean. Key Set this field to True if the software title is installed according to usage thresholds in the SoftwareTitle table.

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key. Nullable The link to the license this install has been counted against. Foreign key to the SoftwareLicense table.
SoftwareLicense AllocationID	*Type:* integer. Key. Nullable The link to the license allocation this installation has consumed. Foreign key to the SoftwareLicenseAllocation table.
IsLicensed	*Type:* boolean Set this field to True when this installation is licensed.
PointsUsed	*Type:* integer. Nullable The number of points this installation consumes on a points-based license.
RawPointsUsed	*Type:* integer. Nullable The number of points this installation consumes on a points-based license before exemptions are considered.
AccessModeID	*Type:* integer. Key The access mode that indicates why this computer was associated with this software title.
LastUsedDate	*Type:* datetime. Nullable The date of the installed software was last used.
PointsCalculated	*Type:* integer The number of calculated points this installation consumes.

ReconcileInstalledSoftwareUsageData Table

This is a staging table for InstalledSoftwareUsage that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 392: Database columns for ReconcileInstalledSoftwareUsageData table

Database Column	Details
ComplianceUserID	*Type:* integer. Key. Nullable The end-user using the application. Foreign key to the ComplianceUser table.

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Nullable The license that covers this installation. Foreign key to the `SoftwareLicense` table.
`SoftwareLicense AllocationID`	*Type:* integer. Key. Nullable A link to any individual allocation that this installation consumes. Foreign key to the `SoftwareLicenseAllocation` table.
`IsLicensed`	*Type:* boolean Set this field to `True` if this usage is licensed.
`UsageSessions`	*Type:* integer The number of sessions for (or times that the application was used by) this end-user on this computer.
`UsageActiveTime`	*Type:* integer The amount of time this application was in active use (in the foreground) for this end-user on this computer.
`ComplianceComputerID`	*Type:* integer. Key The application. Foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key The application. Foreign key to the `SoftwareTitle` table.
`LastUsedDate`	*Type:* datetime. Nullable The last used date of the application.
`AccessModeID`	*Type:* integer. Key The date that the installed software was last used.

ReconcileInstdSWAssignmentReasons Table

`ReconcileInstdSWAssignReasons` lists all license assignments attempted for an installation of an application during an execution of license reconcile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 393: Database columns for `ReconcileInstdSWAssignmentReasons` table

Database Column	Details
`InstalledSoftwareID`	*Type:* integer. Key The link to the installed software record a license assignment was attempted for. Foreign key to the `InstalledSoftware` table.
`SoftwareLicenseID`	*Type:* integer. Key The link to the license this install has been counted against. Foreign key to the `SoftwareLicense` table.
`ComplianceUserID`	*Type:* integer. Key. Nullable The link to the user this install has been counted against. Foreign key to the `ComplianceUser` table.
`MultiProductLicensePhase`	*Type:* boolean This field is set to `True` when this installation is licensed during the multi-product license phase of license reconcile.
`Order`	*Type:* integer The order this license was attempted to be assigned to this installation.
`LicenseAssignment FailureReasonID`	*Type:* integer. Nullable The reason this installation could not be assigned to this license. Foreign key to the `LicenseAssignmentFailureReason` table.
`LAConsReasonID`	*Type:* integer. Nullable The reason this installation consumed entitlements from this license. Foreign key to the `LicenseAssignmentConsumptionReason` table.
`FirstAvailable`	*Type:* boolean Whether this license was the first available to be assigned to this installation regardless of purchases.
`RequestedValue`	*Type:* text (max 256 characters). Nullable The requested value for this installation on this license.
`AvailableValue`	*Type:* text (max 256 characters). Nullable The available value for this installation on this license.
`AddedBySQLPhase`	*Type:* boolean Specifies whether the license allocation was done by SQL or C# code. This is an internal field that can be used when troubleshooting license assignments.

ReconcileInterestingBundleAccessComputer Table

A list of all computers with bundlable accesses for licenses that are interesting to the current execution of license reconcile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 394: Database columns for `ReconcileInterestingBundleAccessComputer` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The unique identifier for a bundle software license that is interesting to an execution of reconcile.
`AccessingDeviceID`	*Type:* integer. Key. Nullable The unique identifier for a accessing device that could consume a bundle software license that is interesting to an execution of reconcile.
`AccessingUserID`	*Type:* integer. Key. Nullable The unique identifier of the accessing user that could consume a bundle software license that is interesting to an execution of reconcile.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The unique identifier for a computer that could consume a bundle software license that is interesting to an execution of reconcile.
`ComplianceUserID`	*Type:* integer. Key. Nullable The unique identifier of the primary user for a computer that could consume a bundle software license that is interesting to an execution of reconcile.
`NumProducts`	*Type:* integer The number of products covered by this license that are accessed on this computer.

ReconcileInterestingBundleInstallComputer Table

A list of all computers with bundlable installs for licenses that are interesting to the current execution of license reconcile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 395: Database columns for `ReconcileInterestingBundleInstallComputer` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The unique identifier for a bundle software license that is interesting to an execution of reconcile.
`ComplianceComputerID`	*Type:* integer. Key The unique identifier for a computer that could consume a bundle software license that is interesting to an execution of reconcile.
`ComplianceUserID`	*Type:* integer. Key. Nullable The unique identifier of the primary user for a computer that could consume a bundle software license that is interesting to an execution of reconcile.
`NumProducts`	*Type:* integer The number of products covered by this license that are installed on this computer.

ReconcileInterestingLicenses Table

A list of all licenses that are interesting to the current execution of license reconcile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 396: Database columns for `ReconcileInterestingLicenses` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The unique identifier for a software license that is interesting to an execution of reconcile.

ReconcileInterestingTitles Table

A list of all titles that are interesting to the current execution of license reconcile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 397: Database columns for `ReconcileInterestingTitles` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The unique identifier for a software title that is interesting to an execution of reconcile.

ReconcileSoftwareAccessDeviceLicensePointsConsumedData Table

This is a staging table for `SoftwareAccessDeviceLicensePointsConsumed` that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 398: Database columns for `ReconcileSoftwareAccessDeviceLicensePointsConsumedData` table

Database Column	Details
`AccessingDeviceID`	*Type:* integer. Key. Nullable The accessing device under examination. Foreign key to the `AccessingDevice` table.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The compliance computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations actually being used.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

ReconcileSoftwareAccessUserLicensePointsConsumedData Table

This is a staging table for `SoftwareAccessUserLicensePointsConsumed` that is used during license reconciliation

process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 399: Database columns for `ReconcileSoftwareAccessUserLicensePointsConsumedData` table

Database Column	Details
`AccessingUserID`	*Type:* integer. Key. Nullable The accessing user under examination. Foreign key to the `AccessingUser` table.
`ComplianceUserID`	*Type:* integer. Key. Nullable The Compliance user under examination. Foreign key to the `ComplianceUser` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations actually being used.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

ReconcileSoftwareLicenseComputerProblem Table

`ReconcileSoftwareLicenseComputerProblem` is a license reconciliation staging table for the `SoftwareLicenseComputerProblemData` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 400: Database columns for `ReconcileSoftwareLicenseComputerProblem` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The software license. Foreign key to the `SoftwareLicense` table.
`ComplianceComputerID`	*Type:* integer. Key The computer consuming license entitlements. Foreign key to the `ComplianceComputer` table.

Database Column	Details
`SoftwareLicenseComputerProblemTypeID`	*Type:* integer The type of problem this computer's inventory causes for a given license. For example, core-based licenses require accurate inventory of processor core counts to determine ther compliance status. Foreign key to the `SoftwareLicenseComputerProblemType` table.

ReconcileSoftwareLicenseCoresConsumedData Table

This is a staging table for `SoftwareLicenseCoresConsumedData` that stores values calculated by license reconciliation. The main table is populated at the end of license reconciliation by a single bulk update.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 401: Database columns for `ReconcileSoftwareLicenseCoresConsumedData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`CoresConsumed`	*Type:* integer The number of cores that have contributed to license point consumption for the license on the computer.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

ReconcileSoftwareLicenseGroupPointsConsumedData Table

This serves as a staging table for `SoftwareLicenseGroupPointsConsumed` during reconciliation process.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 402: Database columns for `ReconcileSoftwareLicenseGroupPointsConsumedData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The license that owns the pre-calculated totals for a group. Foreign key to the `SoftwareLicense` table.
`GroupTypeID`	*Type:* integer. Key Type of the group(Location, Cost center, etc)
`GroupExID`	*Type:* text (max 128 characters). Key. Nullable The group where the local and rolledup values are calculated. Foreign key to the `GroupEx` table.
`RolledUpNumberConsumed`	*Type:* integer The sum of points consumed of the current group and of all its child groups.
`LocalNumberConsumed`	*Type:* integer The sum of points consumed of the current group
`RolledUpNumberUsed`	*Type:* integer The sum of used points f the current group and of all its child groups.
`LocalNumberUsed`	*Type:* integer The sum of used points of the current group
`RolledUpNumberPurchased`	*Type:* integer The rolled up purchase counts of the license.
`LocalNumberPurchased`	*Type:* integer The local purchase counts of the license
`RolledUpNumberCalculated`	*Type:* integer The sum of points calculated for the current group and of all its child groups.
`LocalNumberCalculated`	*Type:* integer The sum of points calculated for the current group.

ReconcileSoftwareLicenseILMTPointsConsumedData Table

This is a staging table for `SoftwareLicenseILMTPointsConsumed` that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 403: Database columns for `ReconcileSoftwareLicenseILMTPointsConsumedData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`CoreCount`	*Type:* integer The number of licensable cores for the license on the computer.
`PVUCount`	*Type:* integer The number of PVU counts consumed for the license on the computer.
`PeakPVUCount`	*Type:* integer The number of PVU counts consumed for the license on the computer at the time where the peak for this license occurred.
`ProductCount`	*Type:* integer The number of products that are consuming same license.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

ReconcileSoftwareLicensePointsConsumedData Table

This is a staging table for `SoftwareLicensePointsConsumed` that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 404: Database columns for `ReconcileSoftwareLicensePointsConsumedData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`LicensesConsumed`	*Type:* integer The number of entitlements (or points) consumed for the license on the computer.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations actually being used.

ReconcileSoftwareLicensePointsConsumedReason Table

This is a staging table for `SoftwareLicensePointsConsumedReasonData` that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 405: Database columns for `ReconcileSoftwareLicensePointsConsumedReason` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.

Database Column	Details
ReasonTypeID	*Type:* integer. Key The reason for the points to be consumed here. Foreign key to the SoftwareLicensePointsConsumedReasonType table.

ReconcileSoftwareLicenseProcessorData Table

This serves as an intermediate table during reconciliation process to store the number of processors (or cores) on which licensed software is installed and used for each computer, and the calculated points.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 406: Database columns for ReconcileSoftwareLicenseProcessorData table

Database Column	Details
ComplianceComputerID	*Type:* integer. Key The host computer under examination. Foreign key to the ComplianceComputer table.
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
VMLayerID	*Type:* integer. Key. Nullable The virtual machine layer under examination. Foreign key to the ReconcileVirtualMachineLayer table.
IsHost	*Type:* boolean. Key Does this refer to the top layer for this host?
IsCapped	*Type:* boolean Does this layer implement hard partitioning for this license?
InstalledCount	*Type:* decimal The number of processors/cores on which a software title licensed by the license is installed.
UsedCount	*Type:* decimal The number of processors/cores on which a software title licensed by the license is used.
CapacityCount	*Type:* decimal The number of processors/cores that apply to a software title licensed by the license under full capacity counting rules.

Database Column	Details
`PointsFactor`	*Type:* decimal The number of points consumed per processor/core on this computer.
`InstalledPoints`	*Type:* integer The number of processor/core points required to cover the above InstalledCount.
`UsedPoints`	*Type:* integer The number of processor/core points required to cover the above UsedCount.
`CapacityPoints`	*Type:* integer The number of processor/core points required to cover the above CapacityCount.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.
`Overridden`	*Type:* boolean Whether this consumption value was the result of an override.

ReconcileSoftwareLicenseSecondUseMappingData Table

This is a staging table for `SoftwareLicenseSecondUseMapping` that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 407: Database columns for `ReconcileSoftwareLicenseSecondUseMappingData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The license conferring the right of second use. Foreign key to the `SoftwareLicense` table.
`DesktopComputerID`	*Type:* integer. Key The desktop or primary computer on which the related software in installed. Foreign key to the `ComplianceComputer` table.

Database Column	Details
SecondUseComputerID	*Type:* integer. Key The laptop or second computer covered by this license's right of second use, relative to the installation on the primary computer tracked in the previous field. Foreign key to the ComplianceComputer table.
TotalLicenseGrabs	*Type:* integer For internal use only. Temporary storage for calculations of overlapping second use and multiple install rights.
IsExternalRoamingLink	*Type:* boolean Is this a second use link or is it actually an 'external roaming' right?

ReconcileSoftwareUserLicensePointsConsumedData Table

This is a staging table for SoftwareUserLicensePointsConsumed that is used during license reconciliation process, to store calculated values, and then bulk update the main table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 408: Database columns for ReconcileSoftwareUserLicensePointsConsumedData table

Database Column	Details
ComplianceUserID	*Type:* integer. Key The end-user. Foreign key to the ComplianceUser table.
SoftwareLicenseID	*Type:* integer. Key The license. Foreign key to the SoftwareLicense table.
LicensesConsumed	*Type:* integer The number of points (or entitlements) consumed for the license by the end-user.
LicensesUsed	*Type:* integer How many of the points consumed are for installations that are actually being used.
CalculatedConsumption	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

Database Column	Details
`LicenseMeasurementID`	*Type:* integer. Key. Nullable The associated SAP license measurement snapshot, where appropriate. Foreign key to the `LicenseMeasurement` table.

ReconcileVirtualMachineLayer Table

This serves as an intermediate table during reconciliation process to store virtual machines, pools and hosts in a generalized tree structure.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 409: Database columns for `ReconcileVirtualMachineLayer` table

Database Column	Details
`VMLayerID`	*Type:* integer. Key A unique identifier for a ReconcileVirtualMachineLayer.
`HostComplianceComputerID`	*Type:* integer. Key The host computer on which the layer resides, or the computer itself. Foreign key to the `ComplianceComputer` table.
`VMPoolID`	*Type:* integer. Key. Nullable The identifier of the virtual pool containing this VM, or the pool itself. Foreign key to the `VMPool` table.
`VMPoolTypeID`	*Type:* integer. Nullable The type of this VM pool. Foreign key to the `VMPoolType` table.
`VirtualMachineID`	*Type:* integer. Key. Nullable The identifier of this virtual machine. Foreign key to the `VirtualMachine` table.
`VMTypeID`	*Type:* integer. Nullable The type of this virtual machine. Foreign key to the `VMType` table.
`ParentVMPoolID`	*Type:* integer. Nullable The identifier of the parent VM pool of this pool. Foreign key to the `VMPool` table.
`ParentVMLayerID`	*Type:* integer. Key. Nullable The parent layer. Foreign key to the ReconcileVirtualMachineLayer table.

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key. Nullable The identifier of the computer running inside this virtual machine. Foreign key to the `ComplianceComputer` table.
`Name`	*Type:* text (max 256 characters). Nullable The name of the layer (host/pool/VM).
`PartialNumberOfProcessors`	*Type:* decimal. Nullable The fractional processor count available to this layer.
`NumberOfProcessors`	*Type:* decimal. Nullable The processor count for this layer.
`NumberOfCores`	*Type:* decimal. Nullable The core count for this layer.
`MaxNumberOfLogical Processors`	*Type:* decimal. Nullable The maximum number of logical processors count for this layer.
`NumberOfLogicalProcessors`	*Type:* decimal. Nullable The thread count for this layer.
`Depth`	*Type:* integer. Key The number of layers between this and the host computer.

RegistryEvidence Table

Reserved for future expansion.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 410: Database columns for `RegistryEvidence` table

Database Column	Details
`RegistryEvidenceID`	*Type:* integer. Key. Generated ID A unique identifier for a software registry evidence record.
`RegistryEvidenceHiveID`	*Type:* integer. Key The registry hive for the registry evidence.
`RegistryEvidenceKeyID`	*Type:* integer. Key The registry key for the registry evidence.

Database Column	Details
`RegistryEvidenceValueID`	*Type:* integer. Key The value of the registry evidence.
`RegistryData`	*Type:* text (max 400 characters). Key The data contained in the registry value for the registry evidence.
`Ignored`	*Type:* boolean If `True` this registry evidence is ignored for application recognition.
`IsShared`	*Type:* boolean

RegistryEvidenceHive Table

Reserved for future use.

Table 411: Database columns for `RegistryEvidenceHive` table

Database Column	Details
`RegistryEvidenceHiveID`	*Type:* integer. Key. Generated ID Unique identifier for a registry hive.
`RegistryHive`	*Type:* text (max 50 characters). Key The registry hive for the registry evidence.

RegistryEvidenceKey Table

Reserved for future use.

Table 412: Database columns for `RegistryEvidenceKey` table

Database Column	Details
`RegistryEvidenceKeyID`	*Type:* integer. Key. Generated ID Unique identifier for a registry key.
`RegistryKey`	*Type:* text (max 200 characters). Key The registry key for the registry evidence.

RegistryEvidenceValue Table

Reserved for future use.

Table 413: Database columns for `RegistryEvidenceValue` table

Database Column	Details
`RegistryEvidenceValueID`	*Type:* integer. Key. Generated ID Unique identifier for a registry value
`RegistryValue`	*Type:* text (max 50 characters). Key The registry value for the registry evidence.

RelatedInstalledInstallerEvidence Table

`RelatedInstalledInstallerEvidence` table holds parent-child relationship between installer evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 414: Database columns for `RelatedInstalledInstallerEvidence` table

Database Column	Details
`RelatedInstalled InstallerEvidenceID`	*Type:* integer. Key. Generated ID A synthetic unique identifier
`ParentInstallerEvidenceID`	*Type:* integer. Key An parent identifier for an installer evidence record. Foreign key to the `InstallerEvidence` table.
`ParentCompliance ComputerID`	*Type:* integer. Key An parent identifier for a computer record. Foreign key to the `ComplianceComputer` table.
`ParentAccessModeID`	*Type:* integer. Key The state an application was considered accessed. Foreign key to the `AccessMode` table.
`ChildInstallerEvidenceID`	*Type:* integer. Key An child identifier for an installer evidence record. Foreign key to the `InstallerEvidence` table.
`ChildComplianceComputerID`	*Type:* integer. Key An child identifier for a computer record. Foreign key to the `ComplianceComputer` table.

Database Column	Details
`ChildAccessModeID`	*Type:* integer. Key The state an application was considered accessed. Foreign key to the `AccessMode` table.
`IsCharged`	*Type:* boolean. Key The identifier used in the source connection to determine the pricing relation between parent and child installer evidence (specifies if it is charged = 1 or free = 0).
`ConfidenceLevel`	*Type:* integer. Nullable Confidence level for each bundled installer evidence (as a percentage).

RelatedInstalledInstallerEvidenceSourceMap Table

`RelatedInstalledInstallerEvidenceSourceMap` Maps related installed installer evidence to the evidence source type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 415: Database columns for `RelatedInstalledInstallerEvidenceSourceMap` table

Database Column	Details
`RelatedInstalled InstallerEvidence SourceMapID`	*Type:* integer. Key. Generated ID A synthetic unique identifier
`RelatedInstalled InstallerEvidenceID`	*Type:* integer. Key An identifier for an related installer evidence record. Foreign key to the `RelatedInstalledInstallerEvidence` table.
`ComplianceConnectionID`	*Type:* integer. Key The inventory source where the end-user was reported. Foreign key to the `ComplianceConnection` table.

RelatedInstalledSoftwareData Table

`RelatedInstalledSoftware` stores parent-child relationship among application installations. This is used for modelling application bundling.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 416: Database columns for `RelatedInstalledSoftwareData` table

Database Column	Details
`RelatedInstalled SoftwareID`	*Type:* integer. Key. Generated ID Unique identifier for this record.
`ParentInstalledSoftwareID`	*Type:* integer. Key The parent installed application. Foreign key to the `InstalledSoftware` table.
`ChildInstalledSoftwareID`	*Type:* integer. Key The child installed application. Foreign key to the `InstalledSoftware` table.
`IsCharged`	*Type:* boolean. Key The identifier used in the source connection to determine the pricing relation between parent and child installer evidence (specifies if it is charged = 1 or free = 0).
`ConfidenceLevel`	*Type:* integer. Nullable Confidence level for each bundled installer evidence (as a percentage).

SAPSoftwareLicense Table

`SAPSoftwareLicense` stores additional SAP-specific licensing information for SAP licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 417: Database columns for `SAPSoftwareLicense` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The SAP license. Foreign key to the `SoftwareLicense` table.
`SAPServerName`	*Type:* text (max 256 characters). Nullable The name of the SAP server. Should match the `ComputerName` of the record in the `ComplianceComputer` table which corresponds to the computer on which SAP is installed.

Database Column	Details
SAPBaseLicenseTypeID	*Type:* integer The SAP base license type, coming from the first pair of symbols in the “xx-xx-xx” license code. Foreign key to the SAPSoftwareLicenseType table.
SAPSpecialVersionID	*Type:* integer The SAP special version (language, country, etc.), coming from the second pair of symbols in the “xx-xx-xx” license type code. Foreign key to the SAPSoftwareLicenseType table.
SAPSurchargeID	*Type:* integer The SAP surcharge special version, coming from the third pair of symbols in the “xx-xx-xx” license code. Foreign key to the SAPSoftwareLicenseType table
SAPLicenseCode	*Type:* text (max 32 characters) The SAP license code, consisting of the license type, special version and surcharge.
HasUsage	*Type:* boolean Set this field to True if this license contains SAP usage/optimization information.
Description	*Type:* text (max 512 characters). Nullable A decription of the SAP license.

SAPSoftwareLicenseType Table

SAPSoftwareLicenseType lists the SAP base license types and special versions, and is part of the full “xx-xx-xx” code.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 418: Database columns for SAPSoftwareLicenseType table

Database Column	Details
SAPSoftwareLicenseTypeID	*Type:* integer. Key. Generated ID A unique identifier for this SAP base license type.
LicenseCode	*Type:* text (max 32 characters). Key The unique code for this license type - one of the “xx” parts of the full “xx-xx-xx” code.

Database Column	Details
SAPSpecialVersionID	*Type:* integer. Key. Nullable If this is a base license type, this field is NULL (and the LicenseCode comes from the first “xx” part of the full “xx-xx-xx” code). Otherwise, it is a special SAP version (the LicenseCode comes from the second or third “xx” part), and is foreign key to the SAPSpecialVersion table.
DescriptionResourceName	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing the license code description. Foreign key to the ComplianceResourceString table.
DescriptionDefaultValue	*Type:* text (max 256 characters) The text to display if the license code resource string has no translation.

SAPSpecialVersion Table

SAPSpecialVersion lists the types of special versions, indicating which part of the “xx-xx-xx” code the SAP software license type comes from.

Table 419: Database columns for SAPSpecialVersion table

Database Column	Details
SAPSpecialVersionID	*Type:* integer. Key. Generated ID A unique identifier for each SAPSpecialVersion. Possible values and the corresponding default strings are: • 1 = Generic special version • 2 = Surcharge special version.
InternalDescription	*Type:* text (max 50 characters) Internal description for developers.

ServicePack Table

Table 420: Database columns for ServicePack table

Database Column	Details
ServicePackID	*Type:* integer. Key. Generated ID A unique identifier for an ARL published service pack.

Database Column	Details
`ServicePackUID`	*Type:* integer. Key Factory generated identifier.
`ServicePackNameID`	*Type:* integer. Key Name of the service pack published by software publisher. Foreign key to the `ServicePackName` table.
`ReleaseDate`	*Type:* datetime. Nullable The availability date.
`EndOfSupportDate`	*Type:* datetime. Nullable The end of availability date.
`Notes`	*Type:* text. Nullable Notes for this service pack

ServicePackName Table

Table 421: Database columns for `ServicePackName` table

Database Column	Details
`ServicePackNameID`	*Type:* integer. Key. Generated ID A unique identifier for service pack.
`Name`	*Type:* text (max 256 characters). Key The service pack name

SoftwareAccessDeviceLicensePointsConsumedData Table

`SoftwareAccessDeviceLicensePointsConsumed` records how many license entitlements have been consumed for a given license by a given accessing device.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 422: Database columns for `SoftwareAccessDeviceLicensePointsConsumedData` table

Database Column	Details
`AccessingDeviceID`	*Type:* integer. Key. Nullable The accessing device under examination. Foreign key to the `AccessingDevice` table.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The compliance computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations actually being used.
`LicenseMeasurementID`	*Type:* integer. Key The license measurement ID. Foreign key to the `LicenseMeasurement` table.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

SoftwareAccessMode Table

The `SoftwareAccessMode` table holds the states an application has been accessed.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 423: Database columns for `SoftwareAccessMode` table

Database Column	Details
`SoftwareAccessModeID`	*Type:* integer. Key. Generated ID The primary key of the SoftwareAccessMode table.
`AccessModeID`	*Type:* integer. Key The access mode for the application. Foreign key to the `AccessMode` table.
`InstalledSoftwareID`	*Type:* integer. Key The installed software title to which the access mode applies. Foreign key to the `InstalledSoftware` table

Database Column	Details
IsACL	*Type:* boolean. Key Determines whether the software access mode record came from ACL data.

SoftwareAccessUserLicensePointsConsumedData Table

SoftwareAccessUserLicensePointsConsumed records how many license entitlements have been consumed for a given license by a given accessing user.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 424: Database columns for SoftwareAccessUserLicensePointsConsumedData table

Database Column	Details
AccessingUserID	*Type:* integer. Key. Nullable The accessing user under examination. Foreign key to the AccessingUser table.
ComplianceUserID	*Type:* integer. Key. Nullable The compliance user under examination. Foreign key to the ComplianceUser table.
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
LicensesUsed	*Type:* integer How many of the points consumed are for installations actually being used.
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.
CalculatedConsumption	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

SoftwareLicense Table

SoftwareLicense contains details of the software licenses managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 425: Database columns for `SoftwareLicense` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key. Generated ID A unique identifier for a software license.
`ParentLicenseID`	*Type:* integer. Key. Nullable The id of any bundle that this license is a part of.
`Name`	*Type:* text (max 256 characters). Key Name of the license.
`Version`	*Type:* text (max 60 characters). Key. Nullable Version of the license.
`Edition`	*Type:* text (max 60 characters). Nullable Edition of the license.
`LicenseTypeID`	*Type:* integer. Key The license type. Foreign key to the `SoftwareLicenseType` table.
`SoftwareLicenseMetricID`	*Type:* integer. Nullable Custom licensing metric for this license. Foreign key to the `SoftwareLicenseMetric` table.
`DurationID`	*Type:* integer The duration of this license. Foreign key to the `SoftwareLicenseDuration` table.
`SoftwareLicense ComplianceStatusID`	*Type:* integer The compliance status of this license. Foreign key to the `SoftwareLicenseComplianceStatus` table. Defaults to “Compliant”.
`LicenseStatusID`	*Type:* integer The status of this license. Foreign key to the `LicenseStatus` table.
`SoftwareLicense PurchaseTypeID`	*Type:* integer. Nullable The kind of purchase. Foreign key to the `SoftwareLicensePurchaseType` table.
`VendorID`	*Type:* integer. Key. Nullable The vendor from whom the license was purchased. Foreign key to the `Vendor` table.

Database Column	Details
PublisherID	*Type:* integer. Key. Nullable The software publisher associated with this license. Foreign key to the Vendor table.
ManagerID	*Type:* integer. Key. Nullable The manager of this license. Foreign key to the ComplianceUser table.
PartNo	*Type:* text (max 100 characters). Nullable The publisher's part number for this license.
SerialNumber	*Type:* text (max 256 characters). Nullable The serial number of the license.
LicenseKeyTypeID	*Type:* integer The type of license keys managed on this license. Foreign key to the SoftwareLicenseKeyType table. Defaults to "No key".
LicenseKey	*Type:* text (max 256 characters). Nullable The multiple-use license key of the license. Only used when the license key type is a multi-use key (for example, an Enterprise key used to cover multiple installs).
RequestNo	*Type:* text (max 60 characters). Nullable The request number for the license.
AcquisitionModeID	*Type:* integer The method of acquisition used for the asset this license covers. Defaults to Purchased. Foreign key to the AcquisitionMode table.
PurchaseOrderNumber	*Type:* text (max 50 characters). Nullable The purchase order number which was used to purchase the license.
PurchaseOrderDate	*Type:* datetime. Nullable The original purchase order date for the license.
PurchasePrice	*Type:* currency. Nullable The initial purchase price of the license.
PurchasePriceRateID	*Type:* integer. Nullable The currency rate applied to the purchase price of the license. Foreign key to the CurrencyRate table.
ChargeBackPrice	*Type:* currency. Nullable Amount to be charged for each computer on which the license is installed.
ChargeBackPriceRateID	*Type:* integer. Nullable The currency rate applied to the charge-back price. Foreign key to the CurrencyRate table.

Database Column	Details
ChargeBackPeriodTypeID	*Type:* integer The frequency with which the charge back price is charged. Defaults to None. Foreign key to the PeriodType table.
ExpiryDate	*Type:* datetime. Nullable The date this license expires. A NULL value means the license does not expire.
DeliveryDate	*Type:* datetime. Nullable The date this license became active. A NULL value means the license is inactive.
RetirementDate	*Type:* datetime. Nullable The date this license was retired. A NULL value means the license is active.
WarrantyExpiryDate	*Type:* datetime. Nullable The date the warranty on this license expires. This refers to a warranty Contract associated with the license.
NumberOfProcessors	*Type:* integer The number of processors that this license is for. This field is only used where the SoftwareLicenseType is Device (Processor-Limited) (LicenseTypeID = 11).
NumberOfCores	*Type:* integer The number of cores per processor that this license is for. This field is only used where the SoftwareLicenseType is Device (Core-Limited) (LicenseTypeID = 14).
NumberOfSockets	*Type:* integer The number of sockets that this license is for. The value zero is reserved to mean unlimited. This field is only used where the SoftwareLicenseType is Oracle Processor (LicenseTypeID = 16) or Oracle Named User Plus (LicenseTypeID = 17).
MinimumNumberOfProcessors	*Type:* integer The minimum number of processors that this license is for. This field is only used where the SoftwareLicenseType is Microsoft Server Processor (LicenseTypeID = 22).
MinimumNumberOf LicensesPerVM	*Type:* integer When licensing a Virtual Hardware System with a Microsoft Server Core license (LicenseTypeID = 33), consume license entitlements as though the virtual machine had at least this number of virtual threads.
MSPool	*Type:* text (max 120 characters). Nullable The name of the Microsoft license pool to which the license belongs.

Database Column	Details
`MSPoints`	*Type:* integer The points value of each installed version of this license, for use when calculating Microsoft licensing reports. This field is only valid when the `MSPool` field is set.
`WarrantyTypeID`	*Type:* integer The type of warranty for the license. Defaults to `None`. Foreign key to the `AssetWarrantyType` table.
`EndOfLifeRecipient`	*Type:* text (max 128 characters). Nullable The person or organization who received the asset associated with this license when it was disposed of.
`EndOfLifeReasonID`	*Type:* integer The reason the asset was associated with this license was disposed of. Foreign key to the `EndOfLifeReason` table.
`ResalePrice`	*Type:* currency. Nullable The amount the asset associated with this license was sold for.
`ResalePriceRateID`	*Type:* integer. Nullable The currency rate to be applied to the sale price of the asset associated with this license.
`CreationUser`	*Type:* text (max 256 characters). Nullable The operator who created this license.
`CreationDate`	*Type:* datetime The date and time the license was created.
`UpdatedUser`	*Type:* text (max 256 characters). Nullable The oeprator who last updated this license.
`UpdatedDate`	*Type:* datetime. Nullable The date and time the license was last updated.
`Comments`	*Type:* text. Nullable Comments about the license recorded by an operator. This field may also be used for storing license keys.
`NumberPurchased`	*Type:* integer The quantity of purchased license entitlements.
`NumberInstalled`	*Type:* integer The quantity of software installations accounted for by this license. This value is calculated and updated during the data import process, based on the software inventory details imported.

Database Column	Details
`NumberCalculated`	*Type:* integer The calculated consumption value for this license.
`ResourceUnitsConsumed`	*Type:* decimal The quantity consumed of a resource relevant to this license. The type of resource is identified by the associated `SoftwareLicenseMetric`. On the `IBM Resource Value Unit` license type this will have a points rule set applied to it to calculate the final license consumption value.
`PeakConsumed`	*Type:* integer This value has been deprecated and is no longer used.
`AdditionalBulkUsers Regular`	*Type:* integer A number of regular users associated with this license in addition to those specified individually in `SoftwareLicenseAllocation`. For `IBM User Value Unit` licenses this will have a points rule set applied to it to calculate the final license consumption value.
`AdditionalBulkUsers Infrequent`	*Type:* integer A number of infrequent users associated with this license in addition to those specified individually in `SoftwareLicenseAllocation`. For `IBM User Value Unit` licenses this will have an infrequent user multiplier and points rule set applied to it to calculate the final license consumption value.
`AdditionalBulkUsers External`	*Type:* integer A number of external users associated with this license in addition to those specified individually in `SoftwareLicenseAllocation`. For `IBM User Value Unit` licenses this will have an external user multiplier and points rule set applied to it to calculate the final license consumption value.
`UserMultiplierInfrequent`	*Type:* decimal The fraction of a regular user's consumption to use for infrequent users.
`UserMultiplierExternal`	*Type:* decimal The fraction of a regular user's consumption to use for external users.
`NumberUsed`	*Type:* integer The number of software installations covered by this license that are actually being used.
`NumberAllocated`	*Type:* integer The quantity of license entitlements allocated to individual end-users or computers.

Database Column	Details
`NumberAssigned`	*Type:* integer The quantity of license entitlements that have been assigned to enterprise groups.
`NumberOverridden`	*Type:* integer The quantity of overriden consumption allocated to individual end-users or computers.
`LastCalculatedNUPMinimum`	*Type:* integer. Nullable The last calculated minimum for `Oracle Named User Plus` licenses.
`AlwaysInstalled`	*Type:* boolean If this field is `True`, this license is considered in to be used whenever it is allocated. If `False`, software usage is considered separately, and allocation merely defines the corporation's modelling of who is expected to consume entitlements.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location linked to this license. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise corporate unit linked to this license. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any cost center in the enterprise that is linked to this license. Foreign key to the `GroupEx` table.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise category associated with this license. Foreign key to the `GroupEx` table.
`CoverInstallsOnVirtual Machines`	*Type:* boolean This is known in the UI as "Enable special handling for virtual machines". Its effect usually includes enabling sub-capacity licensing of virtual installs and/or capping of license consumption at the host level, but its exact effect depends on the specific license type. For license types that expose additional virtualization properties, this property must be set for the other properties to be used.
`LimitNumberOfVirtual Installs`	*Type:* boolean If this field is `True`, there is a limit to the number of installations on virtual machines that can be covered by each license entitlement. If `False`, one license entitlement may cover use on any number of virtual machines (typically on one host computer).

Database Column	Details
`NumberOfAllowedVirtual Installs`	*Type:* integer. Nullable If the license covers installations on virtual machines, this field specifies how many installations per host are allowed before an additional license entitlement (or point) is consumed.
`LimitVirtualInstalls IncludesHost`	*Type:* boolean If this field is `True`, the host operating system installations are included in the overall count of operating systems on the host when there is a limit on the number of allowed virtual installs for each license. If `False`, the host operating system is not considered when determining virtual install limits.
`NumberOfAllowed ProcessorsPerHost`	*Type:* integer. Nullable This field specifies how many processors per host are allowed before an additional license entitlement (or point) is consumed. Null provides the default of 1. Zero provides unlimited.
`UseHostProcessor Information`	*Type:* boolean If virtual installs are allowed, set this field to `True` if host information should be used when calculating license points consumed.
`AllowIBMPVUSubCapacity FromNonILMT`	*Type:* boolean If the license does not use host processor information (not full capacity), set this field to `True` to allow non-ILMT sub-capacity PVU consumption calculations to be used.
`LimitNumberOf ApplicationsEach LicensePointCovers`	*Type:* boolean If this field is `True`, there is a limit on the number of application installations allowed per license entitlement (or point). If `False` (the default), then a license entitles you to any number of installations of software linked to this license on the one computer.
`NumberOfApplication InstallsAllowedPer LicensePoint`	*Type:* integer. Nullable Where the previous field is set to `True`, this field defines the limited number of application installations allowed per entitlement (or point).
`LimitNumberOfComputers UserLicenseCanBe InstalledOn`	*Type:* boolean If this field is `True`, there is a limit to the number of computers that a user-based license can be linked to per entitlement (or point) consumed. If `False` (the default), a single end-user is entitled to install related software for his/her own use on any number of computers.
`NumberOfComputers AllowedPerUserLicense Point`	*Type:* integer. Nullable Where the previous field is set to `True`, this field defines the limited number of application installations an end-user is allowed per entitlement (or point).

Database Column	Details
`MinimumNumberOfUsers`	*Type:* integer The minimum number of users allowed for the license. This is used for `Oracle Named User Plus` licenses.
`MinimumNumberOfUsers MultipliedByProcessors`	*Type:* boolean Whether the previous field a fixed value for the license or it is a multiple of the number of processor points consumed by the license. This is used for `Oracle Named User Plus` licenses.
`SecondUsageWorkLaptop`	*Type:* boolean If this field is `True`, the license confers the right of second use on a work laptop. If `False`, there is no right of second use allowed on a work laptop.
`SecondUsageAtHome`	*Type:* boolean If this field is `True`, the license confers the right of second use on a home computer by the same end-user as the primary end-user of the license entitlement consumed at work. If `False` (the default), there is no right of second use allowed on a home computer.
`MultiUseInheritFrom Contract`	*Type:* boolean Set this field to `True` if the license should inherit the values for right of multiple use from a contract.
`MultiUseInheritFrom ContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that right of multiple use is inherited from. Foreign key to `Contract` table.
`SecondUsageInheritFrom Contract`	*Type:* boolean Set this field to `True` if the license should inherit the values for right of second use from a contract.
`SecondUsageInheritFrom ContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that right of second use is inherited from. Foreign key to `Contract` table.
`CoverInstallsOnVM InheritFromContract`	*Type:* boolean Set this field to `True` if the license should inherit virtual machine rights from a contract.
`CoverInstallsOnVM InheritFromContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that virtual machine rights are inherited from. Foreign key to `Contract` table.

Database Column	Details
AutoManageTitles	*Type:* boolean Set this field to True if the license should have its application links automatically managed for upgrade and downgrade rights. When this field is False, the operator must manually manage links between this license and any applications.
DowngradeOnlyToVersionLegacy	*Type:* text (max 60 characters). Nullable A repository for backward-compatible custom data.
UpgradeOnlyToVersionLegacy	*Type:* text (max 60 characters). Nullable A repository for backward-compatible custom data.
TrueUp	*Type:* boolean Set this field to True if the license is a true-up license (and so never goes into at risk).
UnlimitedConsumption	*Type:* boolean Set this field to True if the license is a unlimited license (and so never goes into at risk).
OracleLegacyLicenseTypeID	*Type:* integer. Key. Nullable The specific Oracle legacy license type, where appropriate. Foreign key to the OracleLegacyLicenseType table.
GroupAllocationTypeID	*Type:* integer. Key Allocations of entitlements under any license can be made to only one type of enterprise group, specified here. Foreign key to the GroupType table.
GroupAllocationReportingTypeID	*Type:* integer. Key Determines when enterprise groups will be considered to have risked their allocations of entitlements under this license. Foreign key to the SoftwareLicenseGroupAllocationReportingType table.
GroupAllocationComplianceLevel	*Type:* integer. Nullable Determines the depth level of groups to be used for calculating the risk status for a license.
CannotManuallyUpdateGroupAssignments	*Type:* boolean Set this field to True if the operator must make group assignments through a Assign License Entitlements dialog box. If False, changes can be made directly in the license properties pages.

Database Column	Details
`CalculateCompliance`	*Type:* boolean When this field is `True` (the default), and the associated `SoftwareLicenseType` also has its CalculateCompliance field set to `True` (true for most license types), license consumption must be calculated from imported inventory. When `False`, the compliance state must be imported, not calculated.
`IsSharableToLibrary`	*Type:* boolean Set this field to `True` (the default) if the license is sharable to the downloadable FlexNet Manager Suite ARL library.
`CopyEditionAndVersion`	*Type:* boolean Set this field to `True` (the default) if edition and version should be automatically copied to the license from the primary application.
`SoftwareLicenseTierTypeID`	*Type:* integer. Key. Nullable Type of the tier, for `Tiered Device` license type only. Foreign key to the `SoftwareLicenseTierType` table.
`SoftwareLicenseTierCode`	*Type:* text (max 256 characters). Nullable The actual tier of the license, corresponding to the tier type. For `Tiered Device` license type only.
`ImportedFromFNMEA`	*Type:* boolean Set this to `True` if this license was imported from FlexNet Manager for Engineering Applications.
`SoftwareLicensePoints RuleSetID`	*Type:* integer. Key. Nullable The points rule set used to calculate compliance for this license. Foreign key to the `SoftwareLicensePointsRuleSet` table.
`BaselineQuantity`	*Type:* integer. Nullable The baseline value for this license
`BaselineDate`	*Type:* datetime. Nullable The date at which the baseline applies.
`AlternateNon InventoriedUsers`	*Type:* integer. Nullable Number of non-inventoried users who are consuming this license. For `Oracle Named User Plus` and `Oracle Application User` licenses, this acts as an alternate mean to specify user consumption in the case where no instance users are available from inventory. The number of non-inventoried users are added to the number of unique users found from inventory when number installed and number used are calculated in license reconcile.

Database Column	Details
`InheritLicenseMobility FromContract`	*Type:* boolean Set this field to `True` if this license inherits its license mobility rights from a contract. If `False` (the default), license mobility rights must be configured directly on the license properties.
`InheritLicenseMobility FromContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that mobility rights are inherited from. Foreign key to the `Contract` table.
`InheritLicense ConsumptionFromContract`	*Type:* boolean Set this field to `True` if this license inherits its license consumption rules from a contract. If `False` (the default), license consumption rules must be configured directly on the license properties.
`InheritLicense ConsumptionFrom ContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that license consumption rules are inherited from. Foreign key to the `Contract` table.
`InheritProcessorLimits FromContract`	*Type:* boolean Set this field to `True` if this license inherits its processor limits rights from a contract. If `False` (the default), license processor limits rights must be configured directly on the license properties.
`InheritProcessorLimits FromContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that processor limits rights are inherited from. Foreign key to the `Contract` table.
`AllowMaintenanceToExpire`	*Type:* boolean If the value `True`, maintenance added to license will not be renewed. If this flag is set, use rights for this license will be automatically updated once the maintenance expires.
`AutoSynchronized`	*Type:* boolean If the value is `True` the license information will be synchronized with the information from the source connection.
`UseRightsAutoUpdated`	*Type:* boolean If the value `True`, the use rights of this license has been automatically updated when all the maintenace expired. Else, use rights has not been updated.

SoftwareLicenseAllocation Table

`SoftwareLicenseAllocation` records the allocations of individual computers, end-users, enterprise groups or instances to licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 426: Database columns for `SoftwareLicenseAllocation` table

Database Column	Details
`SoftwareLicenseAllocationID`	*Type:* integer. Key. Generated ID A unique identifier for the license allocation record.
`SoftwareLicenseID`	*Type:* integer. Key The license that has been allocated. Foreign key to the `SoftwareLicense` table.
`ComplianceComputerID`	*Type:* integer. Key. Nullable The computer to which the license is allocated. Foreign key to the `ComplianceComputer` table.
`ComplianceUserID`	*Type:* integer. Key. Nullable The end-user to which the license is allocated. Foreign key to the `ComplianceUser` table.
`InstanceID`	*Type:* integer. Key. Nullable The instance to which the license is allocated. Foreign key to the `Instance` table.
`GroupExID`	*Type:* text (max 128 characters). Key. Nullable The enterprise group to which the license is assigned. Foreign key to the `GroupEx` table.
`LicenseUserID`	*Type:* integer. Key. Nullable The external end-user to whom the license is allocated. Foreign key to the `LicenseUser` table.
`AccessingUserID`	*Type:* integer. Key. Nullable The external accessing user to whom the license is allocated. Foreign key to the `AccessingUser` table.
`SoftwareLicenseAllocationUserTypeID`	*Type:* integer. Key. Nullable Indicates for user allocations whether they are a regular user or some special type of user for this license. Foreign key to the `SoftwareLicenseAllocationUserType` table.
`NumberAllocated`	*Type:* integer. Nullable The number of license entitlements assigned. This is used for group assignments.
`NumberUsed`	*Type:* integer. Nullable The number of license entitlements where the application is recorded as being used.

Database Column	Details
SoftwareLicenseAllocationStatusID	*Type:* integer. Nullable Indicates the status of an allocation. Foreign key to the SoftwareLicenseAllocationStatus table.
SoftwareLicenseKeyID	*Type:* integer. Key. Nullable The software license key that is allocated to this end-user/computer. Foreign key to the SoftwareLicenseKey table.
SoftwareLicenseExemptionReasonID	*Type:* integer. Key. Nullable The reason why this allocation is exempted from consuming a license entitlement. Foreign key to the SoftwareLicenseExemptionReason table.
IsIncludedForLicenseRec	*Type:* boolean. Key Indicates allocations whether they should be used in regular license reconciliation.

SoftwareLicenseAllocationStatus Table

SoftwareLicenseAllocationStatus is a static table storing a collection of status values for a license allocation.

Table 427: Database columns for SoftwareLicenseAllocationStatus table

Database Column	Details
SoftwareLicenseAllocationStatusID	*Type:* integer. Key. Generated ID A unique identifier for each SoftwareLicenseAllocationStatus. Possible values and the corresponding default strings are: • 1 = Allocated • 2 = Awaiting Inventory • 3 = Permanent • 4 = Unallocated.
StatusResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an allocation status. Foreign key to the ComplianceResourceString table.
StatusDefaultValue	*Type:* text (max 50 characters) The text to display if the status resource string has no translation.

SoftwareLicenseAllocationUserType Table

`SoftwareLicenseAllocationUserType` is a static table storing a collection of user type values for a license allocation.

Table 428: Database columns for `SoftwareLicenseAllocationUserType` table

Database Column	Details
`SoftwareLicenseAllocationUserTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseAllocationUserType`. Possible values and the corresponding default strings are: • 1 = Normal • 2 = Infrequent • 3 = External.
`UserTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a user allocation type. Foreign key to the `ComplianceResourceString` table.
`UserTypeDefaultValue`	*Type:* text (max 50 characters) The text to display if the user type resource string has no translation.

SoftwareLicenseBreachReasonData Table

`SoftwareLicenseBreachReasonData` identifies the reasons why non-compliant software licenses are in this state.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 429: Database columns for `SoftwareLicenseBreachReasonData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The software license. Foreign key to the `SoftwareLicense` table.
`LicenseBreachReasonID`	*Type:* integer. Key The license risk reason. Foreign key to the `LicenseBreachReason` table.
`LicenseMeasurementID`	*Type:* integer. Key The license measurement ID. Foreign key to the `LicenseMeasurement` table.

SoftwareLicenseChangeEvent Table

The `SoftwareLicenseChangeEvent` table holds the details of all license change events.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 430: Database columns for `SoftwareLicenseChangeEvent` table

Database Column	Details
`ComplianceEventID`	*Type:* integer. Key A unique identifier and foreign key to the `ComplianceEvent` table.
`SoftwareLicenseID`	*Type:* integer. Key The license involved in the change event. Foreign key to the `SoftwareLicense` table.
`SoftwareTitleID`	*Type:* integer. Key The software title that needs to be added or removed. Foreign key to the `SoftwareTitle` table.
`SoftwareLicenseChange EventSourceID`	*Type:* integer What caused the event. Foreign key to the `SoftwareLicenseChangeEventSource` table.
`SoftwareLicenseChange EventReasonID`	*Type:* integer The type of event. Foreign key to the `SoftwareLicenseChangeEventReason` table.
`SoftwareTitleLicense ReasonID`	*Type:* integer. Nullable When a software title has been added to a license, the reason it has been added (ie because upgrade rights allow it, for example). Foreign key to the `SoftwareTitleLicenseReason` table.

SoftwareLicenseChangeEventReason Table

`SoftwareLicenseChangeEventReason` is a static table holding all the valid reasons why a license change event was generated.

Table 431: Database columns for `SoftwareLicenseChangeEventReason` table

Database Column	Details
`SoftwareLicenseChange EventReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseChangeEventReason`. Possible values and the corresponding default strings are: • 1 = Add Application • 2 = Remove Application.
`ChangeEventReason ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a change event reason. Foreign key to the `ComplianceResourceString` table.
`ChangeEventReason DefaultValue`	*Type:* text (max 100 characters) The text to display if the reason resource string has no translation.

SoftwareLicenseChangeEventSource Table

`SoftwareLicenseChangeEventSource` is a static table holding all the valid sources of license change events.

Table 432: Database columns for `SoftwareLicenseChangeEventSource` table

Database Column	Details
`SoftwareLicenseChange EventSourceID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseChangeEventSource`. Possible values and the corresponding default strings are: • 1 = ARL • 2 = Software License • 3 = Software Title (the application properties) • 4 = Contract • 5 = Version (changing the relative priorities or weights of application versions linked to a license) • 6 = Edition (changing the relative priorities or weights of application editions linked to a license).
`ChangeEventSource ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a change event source. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
`ChangeEventSource DefaultValue`	*Type:* text (max 100 characters) Default value for a license change event source if the source resource has no translation.

SoftwareLicenseCloudServiceProvider Table

`SoftwareLicenseCloudServiceProvider` contains the cloud service providers supported by licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 433: Database columns for `SoftwareLicenseCloudServiceProvider` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key Identifies a software license. Foreign key to the `SoftwareLicense` table.
`CloudServiceProviderID`	*Type:* integer. Key Identifies a cloud service provider. Foreign key to the `CloudServiceProvider` table.

SoftwareLicenseComplianceStatus Table

`SoftwareLicenseComplianceStatus` is a static table listing valid compliance states for a license.

Table 434: Database columns for `SoftwareLicenseComplianceStatus` table

Database Column	Details
`SoftwareLicense ComplianceStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseComplianceStatus`. Possible values and the corresponding default strings are: • 1 = Compliant • 2 = At Risk • 3 = Unknown • 4 = Not Tracked.

Database Column	Details
StatusResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a compliance status. Foreign key to the ComplianceResourceString table.
StatusDefaultValue	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

SoftwareLicenseComputerProblemData Table

SoftwareLicenseComputerProblemData identifies the problems with individual ComplianceComputers that contributed to an associated license having an unknown compliance status. For example, some license types caculate entitlement consumption based on the number of processor cores present in a computer, but that detail is not available from Microsoft SCCM before version 2012, so computers from this inventory source will cause associated licenses to have unknown compliance status.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 435: Database columns for SoftwareLicenseComputerProblemData table

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key The software license. Foreign key to the SoftwareLicense table.
ComplianceComputerID	*Type:* integer. Key The computer consuming license entitlements. Foreign key to the ComplianceComputer table.
SoftwareLicense ComputerProblemTypeID	*Type:* integer The type of problem this computer's inventory causes for a given license. For example, core-based licenses require accurate inventory of processor core counts to determine ther compliance status. Foreign key to the SoftwareLicenseComputerProblemType table.
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.

SoftwareLicenseComputerProblemType Table

SoftwareLicenseComputerProblemType is a static table holding the collection of problems that a computer can have which might cause a particular type of license to have an unknown compliance status.

Table 436: Database columns for `SoftwareLicenseComputerProblemType` table

Database Column	Details
`SoftwareLicenseComputerProblemTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseComputerProblemType`. Possible values and the corresponding default strings are: • 1 = Core count missing from inventory • 2 = Processor count missing from inventory • 3 = Socket count missing from inventory • 4 = Thread count missing from inventory.
`ProblemTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a problem type. Foreign key to the `ComplianceResourceString` table.
`ProblemTypeDefaultValue`	*Type:* text (max 512 characters) The text to display if the problem type resource string has no translation.

SoftwareLicenseConnection Table

`SoftwareLicenseConnection` stores a link between software licenses in `SoftwareLicense` which have been reported in inventory, and external IDs which can be used to identify them in their inventory sources.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 437: Database columns for `SoftwareLicenseConnection` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key A unique identifier for the software license. Foreign key to the `SoftwareLicense` table.
`ComplianceConnectionID`	*Type:* integer. Key The inventory source where the software license was reported. Foreign key to the `ComplianceConnection` table.
`ExternalLicenseID`	*Type:* big integer. Key A (hopefully unique) identifier for the software license in the external inventory source.

SoftwareLicenseContract Table

`SoftwareLicenseContract` links licenses to related contracts.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 438: Database columns for `SoftwareLicenseContract` table

Database Column	Details
`SoftwareLicenseContractID`	*Type:* integer. Key. Generated ID A unique identifier for this record.
`SoftwareLicenseID`	*Type:* integer. Key The license to which the contract is related. Foreign key to the `SoftwareLicense` table.
`ContractID`	*Type:* integer. Key The contract related to the license. Foreign key to the `Contract` table.

SoftwareLicenseCoresConsumedData Table

`SoftwareLicenseCoresConsumedData` records how many cores have contributed to license point consumption for a given license by a given computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 439: Database columns for `SoftwareLicenseCoresConsumedData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`CoresConsumed`	*Type:* integer The number of cores that have contributed to license point consumption for the license on the computer.

Database Column	Details
`LicenseMeasurementID`	*Type:* integer. Key The license measurement ID. Foreign key to the `LicenseMeasurement` table.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

SoftwareLicenseCreation Table

`SoftwareLicenseCreation` records which SKU definition was used to create a software license.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 440: Database columns for `SoftwareLicenseCreation` table

Database Column	Details
`SoftwareLicenseCreationID`	*Type:* integer. Key. Generated ID A unique identifier for this record.
`SoftwareLicenseID`	*Type:* integer. Key The software license created. Foreign key to the `SoftwareLicense` table.
`SoftwareSkuID`	*Type:* integer. Key. Nullable The SKU that was recognized. This value is optional, as the software license could have been created directly using a definition selected by the operator, without a SKU being used as the link. Foreign key to the `SoftwareSku` table.
`SoftwareLicense DefinitionID`	*Type:* integer. Key The license definition used to create the software license. Foreign key to the `SoftwareLicenseDefinition` table.
`LicenseDefinitionVersion`	*Type:* integer. Key Which version of the license definition was used to create the software license.

SoftwareLicenseDefinition Table

`SoftwareLicenseDefinition` maps SKUs to the license definitions and applications that it relates to.

Table 441: Database columns for `SoftwareLicenseDefinition` table

Database Column	Details
`SoftwareLicenseDefinitionID`	*Type:* integer. Key. Generated ID A unique identifier for this record.
`LicenseDefinitionFactoryUID`	*Type:* text (max 30 characters). Key The FlexNet Manager Suite factory unique identifier for this record.
`NonMaintenanceLicenseDefinitionFactoryUID`	*Type:* text (max 30 characters). Key. Nullable The FlexNet Manager Suite factory without maintenance for this record.
`LicenseDefinitionTypeID`	*Type:* integer. Key The lciense definition type. Foreign key to the `LicenseDefinitionType` table.
`LicenseDefinition`	*Type:* text The license definition. Contains information relevant to license creation and application links.
`ProductName`	*Type:* text (max 2000 characters) When a license is created using this definition, this will be its license name.
`ProductVersion`	*Type:* text (max 2000 characters) When a license is created using this definition, this will be its license version.
`ProductPublisher`	*Type:* text (max 2000 characters) When a license is created using this definition, this will be its license publisher.
`LicenseTypeID`	*Type:* integer. Key This definition will create a license of this type. Foreign key to the `LicenseType` table.
`IsUpgrade`	*Type:* boolean Set this field to `True` if this definition will create an upgrade license. If this field is `False`, this definition creates a standard license.
`Version`	*Type:* integer. Key The current version of this SKU definition.
`PreviousVersion`	*Type:* integer. Key. Nullable The version of the SKU definition prior to the current version.
`CreationDate`	*Type:* datetime The date that this record was created.
`UpdatedDate`	*Type:* datetime. Nullable The date that this record was last updated.

SoftwareLicenseDuration Table

The collection of durations for which a license can be active.

Table 442: Database columns for `SoftwareLicenseDuration` table

Database Column	Details
`SoftwareLicenseDurationID`	*Type:* integer. Key. Generated ID A unique identifier for a license duration. Possible values (and associated default names) are: • 1 = Perpetual • 2 = TimeLimited • 3 = Subscription.
`DurationResourceName`	*Type:* text (max 256 characters). Key The name of the resource string containing the text to display on the user interface.
`DurationDefaultValue`	*Type:* text (max 100 characters) The value to display if there is no resource string available for this status

SoftwareLicenseExemptionReason Table

The collection of types exemption reasons that may be associated with software license allocations.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 443: Database columns for `SoftwareLicenseExemptionReason` table

Database Column	Details
`SoftwareLicense ExemptionReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseExemptionReason`. Possible values and the corresponding default strings are: • 1 = Alpha, beta, early support program • 2 = Backup, disaster recovery • 3 = Component of a non-PVU licensed offering • 4 = Component is not compatible with the server or agent system • 5 = Development • 6 = Evaluation, trial • 7 = Fail-over • 8 = Not eligible for PVU licensing • 9 = Other • 10 = Second use • 11 = Test • 12 = Covered by related product • 13 = Covered by virtual application access • 14 = No usage for virtual application within specified time limit
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an exemption reason. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

SoftwareLicenseExemptionRole Table

`SoftwareLicenseExemptionRole` table holds information on role exemption rule for licenses. Contains many to many relationship between licenses and device roles.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 444: Database columns for `SoftwareLicenseExemptionRole` table

Database Column	Details
`SoftwareLicense ExemptionRoleID`	*Type:* integer. Key. Generated ID A unique identifier for the license exemption role record.
`SoftwareLicenseID`	*Type:* integer. Key The license that has role exemption rule. Foreign key to the `SoftwareLicense` table.
`ExemptionRoleID`	*Type:* integer. Key The device role that is exempted from license consumption. Foreign key to the `ComplianceComputerRole` table.
`ExemptionLimit`	*Type:* integer. Nullable The number of devices that can be exempted, having an exempted role.

SoftwareLicenseGroupAllocationReportingType Table

`SoftwareLicenseGroupAllocationReportingType` stores the set of tests that can be used to determine whether a license is in “group at risk” for one or more of its associated enterprise groups.

Table 445: Database columns for `SoftwareLicenseGroupAllocationReportingType` table

Database Column	Details
`SoftwareLicenseGroup AllocationReporting TypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseGroupAllocationReportingType`. Possible values and the corresponding default strings are: • 0 = None • 1 = Consumed Exceeds Purchased • 2 = Consumed Exceeds Assigned.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a group at risk test type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the type resource string has no translation.

SoftwareLicenseGroupAssignmentHistory Table

`SoftwareLicenseGroupAssignmentHistory` is used to keep track of changes made to assignments of software license entitlements to enterprise groups.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 446: Database columns for `SoftwareLicenseGroupAssignmentHistory` table

Database Column	Details
`SoftwareLicenseGroup AssignmentHistoryID`	*Type:* integer. Key. Generated ID A unique identifier for the history record.
`GroupExID`	*Type:* text (max 128 characters). Key This is the primary group that had a change of assignments. Foreign key to the `GroupEx` table.
`FromGroupExID`	*Type:* text (max 128 characters). Key. Nullable If assignments were transferred, this is the source group who had assignments taken away. Foreign key to the `GroupEx` table.
`FromGroupExPath`	*Type:* text (max 500 characters). Nullable The path of the group that assignments were transferred from. This field is used to display the group name (at the time that the transfer took place) when showing history after the group has been deleted.
`HistoryDate`	*Type:* datetime The date of the change.
`SoftwareLicenseID`	*Type:* integer. Key The license for which entitlements are being assigned. Foreign key to the `SoftwareLicense` table.
`UserName`	*Type:* text (max 60 characters) The operator who made the change.
`Comments`	*Type:* text (max 2000 characters). Nullable Comments recorded about the change.
`NumberAdded`	*Type:* integer The number of assignments added to or removed from the group.
`Total`	*Type:* integer The progressive total of assignments to the group following this change.

Database Column	Details
SoftwareLicenseGroup AssignmentHistoryTypeID	*Type:* integer. Key The type of history record. This records the kind of change that was made (eg, a flat increase/decrease of the assignment count, a transfer, and so on). Foreign key to the SoftwareLicenseGroupAssignmentHistoryType table.

SoftwareLicenseGroupAssignmentHistoryType Table

SoftwareLicenseGroupAssignmentHistoryType stores a collection of the types of history record that can be stored in the SoftwareLicenseGroupAssignmentHistory table.

Table 447: Database columns for SoftwareLicenseGroupAssignmentHistoryType table

Database Column	Details
SoftwareLicenseGroup AssignmentHistoryTypeID	*Type:* integer. Key. Generated ID A unique identifier for each SoftwareLicenseGroupAssignmentHistoryType. Possible values and the corresponding default strings are: • 1 = Manual (manual increase/decrease of the group assignment quantity) • 2 = ManualDialog (manual increase/decrease of the group assignment quantity, using the Assign License Entitlements dialog_ • 3 = Transfer (a transfer of entitlements from one enterprise group to another) • 4 = ClearAssignments (the Clear Assignments button has been used to remove all entitlements from a group) • 5 = AssignPurchased (the Assign Purchases button has been used to copy purchases within the group to the group assignment total) • 6 = ChangeGroupType (assignments have been cleared because the group assignment type has been changed).
ResourceName	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing a history type. Foreign key to the ComplianceResourceString table.
SoftwareLicenseGroup AssignmentHistoryType Name	*Type:* text (max 64 characters). Key A description of the history type.
DefaultValue	*Type:* text (max 50 characters) The text to display if the type resource string has no translation.

SoftwareLicenseGroupBreachStatus Table

`SoftwareLicenseGroupBreachStatus` stores the collection of possible outcomes of group at risk testing.

Table 448: Database columns for `SoftwareLicenseGroupBreachStatus` table

Database Column	Details
`SoftwareLicenseGroup BreachStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseGroupBreachStatus`. Possible values and the corresponding default strings are: • 0 = Ignored • 1 = Group Not At Risk • 2 = Group At Risk.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a group at risk status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the status resource string has no translation.

SoftwareLicenseGroupPointsConsumedData Table

`SoftwareLicenseGroupPointsConsumed` records the licenses pre-calculated local and rolledup totals for groups.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 449: Database columns for `SoftwareLicenseGroupPointsConsumedData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The license that owns the pre-calculated totals for a group. Foreign key to the `SoftwareLicense` table.
`GroupTypeID`	*Type:* integer. Key Type of the group(Location, Cost center, etc)
`GroupExID`	*Type:* text (max 128 characters). Key. Nullable The group where the local and rolledup values are calculated. Foreign key to the `GroupEx` table.

Database Column	Details
`RolledUpNumberConsumed`	*Type:* integer The sum of points consumed of the current group and of all its child groups.
`LocalNumberConsumed`	*Type:* integer The sum of points consumed of the current group
`RolledUpNumberUsed`	*Type:* integer The sum of used points f the current group and of all its child groups.
`LocalNumberUsed`	*Type:* integer The sum of used points of the current group
`RolledUpNumberPurchased`	*Type:* integer The rolled up purchase counts of the license.
`LocalNumberPurchased`	*Type:* integer The local purchase counts of the license
`LicenseMeasurementID`	*Type:* integer. Key The license measurement ID. Foreign key to the `LicenseMeasurement` table.
`RolledUpNumberCalculated`	*Type:* integer The sum of points calculated for the current group and of all its child groups.
`LocalNumberCalculated`	*Type:* integer The sum of points calculated for the current group.

SoftwareLicenseILMTPointsConsumedData Table

`SoftwareLicenseILMTPointsConsumed` records how many PVU counts and their corresponding core counts have been consumed for a given license by a given computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 450: Database columns for `SoftwareLicenseILMTPointsConsumedData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
CoreCount	*Type:* integer The number of licensable cores for the license on the computer.
PVUCount	*Type:* integer The number of PVU counts consumed for the license on the computer.
PeakPVUCount	*Type:* integer The number of PVU counts consumed for the license on the computer at the time where the peak for this license occurred.
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.
CalculatedConsumption	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.

SoftwareLicenseKey Table

The SoftwareLicenseKey table contains installation keys that are linked to software licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 451: Database columns for SoftwareLicenseKey table

Database Column	Details
SoftwareLicenseKeyID	*Type:* integer. Key. Generated ID A unique identifier for this license key.
SoftwareLicenseID	*Type:* integer. Key The software license that this installation key belongs to. Foreign key to the SoftwareLicense table.
KeyValue	*Type:* text (max 400 characters). Key The installation key value.

SoftwareLicenseKeyType Table

The collection of types of installation keys that may be associated with software licenses.

Table 452: Database columns for `SoftwareLicenseKeyType` table

Database Column	Details
`SoftwareLicenseKeyTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseKeyType`. Possible values and the corresponding default strings are: • 1 = No keys • 2 = One (multi-install) key per license • 3 = One (multi-install) key per application • 4 = One (single-install) key per installation. • 5 = One (multi-install) key per installation.
`KeyTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a license key type. Foreign key to the `ComplianceResourceString` table.
`KeyTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

SoftwareLicenseMetric Table

`SoftwareLicenseMetric` holds the pre-defined list of licensing custom metrics.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 453: Database columns for `SoftwareLicenseMetric` table

Database Column	Details
`SoftwareLicenseMetricID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseMetric`. Possible values and the corresponding default strings are: • 1 = $M cost of goods sold • 2 = $M freight under management • 3 = $M in revenue • 4 = $M revenue under management • 5 = Active Oracle node • 6 = Cluster • 7 = Compensated individual • 8 = Connector • 9 = Developer • 10 = Drive • 11 = Electronic order line • 12 = Employees • 13 = Expense report • 14 = External connector • 15 = External recipient • 16 = Field technician • 17 = Floating user • 18 = Form • 19 = Front end GB • 20 = Front end TB • 21 = FTE student • 22 = Gateway • 23 = Gigabyte • 24 = Guest • 25 = Host • 26 = Internet connector

Database Column	Details
	• 27 = IP • 28 = Mailbox • 29 = OSE • 30 = Partner organization • 31 = Person • 32 = Per 1000 invoice lines • 33 = Per 1000 records • 34 = Per rule set • 35 = Per tape drive • 36 = Port • 37 = Record • 38 = Server bundle • 39 = Service order line • 40 = Storage domain • 41 = Terabyte • 42 = Tiered NAS device • 43 = Tivoli management point • 44 = Trainee • 45 = Transaction • 46 = UPK module • 47 = Folio download • 48 = Document • 49 = Per 1000 minutes • 50 = Exam • 51 = Support incidents • 52 = Time • 53 = Recipient • 54 = Employees + non employees

Database Column	Details
	• 100 (Oracle Processor) = Cores • 101 (Oracle NUP) = Cores • 102 (Oracle Processor) = Sockets • 103 (Oracle NUP) = Sockets • 150 (IBM RVU) = Million Service Units • 151 (IBM RVU) = Messages • 152 (IBM RVU) = Engines • 153 (IBM RVU) = Terabytes • 154 (IBM RVU) = Tape Drives • 155 (IBM RVU) = Gigabytes • 156 (IBM RVU) = Premium Income $US Billions (1 Resource Per US$500M, rounded up to nearest US$500M) • 157 (IBM RVU) = Capital Asset Value ($US Billions) • 158 (IBM RVU) = Activated Processor Cores • 159 (IBM RVU) = Pages Per Month • 160 (IBM RVU) = Soft Goods & Services Entities • 161 (IBM RVU) = Manufactured Goods Entities • 162 (IBM RVU) = Assets & Commodities Entities • 163 (IBM RVU) = Locations Entities and Trading Partners & Parties Entities • 164 (IBM RVU) = Client Devices • 165 (IBM RVU) = Server Devices • 166 (IBM RVU) = Annual Web Sessions • 167 (IBM RVU) = 1,000 Web Interactions • 168 (IBM RVU) = 1,000,000 Data Source Records • 169 (IBM RVU) = 1,000,000 Monthly Server Calls • 170 (IBM RVU) = 1,000,000 Subscribers • 171 (IBM RVU) = 10,000 Records • 172 (IBM RVU) = 100 Records • 173 (IBM RVU) = 100,000 Records

Database Column	Details
	• 174 (IBM RVU) = Assets • 175 (IBM RVU) = Authorized Retail, Host, and Mobile Sites • 176 (IBM RVU) = Conversion Units • 177 (IBM RVU) = Enterprise Identifiers • 178 (IBM RVU) = Managed Devices • 179 (IBM RVU) = Records • 180 (IBM RVU) = Resources • 181 (IBM RVU) = Revenue $US Billions • 182 (IBM RVU) = Secondary Sites • 183 (IBM RVU) = Servers • 184 (IBM RVU) = Transportation Events Per Calendar Month • 185 (IBM RVU) = Value Units • 186 (IBM RVU) = Virtual Servers • 187 (IBM RVU) = Web Pages.
`SoftwareLicenseTypeID`	*Type:* integer. Key The software license type to which this metric applies. Foreign key to the `SoftwareLicenseType` table.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a licensing metric. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the ResourceName has no translation.

SoftwareLicensePVUPointsConsumedComputersSCD Table

`SoftwareLicensePVUPointsConsumedComputers` relationship between a consuming computer and the computers involved in the consumption. This should include VMs with installations of only supplementary titles as well.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 454: Database columns for `SoftwareLicensePVUPointsConsumedComputersSCD` table

Database Column	Details
`RegionID`	*Type:* integer. Key The region consumption is being recorded for. Foreign key to the `Region` table. If NULL, device not belonging to any region
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`HostID`	*Type:* integer. Key The host machine. For standalone computer, this will be the computer itself. Foreign key to the `ComplianceComputer` table.
`ComplianceComputerID`	*Type:* integer. Key The machine involved in consuming PVU point under the host. For standalone computer, this will be the computer itself. Foreign key to the `ComplianceComputer` table.
`CoreCount`	*Type:* integer Number of cores at the time of consumption calculation
`IsEligibleInventory`	*Type:* boolean Is the device eligible for inventory at consumption calculation
`Pool1Name`	*Type:* text (max 256 characters). Nullable Name of 1st level pool of this consuming device (assumption: max up to 2 levels of pool hierarchy).
`Pool1CoreCount`	*Type:* integer. Nullable Core count of 1st level pool of this consuming device (assumption: max up to 2 levels of pool hierarchy).
`Pool2Name`	*Type:* text (max 256 characters). Nullable Name of 2nd level pool of this consuming device (assumption: max up to 2 levels of pool hierarchy).
`Pool2CoreCount`	*Type:* integer. Nullable Core count of 2nd level pool of this consuming device (assumption: max up to 2 levels of pool hierarchy).
`ValidFrom`	*Type:* datetime. Key The start time of the PVU usage
`ValidTo`	*Type:* datetime The end time of the PVU usage

Database Column	Details
LicenseMeasurementID	*Type:* integer. Key The snapshot ID. Foreign key to the LicenseMeasurement table.

SoftwareLicensePVUPointsConsumedPerHostSCD Table

SoftwareLicensePVUPointsConsumedPerHostSCD records how many PVU counts and their corresponding core counts have been consumed for a given license by a given computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 455: Database columns for SoftwareLicensePVUPointsConsumedPerHostSCD table

Database Column	Details
RegionID	*Type:* integer. Key The region consumption is being recorded for. Foreign key to the Region table. If NULL, device not belonging to any region
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
ComplianceComputerID	*Type:* integer. Key The host computer. Foreign key to the ComplianceComputer table.
EligibleForSubCapacity	*Type:* boolean Whether the device is eligible for contributing sub capacity consumption to the license Device is eligible if The license setting has 'Allow sub-capacity licensing for sources other than approved IBM tools and FlexNet Manager Suite' OR IF license setting is 'Use sub-capacity license calculations where available' AND The Host\ Computer is inventoried by Flexera agent. AND The system has ILMT mode turned on
PVUPerCore	*Type:* integer The number of PVU point per core for the license on the computer.
PVUSubcapacityLimit	*Type:* integer The maximum number of PVU points can be consumed for the license on the computer.

Database Column	Details
PVUSubcapacity	*Type:* integer The number of sub-capacity PVU points consumed for the license on the computer.
CoreSubcapacityLimit	*Type:* integer The maximum number of cores that can be consumed for the license on the computer.
CoreSubcapacity	*Type:* integer The number of cores consumed for the license on the computer.
ProcessorCount	*Type:* integer. Nullable The number of processors consumed for the license on the computer.
ValidFrom	*Type:* datetime. Key The start time of the PVU usage
ValidTo	*Type:* datetime The end time of the PVU usage
LicenseMeasurementID	*Type:* integer. Key The snapshot ID. Foreign key to the LicenseMeasurement table.

SoftwareLicensePVUPointsConsumedPerRegionSCD Table

SoftwareLicensePVUPointsConsumedPerRegionSCD records PVU consumption for a given license in a given region.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 456: Database columns for SoftwareLicensePVUPointsConsumedPerRegionSCD table

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
RegionID	*Type:* integer. Key The region consumption is being recorded for. Foreign key to the Region table. If NULL, consumption is for devices not belonging to any region

Database Column	Details
PVUSubcapacity	*Type:* integer The number of PVU points consumed for the license in the region.
ValidFrom	*Type:* datetime. Key The start time of the PVU usage
ValidTo	*Type:* datetime The end time of the PVU usage
LicenseMeasurementID	*Type:* integer. Key The snapshot ID. Foreign key to the LicenseMeasurement table.

SoftwareLicensePVUPointsPeakConsumedData Table

SoftwareLicensePVUPointsPeakConsumedData records current PVU consumption per host and region for the current reporting period.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 457: Database columns for SoftwareLicensePVUPointsPeakConsumedData table

Database Column	Details
RegionID	*Type:* integer. Key The region consumption is being recorded for. Foreign key to the Region table.
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
ComplianceComputerID	*Type:* integer. Key The license being assessed. Foreign key to the ComplianceComputer table.
IsEligibleInventory	*Type:* boolean Is the device eligible for inventory at consumption calculation
PVUConsumption	*Type:* integer The PVU points consumed for the host and license in the region at the time of the peak.
CalculatedByILMT	*Type:* boolean Is the consumption imported from ILMT?

Database Column	Details
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.

SoftwareLicensePVURegionPeakCalculatedData Table

SoftwareLicensePVURegionPeakCalculatedData records when the region specific peak was consumed for a region on an IBM PVU type license.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 458: Database columns for SoftwareLicensePVURegionPeakCalculatedData table

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key The license being assessed. Foreign key to the SoftwareLicense table.
RegionID	*Type:* integer. Key The region being assessed. Foreign key to the Region table.
PeakStartDate	*Type:* datetime The date when the latest peak consumption from this license was first recorded for devices in this region.
PeakDate	*Type:* datetime The date when the region's peak consumption from this license was recorded. (For peaks that persist, this is the date of the most recent license reconcile.)
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.

SoftwareLicensePartitioningDefault Table

SoftwareLicensePartitioningDefault contains the sub-capacity licensing rules: the types of virtual machines/partitions and pools which each license type treats as "hard" (able to put a hard limit on processor usage).

Table 459: Database columns for `SoftwareLicensePartitioningDefault` table

Database Column	Details
`SoftwareLicenseTypeID`	*Type:* integer. Key The software license type to which this rule applies. Foreign key to the `SoftwareLicenseType` table.
`VMTypeID`	*Type:* integer. Key. Nullable A virtual machine/partition type that is “hard” for the purposes of this license type. Foreign key to the `VMType` table.
`VMPoolTypeID`	*Type:* integer. Key. Nullable A virtual machine/partition pool type that is “hard” for the purposes of this license type. Foreign key to the `VMPoolType` table.

SoftwareLicensePoints Table

The `SoftwareLicensePoints` table holds the criteria for points-based licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 460: Database columns for `SoftwareLicensePoints` table

Database Column	Details
`SoftwareLicensePointsID`	*Type:* integer. Key. Generated ID A unique identifier for a software license criterion.
`SoftwareLicenseID`	*Type:* integer. Key The license to which this information applies. Foreign key to the `SoftwareLicense` table.
`ProcessorType`	*Type:* text (max 256 characters). Key The type of processor a computer must have for this criterion to apply, such as “AMD” or “Intel”.
`ComputerModelNo`	*Type:* text (max 128 characters). Key The model number a computer must have for this criterion to apply, such as “IBM PS701” or “IBM JS12”.
`MinCores`	*Type:* integer. Key The minimum number of processor cores a computer must have for this criterion to apply.

Database Column	Details
`MaxCores`	*Type:* integer. Key The maximum number of processor cores a computer must have for this criterion to apply.
`MinProcessors`	*Type:* integer. Key The minimum number of processors a computer must have for this criterion to apply.
`MaxProcessors`	*Type:* integer. Key The maximum number of processors a computer must have for this criterion to apply.
`MinSockets`	*Type:* integer. Key The minimum number of processor sockets a computer must have for this criterion to apply.
`MaxSockets`	*Type:* integer. Key The maximum number of processor sockets a computer must have for this criterion to apply.
`MinCoresPerSocket`	*Type:* integer. Key The minimum number of processor cores per socket a computer must have for this criterion to apply.
`MaxCoresPerSocket`	*Type:* integer. Key The maximum number of processor cores per socket a computer must have for this criterion to apply.
`Points`	*Type:* decimal The points value per core or processor.

SoftwareLicensePointsConsumedData Table

`SoftwareLicensePointsConsumed` records how many license entitlements have been consumed for a given license by a given computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 461: Database columns for `SoftwareLicensePointsConsumedData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.
`LicensesConsumed`	*Type:* integer The number of entitlements (or points) consumed for the license on the computer.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations actually being used.
`LicenseMeasurementID`	*Type:* integer. Key The license measurement ID. Foreign key to the `LicenseMeasurement` table.

SoftwareLicensePointsConsumedReasonData Table

This table stores information about why an entry in `SoftwareLicensePointsConsumed` exists.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 462: Database columns for `SoftwareLicensePointsConsumedReasonData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputer` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicense` table.

Database Column	Details
ReasonTypeID	*Type:* integer The reason for the points to be consumed here. Foreign key to the SoftwareLicensePointsConsumedReasonType table.
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.

SoftwareLicensePointsConsumedReasonType Table

SoftwareLicensePointsConsumedReasonType stores all the different important attributes that can be stored against a SoftwareLicensePointsConsumed record.

Table 463: Database columns for SoftwareLicensePointsConsumedReasonType table

Database Column	Details
ReasonTypeID	*Type:* integer. Key. Generated ID A unique identifer for the SoftwareLicensePointsConsumedReasonType table.
ReasonResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the reason a license was linked to a title. Foreign key to the ComplianceResourceString table.
ReasonDefaultValue	*Type:* text (max 100 characters) The text to display if the reason resource string has no translation.

SoftwareLicensePointsDefault Table

The SoftwareLicensePointsDefault table stores a collection of default license points associated with a particular license type.

Table 464: Database columns for SoftwareLicensePointsDefault table

Database Column	Details
SoftwareLicensePoints DefaultID	*Type:* integer. Key. Generated ID A unique identifier for a default points record.
SoftwareLicenseTypeID	*Type:* integer. Key The software license type to which this points record applies. Foreign key to the SoftwareLicenseType table.

Database Column	Details
`ProcessorType`	*Type:* text (max 256 characters) The type of processor a computer must have for this criterion to apply, such as “AMD” or “Intel”.
`ComputerModelNo`	*Type:* text (max 128 characters) The model number a computer must have for this criterion to apply, such aas “IBM PS701” or “IBM JS12”.
`MinCores`	*Type:* integer The minimum number of processor cores a computer must have for this criterion to apply.
`MaxCores`	*Type:* integer The maximum number of processor cores a computer must have for this criterion to apply.
`MinProcessors`	*Type:* integer The minimum number of processors a computer must have for this criterion to apply.
`MaxProcessors`	*Type:* integer The maximum number of processors a computer must have for this criterion to apply.
`MinSockets`	*Type:* integer The minimum number of processor sockets a computer must have for this criterion to apply.
`MaxSockets`	*Type:* integer The maximum number of processor sockets a computer must have for this criterion to apply.
`MinCoresPerSocket`	*Type:* integer The minimum number of processor cores per socket a computer must have for this criterion to apply.
`MaxCoresPerSocket`	*Type:* integer The maximum number of processor cores per socket a computer must have for this criterion to apply.
`Points`	*Type:* decimal The points value per core or processor.
`DateEffective`	*Type:* datetime. Nullable The date from which these default values are effective. This is used to group sets of rows into sets.

Database Column	Details
`Description`	*Type:* text (max 1024 characters). Nullable A description of the points rules.

SoftwareLicensePointsRule Table

The `SoftwareLicensePointsRule` table stores individual license points rules (mapping of criteria to point value) belonging to a given points rule set.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 465: Database columns for `SoftwareLicensePointsRule` table

Database Column	Details
`SoftwareLicensePoints RuleID`	*Type:* integer. Key. Generated ID A unique identifier for a points rule record.
`SoftwareLicensePoints RuleSetID`	*Type:* integer. Key The set to which this rule applies. Foreign key to the `SoftwareLicensePointsRuleSet` table.
`IsCustom`	*Type:* boolean. Key True if the rule is managed by the customer (versus by the ARL).
`Description`	*Type:* text (max 1024 characters). Nullable A human-readable description or identifier for the rule.
`Points`	*Type:* decimal. Key The points value per core, processor, user, or other resource metric.
`ProcessorType`	*Type:* text (max 256 characters) The type of processor a computer must have for this criterion to apply, such as “AMD” or “Intel”.
`ComputerModelNo`	*Type:* text (max 128 characters) The model number a computer must have for this criterion to apply, such as “IBM PS701” or “IBM JS12”.
`MinCores`	*Type:* integer The minimum number of processor cores a computer must have for this criterion to apply.

Database Column	Details
`MaxCores`	*Type:* integer The maximum number of processor cores a computer must have for this criterion to apply.
`MinProcessors`	*Type:* integer The minimum number of processors a computer must have for this criterion to apply.
`MaxProcessors`	*Type:* integer The maximum number of processors a computer must have for this criterion to apply.
`MinSockets`	*Type:* integer The minimum number of processor sockets a computer must have for this criterion to apply.
`MaxSockets`	*Type:* integer The maximum number of processor sockets a computer must have for this criterion to apply.
`MinCoresPerSocket`	*Type:* integer The minimum number of processor cores per socket a computer must have for this criterion to apply.
`MaxCoresPerSocket`	*Type:* integer The maximum number of processor cores per socket a computer must have for this criterion to apply.
`MinResource`	*Type:* decimal. Key The minimum resource value for an IBM RVU license for this criterion to apply.
`MaxResource`	*Type:* decimal. Key The maximum resource value for an IBM RVU license for this criterion to apply.
`MinUsers`	*Type:* integer. Key The minimum number of users relevant to an IBM UVU license for this criterion to apply.
`MaxUsers`	*Type:* integer. Key The maximum number of users relevant to an IBM UVU license for this criterion to apply.
`MinClockSpeed`	*Type:* integer The minimum value of the highest frequency of fastest processor, measured in megaherz, for this criterion to apply.

Database Column	Details
`MaxClockSpeed`	*Type:* integer The maximum value of the highest frequency of fastest processor, measured in megaherz, for this criterion to apply.
`MinPurchaseDate`	*Type:* datetime. Nullable The earliest date on which the asset must have been purchased for this criterion to apply.
`MaxPurchaseDate`	*Type:* datetime. Nullable The latest date on which the asset must have been purchased for this criterion to apply.
`CoverOnPremise`	*Type:* boolean If the value `True`, the points rule created can be used on-premises devices
`CoverAllCloudProviders`	*Type:* boolean If the value `True`, the points rule created can be used on all cloud service providers. If the value is `False`, list of cloud providers could be found from `SoftwareLicensePointsRuleCloudServiceProvider` table
`IsShared`	*Type:* boolean

SoftwareLicensePointsRuleCloudServiceProvider Table

`SoftwareLicensePointsRuleCloudServiceProvider` contains the cloud service providers supported by points rule.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 466: Database columns for `SoftwareLicensePointsRuleCloudServiceProvider` table

Database Column	Details
`SoftwareLicensePoints RuleCloudService ProviderID`	*Type:* integer. Key. Generated ID A unique identifier for a points rule and a cloud service provider.
`SoftwareLicensePoints RuleID`	*Type:* integer. Key Identifies a software license. Foreign key to the `SoftwareLicensePointsRule` table.

Database Column	Details
`CloudServiceProviderID`	*Type:* integer. Key Identifies a cloud service provider. Foreign key to the `CloudServiceProvider` table.
`IsShared`	*Type:* boolean

SoftwareLicensePointsRuleSet Table

The `SoftwareLicensePointsRuleSet` table stores named sets of points rules associated with a particular license type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 467: Database columns for `SoftwareLicensePointsRuleSet` table

Database Column	Details
`SoftwareLicensePoints RuleSetID`	*Type:* integer. Key. Generated ID A unique identifier for a points rule set record.
`SoftwareLicenseTypeID`	*Type:* integer. Key The software license type to which this set applies. Foreign key to the `SoftwareLicenseType` table.
`Description`	*Type:* text (max 256 characters). Key. Nullable A human-readable description or identifier for the set.
`IsShared`	*Type:* boolean

SoftwareLicenseProcessorPointsData Table

Stores the number of processors/cores on which points-based licensed software is installed and used, and the corresponding points and factors.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 468: Database columns for `SoftwareLicenseProcessorPointsData` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer under examination. Foreign key to the `ComplianceComputerSnapshot` table.
`SoftwareLicenseID`	*Type:* integer. Key The license being assessed. Foreign key to the `SoftwareLicenseSnapshot` table.
`InstalledCount`	*Type:* decimal The number of processors/cores on which a software title licensed by the license is installed.
`UsedCount`	*Type:* decimal The number of processors/cores on which a software title licensed by the license is used.
`CapacityCount`	*Type:* decimal The number of processors/cores that apply to a software title licensed by the license under full capacity counting rules.
`PointsFactor`	*Type:* decimal The number of points consumed per processor/core on this computer for this license.
`InstalledPoints`	*Type:* integer The number of processor/core points required to cover the above InstalledCount.
`UsedPoints`	*Type:* integer The number of processor/core points required to cover the above UsedCount.
`CapacityPoints`	*Type:* integer The number of processor/core points required to cover the above CapacityCount.
`LicenseMeasurementID`	*Type:* integer. Key The license measurement ID. Foreign key to the `LicenseMeasurement` table.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.
`Overridden`	*Type:* boolean Whether this consumption value was the result of an override.

SoftwareLicenseProduct Table

`SoftwareLicenseProduct` contains the set of `SoftwareTitleProducts` covered by a `SoftwareLicense`. Their product specific use rights on this license are also covered.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 469: Database columns for `SoftwareLicenseProduct` table

Database Column	Details
`SoftwareLicenseProductID`	*Type:* integer. Key. Generated ID A unique identifier for a software license product.
`SoftwareLicenseID`	*Type:* integer. Key License whose property value is being stored. Foreign key to the `SoftwareLicense` table
`SoftwareTitleProductID`	*Type:* integer. Key. Nullable The product covered by this license. Foreign key to the `SoftwareTitleProduct` table.
`CurrentSoftwareTitleID`	*Type:* integer. Nullable Identifies the current application for this product on this license (which may change over time as upgrade rights are applied). Foreign key to the `SoftwareTitle` table.
`Supplementary`	*Type:* boolean Whether this product on this license is supplementary (counted for consumption) or not.
`MeasuredForCompliance`	*Type:* boolean Whether this product on this license is measured for compliance risks.
`ProductRatio`	*Type:* integer If this product is supplementary on the license, the number of entitlements consumed related to the entitlements consumed for the parent product.
`ParentProductRatio`	*Type:* integer If this product is supplementary on the license, the number of entitlements consumed related to the entitlements consumed for the supplementary product.
`InheritDowngradeFrom Contract`	*Type:* boolean Set this field to `True` if this license inherits its downgrade rights from a contract. If `False` (the default), downgrade rights must be configured directly on the license properties.

Database Column	Details
`InheritDowngradeFromContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that downgrade rights are inherited from. Foreign key to the `Contract` table.
`InheritUpgradeFromContract`	*Type:* boolean Set this field to `True` if this license inherits its upgrade rights from a contract. If `False` (the default), upgrade rights must be configured directly on the license properties.
`InheritUpgradeFromContractID`	*Type:* integer. Nullable If the previous field is `True`, this is the contract that downgrade rights are inherited from. Foreign key to the `Contract` table.
`DowngradeEnabled`	*Type:* boolean If this field is `True`, this license can cover previous releases, or lower editions, of applications linked to this license. If this field is `False` (the default), there is no downgrade right conferred by this license.
`DowngradeToVersion`	*Type:* boolean If this field is `True`, the license covers previous releases (with the same edition) of the primary application. If this field is `False` (the default), earlier versions of the primary application are not covered by downgrade rights.
`DowngradeToVersionID`	*Type:* integer. Nullable If the previous field is `True` and the value of this field is NULL, downgrade rights cover all earlier releases (with the same edition) of the primary application. If not NULL, downgrade rights cover all versions of the primary application down to and including this version. Foreign key to the `SoftwareTitleVersion` table.
`DowngradeToEdition`	*Type:* boolean If this field is `True`, the license covers lower editions (with the same version) of the primary application. If this field is `False` (the default), lower editions of the primary application are not covered by downgrade rights.
`DowngradeToEditionID`	*Type:* integer. Nullable If the previous field is `True` and the value of this field is NULL, downgrade rights cover all lower editions (with the same version) of the primary application. If not NULL, downgrade rights cover all editions of the primary application down to and including this edition. Foreign key to the `SoftwareTitleEdition` table.
`UpgradeEnabled`	*Type:* boolean If this field is `True`, the license can cover future releases (with the same edition) of the primary application. If this bit is `False` (the default), there is no upgrade right conferred by this license.

Database Column	Details
UpgradeToVersion	*Type:* boolean If this field is True, the license covers later releases (with the same edition) of the primary application. If this field is False (the default), later versions of the primary application are not covered by upgrade rights.
UpgradeToVersionID	*Type:* integer. Nullable If the previous field is True and the value of this field is NULL, upgrade rights cover all later version (with the same edition) of the primary application. If not NULL, upgrade rights cover all versions of the primary application up to and including this version. Foreign key to the SoftwareTitleEdition table.
UpgradeUntil	*Type:* boolean If this bit is 1, the upgrade right covers future releases of applications that get linked to this license, provided that the release date of each version is before (or on) a specified date. If this bit is zero (the default), the upgrade right is not date limited.
UpgradeUntilDate	*Type:* datetime. Nullable If this field is set, only applications released before this date are covered by upgrade rights.

SoftwareLicensePropertyValue Table

For each end-user, SoftwareLicensePropertyValue stores the values for the custom properties defined in SoftwareLicenseTypeProperty.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 470: Database columns for SoftwareLicensePropertyValue table

Database Column	Details
SoftwareLicense PropertyValueID	*Type:* integer. Key. Generated ID A unique identifier for a property value.
SoftwareLicenseType PropertyID	*Type:* integer. Key The property whose value is being stored. The type of the license should match the type that the property is associated with. Foreign key to the SoftwareLicenseTypeProperty table.

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key License whose property value is being stored. Foreign key to the SoftwareLicense table
PropertyValue	*Type:* text (max 4000 characters) The property value.
CreationUser	*Type:* text (max 128 characters). Nullable The operator who created the record.
CreationDate	*Type:* datetime The date the record was created.
UpdatedUser	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
UpdatedDate	*Type:* datetime. Nullable The date the record was last updated.

SoftwareLicenseProposalStatus Table

SoftwareLicenseProposalStatus is a static table listing all of the states that a license change proposal can be in.

Table 471: Database columns for SoftwareLicenseProposalStatus table

Database Column	Details
SoftwareLicense ProposalStatusID	*Type:* integer. Key. Generated ID A unique identifier for each SoftwareLicenseProposalStatus. Possible values and the corresponding default strings are: • 1 = Pending • 2 = Accepted • 3 = Ignored
ResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the license change proposal state. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

SoftwareLicensePurchaseType Table

`SoftwareLicensePurchaseType` holds a list of purchase types for licenses.

Table 472: Database columns for `SoftwareLicensePurchaseType` table

Database Column	Details
`SoftwareLicense PurchaseTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicensePurchaseType`. Possible values and the corresponding default strings are: • 1 = Volume • 2 = Shrink Wrap • 3 = OEM • 4 = Subscription.
`SoftwareLicense PurchaseTypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a purchase type. Foreign key to the `ComplianceResourceString` table.
`SoftwareLicense PurchaseTypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

SoftwareLicenseReservation Table

The `SoftwareLicenseReservation` table lists all reservations for a license entitlement for an application.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 473: Database columns for `SoftwareLicenseReservation` table

Database Column	Details
`SoftwareLicense ReservationID`	*Type:* integer. Key. Generated ID A unique identifier for this reservation.
`SoftwareTitleID`	*Type:* integer. Key The application being reserved. Foreign key to the `SoftwareTitle` table.

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key. Nullable The license affected by this reservation, null if any license for the application can be consumed. Foreign key to the SoftwareLicense table.
ComplianceComputerID	*Type:* integer. Key The computer making the reservation. Foreign key to the ComplianceComputer table.
ComplianceUserID	*Type:* integer. Key. Nullable The user making the reservation. Foreign key to the ComplianceUser table.
PointsReserved	*Type:* integer The number of points this reservation will ultimately consume.
CreationUser	*Type:* text (max 128 characters). Nullable The operator who created the record.
CreationDate	*Type:* datetime The date the record was created.
SoftwareLicense ReservationTypeID	*Type:* integer The type of reservation.
SoftwareLicense ReservationStatusID	*Type:* integer Stores the status of the reservation

SoftwareLicenseReservationNecessityCheckResult Table

The SoftwareLicenseReservationNecessityCheckResult table saves the results of the necessity to do software license reservation for a license entitlement for an application.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 474: Database columns for SoftwareLicenseReservationNecessityCheckResult table

Database Column	Details
SoftwareLicense ReservationNecessity CheckResultID	*Type:* integer. Key. Generated ID A unique identifier for this reservation necessity check result.

Database Column	Details
SoftwareTitleID	*Type:* integer. Key The application being reserved. Foreign key to the SoftwareTitle table.
SoftwareLicenseID	*Type:* integer. Key. Nullable The license affected by this reservation, null if any license for the application can be consumed. Foreign key to the SoftwareLicense table.
ComplianceComputerID	*Type:* integer. Key The computer making the reservation. Foreign key to the ComplianceComputer table.
ComplianceUserID	*Type:* integer. Key. Nullable The user making the reservation. Foreign key to the ComplianceUser table.
HasSecondUseRight	*Type:* boolean No reservation is actually needed because of second use right.
HasUpgradeDowngradeRight	*Type:* boolean No reservation is actually needed because of upgrade/downgrade right
HasExemptionByDeviceRole	*Type:* boolean No reservation is actually needed because of exemption by device role.
CreationUser	*Type:* text (max 128 characters). Nullable The operator who created the record.
CreationDate	*Type:* datetime The date the record was created.

SoftwareLicenseReservationStatus Table

The collection of status values for reservation.

Table 475: Database columns for SoftwareLicenseReservationStatus table

Database Column	Details
SoftwareLicense ReservationStatusID	*Type:* integer. Key. Generated ID A unique identifier for the reservation status id
SoftwareLicense ReservationStatusName	*Type:* text (max 128 characters) The name of the reservation status.

SoftwareLicenseReservationType Table

The collection of status values for reservation types.

Table 476: Database columns for `SoftwareLicenseReservationType` table

Database Column	Details
`SoftwareLicenseReservationTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the reservation type id
`SoftwareLicenseReservationTypeName`	*Type:* text (max 128 characters) The name of the reservation type.

SoftwareLicenseScopeTag Table

Reserved for future development.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 477: Database columns for `SoftwareLicenseScopeTag` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key Foreign key to the `SoftwareLicense` table.
`TagID`	*Type:* integer. Key Foreign key to the `Tag` table.
`ScopeTagTypeID`	*Type:* integer. Key Foreign key to the `SoftwareLicenseScopeTagType` table.

SoftwareLicenseScopeTagType Table

Reserved for future development.

Table 478: Database columns for `SoftwareLicenseScopeTagType` table

Database Column	Details
`ScopeTagTypeID`	*Type:* integer. Key. Generated ID A unique ID for this record.

Database Column	Details
`TypeDescription`	*Type:* text (max 50 characters). Key The text value for this type.

SoftwareLicenseScoping Table

`SoftwareLicenseScoping` links software licenses to enterprise groups, to restrict the rights granted by the licenses to the selected group and its descendents (license scoping).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 479: Database columns for `SoftwareLicenseScoping` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The scoped license. Foreign key to the `SoftwareLicense` table.
`GroupExID`	*Type:* text (max 128 characters). Key The enterprise group that this license is restricted to. Any children of this enterprise group are also included in the scope of the license. Foreign key to the `GroupEx` table.

SoftwareLicenseSecondUseMappingData Table

`SoftwareLicenseSecondUseMapping` maps pairs of desktop computers and laptop computers against each license conferring the right of second use and covering installations on these computers.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 480: Database columns for `SoftwareLicenseSecondUseMappingData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The license conferring the right of second use. Foreign key to the `SoftwareLicense` table.

Database Column	Details
DesktopComputerID	*Type:* integer. Key The desktop or primary computer on which the related software in installed. Foreign key to the ComplianceComputer table.
SecondUseComputerID	*Type:* integer. Key The laptop or second computer covered by this license's right of second use, relative to the installation on the primary computer tracked in the previous field. Foreign key to the ComplianceComputer table.
TotalLicenseGrabs	*Type:* integer For internal use only. Temporary storage for calculations of overlapping second use and multiple install rights.
IsExternalRoamingLink	*Type:* boolean Is this a second use link or is it actually an 'external roaming' right?

SoftwareLicenseSnapshot Table

The SoftwareLicenseSnapshot table lists all the snapshotted software licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 481: Database columns for SoftwareLicenseSnapshot table

Database Column	Details
SoftwareLicenseID	*Type:* integer. Key The snapshotted SoftwareLicenseID.
Name	*Type:* text (max 256 characters) The snapshotted license name.
LicenseTypeID	*Type:* integer. Key The license type. Foreign key to the SoftwareLicenseType table.
SoftwareLicense ComplianceStatusID	*Type:* integer. Nullable The compliance status of this license. Foreign key to the SoftwareLicenseComplianceStatus table. Defaults to "Compliant".
Consumed	*Type:* integer. Nullable The snapshotted license consumed count.

Database Column	Details
`PurchaseQuantity`	*Type:* integer. Nullable The snapshotted license purchase quantity.
`PurchasePrice`	*Type:* currency. Nullable The initial purchase price of the license.
`PurchasePriceRateID`	*Type:* integer. Nullable The currency rate applied to the purchase price of the license. Foreign key to the `CurrencyRate` table.
`LicenseMeasurementID`	*Type:* integer. Key The snapshot ID. Foreign key to the `LicenseMeasurement` table.
`NumberUsed`	*Type:* integer. Nullable The snapshotted license number used count.
`LastCalculatedNUPMinimum`	*Type:* integer. Nullable The snapshotted license last calculated minimum for `Oracle Named User Plus` licenses.
`CalculatedConsumed`	*Type:* integer The calculated consumption value for this license.

SoftwareLicenseTierType Table

`SoftwareLicenseTierType` is a static table listing the tier types that a software license can have. Used for `Tiered Device` license type.

Table 482: Database columns for `SoftwareLicenseTierType` table

Database Column	Details
`SoftwareLicenseTierTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseTierType`. Possible values and the corresponding default strings are: • 1 = Generic • 2 = Per Processor • 3 = Symantec Server • 4 = Symantec Processor Type • 5 = Symantec Installed Operating System.

Database Column	Details
`TierTypeResourceName`	*Type:* text (max 256 characters). Key. Nullable The unique name of the localizable resource string representing a tier type. Foreign key to the `ComplianceResourceString` table.
`TierTypeDefaultValue`	*Type:* text (max 256 characters) The text to display if the type resource string has no translation.
`TierCodeValidationRegEx`	*Type:* text (max 256 characters). Nullable The regular expression used to validate the tier code.
`TierCodeValidationMsg ResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing the message shown when tier code validation fails. Foreign key to the `ComplianceResourceString` table.
`TierCodeValidationMsg DefaultValue`	*Type:* text (max 256 characters). Nullable The text to display if the resource string (for the message shown when tier code validation fails) has no translation.

SoftwareLicenseType Table

`SoftwareLicenseType` holds the collection of all valid license types.

Table 483: Database columns for `SoftwareLicenseType` table

Database Column	Details
`SoftwareLicenseTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseType`. Possible values and the corresponding default strings are: • 1 = Enterprise • 2 = Device • 3 = Node-Locked • 4 = User • 5 = Concurrent User • 6 = Appliance • 7 = Client Server • 8 = OEM • 9 = Evaluation • 10 = Run-Time • 11 = Device (Processor-Limited) • 12 = Site • 13 = Named User • 14 = Device (Core-Limited) • 15 = Core Points • 16 = Oracle Processor • 17 = Oracle Named User Plus • 18 = Processor Points • 19 = Oracle Legacy • 20 = Enterprise Agreement • 21 = SAP Named User • 22 = Microsoft Server Processor • 23 = CAL Legacy • 24 = Tiered Device • 25 = IBM Processor Value Unit • 26 = IBM Authorized User

Database Column	Details
	• 27 = IBM Concurrent User • 28 = IBM Floating User • 29 = Custom Metric • 30 = Processor • 31 = IBM Resource Value Unit • 32 = IBM User Value Unit • 33 = Microsoft Server Core • 34 = Oracle User • 35 = SAP Package • 36 = Microsoft SCCM Client Device • 37 = Microsoft SCCM Client User • 38 = Microsoft Developer Network • 39 = Microsoft Device CAL • 40 = Microsoft User CAL • 41 = Microsoft Server/Management Core • 42 = SaaS User
`TypeResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a license type. Foreign key to the `ComplianceResourceString` table.
`TypeDefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`XMLFile`	*Type:* text. Nullable The layout of the property dialog for this type of computer, stored in XML format.
`CustomProcedureName`	*Type:* text (max 256 characters). Nullable The stored procedure used to assign licenses for this license type.
`DoesLicenseAllowUser Allocations`	*Type:* boolean Set this field to `True` if the license supports allocations to individual end-users. When `False`, it cannot be allocated to end-users.

Database Column	Details
DoesLicenseAllow ComputerAllocations	*Type:* boolean Set this field to True if the license supports allocations to individual computers. When it is False, it cannot be allocated to computers. (Note that for a custom license type, both this and the previous field may be set at the same time.)
DoesLicenseAllow VirtualApplications	*Type:* boolean Set this field to True if the license supports virtual applications. When it is False, it cannot be consumed by virtual applications. (Note that virtual applications have AccessModeID > 1.)
CanConvertToAndFromType	*Type:* boolean Set this field to True if an operator is allowed to change the type of this license after it has been created. This field also determines whether this license type is included in the list of types that can be converted to. Oracle licenses, for example, cannot be converted to or from.
ExclusionReasonName	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing the reason why an installation linked to a license of this type may appear in the Unlicensed Installs node. Foreign key to the ComplianceResourceString table.
ExclusionReasonDefault	*Type:* text (max 500 characters) The text to display if the reason resource string has no translation.
IncludeInSQLAssignment	*Type:* boolean Set this field to True if licenses of this type should be processed during the SQL part of the license reconciliation process.
CalculateCompliance	*Type:* boolean When this field is True (the default), and a SoftwareLicense of this type also has its CalculateCompliance field set to True (the default), that license must have its consumption calculated from imported inventory. When False, the compliance state of licenses with this type must be imported or otherwise set manually, not calculated.
ReconcileAsSoftware LicenseTypeID	*Type:* integer. Nullable If specified, treat this license type as if it were another for license reconciliation purposes. Foreign key to another type in this SoftwareLicenseType table.
Enabled	*Type:* boolean Indicates whether this license type is enabled
CoverOnPremise	*Type:* boolean If the value True, the license created for this type will have on-premise selected in the use rights.

Database Column	Details
CoverAllCloudProviders	*Type:* boolean If the value True, the license created for this type will have all cloud service provider selected in the use rights. If the value is False, the specific cloud provider values could be found from SoftwareLicenseTypeCloudServiceProviderDefault table

SoftwareLicenseTypeChangeProposal Table

The SoftwareLicenseTypeChangeProposal table is used to store a proposed change of type for a particular software license. The changes have been inferred from changes to the license definition used to create the software license.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 484: Database columns for SoftwareLicenseTypeChangeProposal table

Database Column	Details
SoftwareLicenseTypeChangeProposalID	*Type:* integer. Key. Generated ID Primary key for the SoftwareLicenseTypeChangeProposal table.
SoftwareLicenseID	*Type:* integer. Key Foreign key to the SoftwareLicense table.
SoftwareLicenseDefinitionID	*Type:* integer. Key Foreign key to the SoftwareLicenseDefinition table.
LicenseDefinitionVersion	*Type:* integer The version of the license definition that has been used for these proposed changes.
SoftwareLicenseUseRightNameID	*Type:* integer The proposed use right being changed on the software license.
SoftwareLicenseTypeID	*Type:* integer. Key The proposed license type for the software license.
OldSoftwareLicenseTypeID	*Type:* integer The existing license type for the software license.
SoftwareLicenseProposalStatusID	*Type:* integer The state of this software license change proposal.

Database Column	Details
`Conflicted`	*Type:* boolean Whether this license type change proposal conflicts with another type proposed for the same license.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was updated.

SoftwareLicenseTypeCloudServiceProviderDefault Table

The `SoftwareLicenseTypeCloudServiceProvider` table stores default cloud service providers supported by a software license type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 485: Database columns for `SoftwareLicenseTypeCloudServiceProviderDefault` table

Database Column	Details
`SoftwareLicenseTypeID`	*Type:* integer. Key Identifies a software license type. Foreign key to the `SoftwareLicenseType` table
`CloudServiceProviderID`	*Type:* integer. Key Identifies a cloud service provider. Foreign key to the `CloudServiceProvider` table.
`IsShared`	*Type:* boolean

SoftwareLicenseTypePriority Table

`SoftwareLicenseTypePriority` holds the priority order of license types.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 486: Database columns for `SoftwareLicenseTypePriority` table

Database Column	Details
`SoftwareLicenseTypeID`	*Type:* integer. Key The software license type to which this priority applies. Foreign key to the `SoftwareLicenseType` table.
`CompliancePriority`	*Type:* integer The priority order of the license type when calculating compliance. Licenses with higher priority will be consumed first.

SoftwareLicenseTypeProperty Table

`SoftwareLicenseTypeProperty` defines extra custom properties for all end-users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 487: Database columns for `SoftwareLicenseTypeProperty` table

Database Column	Details
`SoftwareLicenseTypePropertyID`	*Type:* integer. Key. Generated ID A unique identifier for an individual property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.
`SoftwareLicenseTypeID`	*Type:* integer. Key License type with which this property is associated. Foreign key to the `LicenseType` table.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

SoftwareLicenseUseRight Table

`SoftwareLicenseUseRight` contains licensing rules most of which can be set by PURL.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 488: Database columns for `SoftwareLicenseUseRight` table

Database Column	Details
`SoftwareLicenseUseRightID`	*Type:* integer. Key. Generated ID A unique identifier
`SoftwareLicenseID`	*Type:* integer. Key A unique identifier for a software license.
`ReassignmentTimeLimitApplies`	*Type:* boolean If 1 then the license cannot be reassigned for some period of time (example is Microsoft 90 day rule)
`ReassignmentTimeLimit`	*Type:* integer. Nullable The period (in days) within which the license cannot be reassigned
`LicenseMobilityApplies`	*Type:* boolean 1 if eligible for bringing your own license to cloud environment
`NumberOfOSEPerLicense`	*Type:* integer. Nullable Number of OSE per license
`NumberOfProcessorsPerOSE`	*Type:* integer. Nullable Number of processors per OSE
`TotalNumberOfCoresPerVMPerLicense`	*Type:* integer. Nullable Total number of cores per VM per license
`NumberOfCoresPerSocket`	*Type:* integer. Nullable Number of cores per socket
`ThirdPartyAccessAllowed`	*Type:* boolean Access to applications is allowed to third party users. This field is defaulted to `True`
`PURLComment`	*Type:* text. Nullable Additional information provided by PURL

Database Column	Details
`AllowExternalRoamingUse`	*Type:* boolean. Nullable Set this field to `True` if license allows external roaming use. This field is defaulted to `False`. This is applicable for both device and user licenses and is related to virtual application access. If 1, this license will consume 1 entitlement per each user. If 0, this license will consume 1 license per each user device. And, if NULL, ignore virtual application access. This can be used in conjunction with VirtualApplicationAccessMaximumUsagePeriod.
`MeasurementDate`	*Type:* datetime. Nullable The date of the license measurment.
`ConsumptionUnit`	*Type:* text. Nullable Unit description to describe the consumption amount.
`TargetOperatingSystem TypeID`	*Type:* integer Type of Operating Systems to target
`VirtualApplication AccessMaximumUsagePeriod`	*Type:* integer. Nullable This is a rule for virtual application access. This is used in conjunction with the AllowExternalRoamingUse. For Device licenses, a license will consume 1 entitlement per each user device when used in period specified here. For user licenses, if 1, this license will consume only when used in period specified here.
`ExemptCALs`	*Type:* boolean If the value `True`, clients accessing servers that consume this license will be exempted from CAL (no CALs required). Only applicable to Microsoft Server Processor, Microsoft Server Core and Device license types.
`CoverOnPremise`	*Type:* boolean If the value `True`, the license is eligible for use on on-premise platform.
`CoverAllCloudProviders`	*Type:* boolean If the value `True`, the license is eligible for use on all cloud platforms. If the value is `False`, the specific cloud provider values could be found from `SoftwareLicenseCloudServiceProvider` table.
`MinimumCoresPerProcessor`	*Type:* integer. Nullable Specifies minimum number of cores need to be licensed per processor.
`MinimumCoresPerHost`	*Type:* integer. Nullable Specifies minimum number of cores need to be licensed per server.

SoftwareLicenseUseRightIBM Table

`SoftwareLicenseUseRightIBM` contains IBM licensing rules most of which can be set by PURL.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 489: Database columns for `SoftwareLicenseUseRightIBM` table

Database Column	Details
`SoftwareLicenseUseRightIBMID`	*Type:* integer. Key. Generated ID A unique identifier
`SoftwareLicenseID`	*Type:* integer. Key A unique identifier for a software license.
`PVULimitApplies`	*Type:* boolean If 1 then PVU limits apply
`PVULimit`	*Type:* integer. Nullable PVU limit

SoftwareLicenseUseRightName Table

`SoftwareLicenseUseRightName` is a static table listing all of the use rights that can be applied to a software license.

Table 490: Database columns for `SoftwareLicenseUseRightName` table

Database Column	Details
`SoftwareLicenseUse RightNameID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareLicenseUseRightName`. Possible values and the corresponding default strings are: • 1 = License type • 2 = Cover installs on virtual machines • 3 = Limit number of virtual installs • 4 = Number of allowed virtual installs • 5 = Limit virtual installs includes host • 6 = Use host processor information • 7 = Allow IBM PVU sub-capacity from non ILMT • 8 = Limit number of applications each license point covers • 9 = Number of application installs allowed per license point • 10 = Limit number of computers user license can be installed on • 11 = Number of computers allowed per license point • 12 = Minimum number of users • 13 = Minimum number of users multiplied by processors • 14 = Second usage work laptop • 15 = Second usage at home • 16 = Downgrade enabled • 17 = Downgrade to version • 18 = Downgrade to version ID • 19 = Downgrade to edition • 20 = Downgrade to edition ID • 21 = Upgrade enabled • 22 = Upgrade to version • 23 = Upgrade to version ID • 24 = Upgrade until • 25 = Upgrade until date • 26 = Reassignment time limit applies

Database Column	Details
	• 27 = Reassignment time limit • 28 = License mobility applies • 29 = Number of OSE per license • 30 = Number of processors per OSE • 31 = Total number of cores per VM per license • 32 = Number of cores per socket • 33 = Third party access allowed • 34 = PURL comment • 35 = Allow external roaming use • 36 = Measurement date • 37 = Consumption unit • 38 = PVU limit applies • 39 = PVU limit • 40 = Points rule set • 41 = Minimum number of processors • 42 = Minimum number of licenses per virtual machine • 43 = Number of sockets • 44 = User multiplier external • 45 = User multiplier infrequent • 46 = Exempted roles • 47 = Exempted role limit • 48 = Measure for compliance • 49 = Ratio from primary • 50 = Ratio to primary • 51 = Exempt CALs • 52 = Minimum cores per processor • 53 = Minimum cores per host

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the proposed action. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

SoftwareLicenseUseRightProposal Table

The `SoftwareLicenseUseRightProposal` table is used to store a summary of use right changes to a particular software license. The changes have been inferred from changes to the license definition used to create the software license.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 491: Database columns for `SoftwareLicenseUseRightProposal` table

Database Column	Details
`SoftwareLicenseUse RightProposalID`	*Type:* integer. Key. Generated ID Primary key for the `SoftwareLicenseUseRightProposal` table.
`SoftwareLicenseID`	*Type:* integer. Key Foreign key to the `SoftwareLicense` table.
`SoftwareLicense DefinitionID`	*Type:* integer. Key Foreign key to the `SoftwareLicenseDefinition` table.
`LicenseDefinitionVersion`	*Type:* integer The version of the license definition that has been used for these proposed changes.
`SoftwareLicenseUse RightNameID`	*Type:* integer. Key The proposed use right being changed on the software license.
`SoftwareTitleProductID`	*Type:* integer. Key. Nullable Foreign key to the `SoftwareTitleProduct` table.
`Enabled`	*Type:* boolean. Key. Nullable Is this use right being enabled?

Database Column	Details
`MD5Value`	*Type:* text (max 32 characters). Key. Nullable The MD5 of Value in Hex.
`Value`	*Type:* text (max 4000 characters). Nullable The proposed value for this use right.
`OldValue`	*Type:* text (max 4000 characters). Nullable The existing value for this use right.
`RelatedID`	*Type:* integer. Nullable The database ID of the proposed object associated with this use right.
`OldRelatedID`	*Type:* integer. Nullable The database ID of the old object associated with this use right.
`SoftwareLicense ProposalStatusID`	*Type:* integer The state of this software license change proposal.
`Conflicted`	*Type:* boolean Whether this license type change proposal conflicts with another type proposed for the same license.
`ContractInherited`	*Type:* boolean Whether this license type change proposal is for a use right currently inherited from contract by the license.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was updated.

SoftwareLifeCycle Table

Table 492: Database columns for `SoftwareLifeCycle` table

Database Column	Details
`SoftwareLifeCycleID`	*Type:* integer. Key. Generated ID A unique identifier for an ARL published software life cycle.
`SoftwareLifeCycleUID`	*Type:* text (max 64 characters). Key Factory generated identifier
`Name`	*Type:* text (max 350 characters) Name of the software life cycle published by software publisher
`AvailabilityDate`	*Type:* datetime. Nullable The availability date.
`EndOfAvailabilityDate`	*Type:* datetime. Nullable The end of availability date.

SoftwareRecognition Table

Table 493: Database columns for `SoftwareRecognition` table

Database Column	Details
`SoftwareRecognitionID`	*Type:* text (max 30 characters). Key Factory generated identity.
`UpdateMode`	*Type:* text (max 20 characters). Nullable Update behavior.
`LastCollectiveUpdated`	*Type:* datetime. Nullable Last updated datetime by ARL on all software titles and evidence
`LastLinkUpdated`	*Type:* datetime. Nullable Last updated datetime by ARL on the software title links
`LastRecordUpdated`	*Type:* datetime. Nullable Last updated datetime by ARL on the software title or evidence records. To know which record this column refers to, see TypeOfID.
`LastCollectiveChecksum`	*Type:* integer. Nullable Last collective checksum on successful ARL update

Database Column	Details
`LastLinkChecksum`	*Type:* integer. Nullable Last link checksum on successful ARL update
`LastRecordChecksum`	*Type:* integer. Nullable Last record checksum on successful ARL update. To know which record this column refers to, see TypeOfID.
`LastCollectiveUpdateResult`	*Type:* integer. Nullable Last collective ARL update result
`LastLinkUpdateResult`	*Type:* integer. Nullable Last ARL link update result
`LastRecordUpdateResult`	*Type:* integer. Nullable Last ARL record update result
`RecordAdoptedByARL`	*Type:* boolean When an existing customer record is updated by the ARL, this flag will be set
`SoftwareTitleID`	*Type:* integer. Key. Nullable The related `SoftwareTitle`
`ChildSoftwareTitleID`	*Type:* integer. Key. Nullable The related child `SoftwareTitle`
`SoftwareTitleProductID`	*Type:* integer. Key. Nullable The related `SoftwareTitleProduct`
`SoftwareTitleVersionID`	*Type:* integer. Key. Nullable The related `SoftwareTitleVersion`
`SoftwareTitleEditionID`	*Type:* integer. Key. Nullable The related `SoftwareTitleEdition`
`SoftwareTitlePublisherID`	*Type:* integer. Key. Nullable The related `SoftwareTitlePublisher`
`FileEvidenceID`	*Type:* integer. Key. Nullable The related `FileEvidence`
`InstallerEvidenceID`	*Type:* integer. Key. Nullable The related `InstallerEvidence`
`WMIEvidenceID`	*Type:* integer. Key. Nullable The related `WMIEvidence`

Database Column	Details
AccessEvidenceID	*Type:* integer. Key. Nullable The related AccessEvidence
RegistryEvidenceID	*Type:* integer. Nullable The related registry WMIEvidence
SoftwareLicensePoints DefaultID	*Type:* integer. Key. Nullable The related SoftwareLicensePointsDefault
SoftwareLicensePoints RuleSetID	*Type:* integer. Key. Nullable The related SoftwareLicensePointsRuleSet
SoftwareLicensePoints RuleID	*Type:* integer. Key. Nullable The related SoftwareLicensePointsRule
TypeOfID	*Type:* text (max 32 characters). Key The type of the last updated ARL record

SoftwareSKULookup Table

SoftwareSKULookup maps licenses imported from external source to SKU published by FNMS

Table 494: Database columns for SoftwareSKULookup table

Database Column	Details
SoftwareSKULookupID	*Type:* integer. Key. Generated ID A unique identifier for this record.
SourceType	*Type:* text (max 32 characters). Key
LookupName	*Type:* text (max 128 characters). Key
SKU	*Type:* text (max 100 characters) Holds the SKU value.

SoftwareSku Table

SoftwareSku defines all software SKU (stock-keeping unit) numbers.

Table 495: Database columns for `SoftwareSku` table

Database Column	Details
`SoftwareSkuID`	*Type:* integer. Key. Generated ID A unique identifier for a software SKU.
`SKUFactoryUID`	*Type:* text (max 30 characters). Key A FlexNet Manager Suite factory unique ID for this SKU.
`SKU`	*Type:* text (max 100 characters). Key Holds the SKU value.
`SKUDefinition`	*Type:* text Encrypted data that describes this SKU.
`SoftwareLicense DefinitionID`	*Type:* integer. Key SKU license definition. Used to create new licenses and link them to applications. Foreign key to the `SoftwareLicenseDefinition` table.
`SoftwareSkuTypeID`	*Type:* integer. Key For internal use only. A numerical representation of the type of SKU.
`MaintenanceTypeID`	*Type:* integer For internal use only. A numerical representation of the maintenance type (if any) of the SKU.
`Version`	*Type:* integer. Key The current version of the SKU definition.
`PreviousVersion`	*Type:* integer. Key. Nullable The version of the SKU definition prior to the current version.
`CreationDate`	*Type:* datetime The date that this SKU definition was created.
`UpdatedDate`	*Type:* datetime. Nullable The date that this SKU definition was last updated.
`SKUPrefixLength`	*Type:* integer. Key. Nullable The location of the % wildcard.

SoftwareTitle Table

The `SoftwareTitle` table contains the application titles managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 496: Database columns for `SoftwareTitle` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key. Generated ID A unique identifier for a software record.
`SoftwareTitleTypeID`	*Type:* integer. Key The application type. Foreign key to the `SoftwareTitleType` table.
`SoftwareTitleProductID`	*Type:* integer. Key The application product, which also may specify a publisher. Foreign key to the `SoftwareTitleProduct` table.
`SoftwareTitleVersionID`	*Type:* integer. Key. Nullable The application version. Foreign key to the `SoftwareTitleVersion` table.
`SoftwareTitleEditionID`	*Type:* integer. Key. Nullable The application edition. Foreign key to the `SoftwareTitleEdition` table.
`OperatorManageStateID`	*Type:* integer. Key The management responsibility for this information. Foreign key to the `OperatorManageState` table.
`FullName`	*Type:* text (max 512 characters) By default, the full name of the application is the concatenation of the product, version, and edition fields. The operator may overwrite this with any preferred value.
`SoftwareTitle ClassificationID`	*Type:* integer. Nullable The classification of the title. Defaults to `None`. Foreign key to the `SoftwareTitleClassification` table.
`IsMonitoringSessions`	*Type:* boolean Set this field to `True` if sessions are being monitored.
`UsageSessions`	*Type:* integer An application is considered used if it is opened more than this many times within the monitoring period.
`IsMonitoringActiveTime`	*Type:* boolean Set this field to `True` if active time is being monitored.

Database Column	Details
`UsageActiveTime`	*Type:* integer An application is considered used if the application active time (time it is in the foreground) exceeds this value during the monitoring period.
`UsagePeriod`	*Type:* integer The period in months over which to consider usage.
`Comments`	*Type:* text. Nullable Stores any comments an operator wants to make about a particular application title.
`SKU`	*Type:* text (max 200 characters). Nullable Deprecated: now use `LicensePartNo` of the `PurchaseOrderDetail` table. Stock Keeping Unit (SKU) for the application.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise category associated with this application title. Foreign key to the `GroupEx` table.
`IsLicensable`	*Type:* boolean Set this field to `True` if this application needs a license. If `False`, the application doesn't need a license.
`ReleaseDate`	*Type:* datetime. Nullable The date the application was released.
`IsSharableToLibrary`	*Type:* boolean Set this field to `True` if the application is sharable to the FlexNet Manager Suite ARL library.
`AutoManageLicensePriority`	*Type:* boolean Set this field to `True` if the application should automatically manage the priority of attached licenses.
`TitleRequiresStrict Matching`	*Type:* boolean Set this field to `True` if the application should use stricter matching rules, requiring all evidence of all types to be present.
`SupportedUntil`	*Type:* datetime. Nullable The date the application will be supported
`ExtendedSupportUntil`	*Type:* datetime. Nullable The date the application will be supported, in extended case
`StartOfLifeDate`	*Type:* datetime. Nullable Start of life Date

Database Column	Details
`EndOfSalesDate`	*Type:* datetime. Nullable End of sales Date
`EndOfLifeDate`	*Type:* datetime. Nullable End of life Date
`SoftwareTitleActionID`	*Type:* integer A categorization for the application in the enterprise. Defaults to `New`.Foreign key to the `SoftwareTitleAction` table.
`HasInstalls`	*Type:* boolean If this field is `True` this application has at least one installation. If `False`, the application has no installations.
`SoftwareLifeCycleID`	*Type:* integer. Key. Nullable Foreign key to the `SoftwareLifeCycle` table.
`HasCustomEndOfSupportLife`	*Type:* boolean. Nullable Set this field to indicate custom end of support life for this application.
`CustomEndOfSupportLife Date`	*Type:* datetime. Nullable Custom end of support life date.
`IsShared`	*Type:* boolean

SoftwareTitleAccessEvidence Table

`SoftwareTitleAccessEvidence` links software (application) titles to access evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 497: Database columns for `SoftwareTitleAccessEvidence` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The software title to which the access evidence is related. Foreign key to the `SoftwareTitle` table.
`AccessEvidenceID`	*Type:* integer. Key The access evidence related to the software title. Foreign key to the `AccessEvidence` table.

Database Column	Details
IsLocal	*Type:* boolean If this field is False, the link has come from the ARL. If it is True, then the link has been created by an operator.
IsShared	*Type:* boolean

SoftwareTitleAction Table

SoftwareTitleAction is a static table listing action outcomes for the application in the enterprise.

Table 498: Database columns for SoftwareTitleAction table

Database Column	Details
SoftwareTitleActionID	*Type:* integer. Key. Generated ID A unique identifier for each SoftwareTitleAction. Possible values and the corresponding default strings are: • 1 = Unmanaged (recently created application, not yet categorized) • 2 = Authorized (application is authorized for use in the enterprise) • 3 = Unauthorized (application is not authorized for use) • 4 = Ignored (application will not be tracked by the enterprise) • 5 = Inactive (application is not in use in the enterprise). • 6 = Deferred (application installed in enterprise but marked for later attention).
ActionResourceName	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an action outcome. Foreign key to the ComplianceResourceString table.
ActionDefaultValue	*Type:* text (max 100 characters) The text to display if the action outcome resource string has no translation.

SoftwareTitleClassification Table

SoftwareTitleClassification is a static table listing the possible classifications for software titles.

Table 499: Database columns for `SoftwareTitleClassification` table

Database Column	Details
`SoftwareTitle ClassificationID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareTitleClassification`. Possible values and the corresponding default strings are: • 1 = Shareware • 2 = Freeware • 3 = Commercial • 4 = Update • 5 = Malware • 6 = Beta • 7 = XRated • 8 = None • 9 = Component.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an application classification. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the classification resource string has no translation.

SoftwareTitleEOSL Table

The `SoftwareTitleEOSL` table stores attributes of an application EOSL dates. These will only be populated if the FNMS for EOSL is present.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 500: Database columns for `SoftwareTitleEOSL` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The EOSL dates. Foreign key to the `SoftwareTitle` table.

Database Column	Details
`StartOfLifeDate`	*Type:* datetime. Nullable Start of life Date
`ReleaseDate`	*Type:* datetime. Nullable The date the application was released.
`EndOfSalesDate`	*Type:* datetime. Nullable End of sales Date
`SupportedUntil`	*Type:* datetime. Nullable The date the application will be supported
`ExtendedSupportUntil`	*Type:* datetime. Nullable The date the application will be supported, in extended case
`EndOfLifeDate`	*Type:* datetime. Nullable End of life Date
`IsShared`	*Type:* boolean

SoftwareTitleEdition Table

A list of application editions, which must be unique for a given product. Examples include “Ultimate”, “Professional” and “32 bit”.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 501: Database columns for `SoftwareTitleEdition` table

Database Column	Details
`SoftwareTitleEditionID`	*Type:* integer. Key. Generated ID The unique identifier for an edition.
`SoftwareTitleProductID`	*Type:* integer. Key The edition’s product. Foreign key to the `SoftwareTitleProduct` table.
`EditionName`	*Type:* text (max 50 characters). Key The text for this application edition.
`EditionWeight`	*Type:* decimal Edition weight (for ordering, so we know which editions are upgrades/downgrades of other editions).

Database Column	Details
IsLocal	*Type:* boolean If this field is False, the edition has come from the ARL. If it is True, then the edition has been created by an operator.
IsShared	*Type:* boolean

SoftwareTitleEx Table

The SoftwareTitleEx table contains additional information on the application titles managed by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 502: Database columns for SoftwareTitleEx table

Database Column	Details
SoftwareTitleID	*Type:* integer. Key A unique identifier for a software record.
OperatorManageStateID	*Type:* integer. Nullable The management responsibility for this information. Foreign key to the OperatorManageState table.
AutoManageLicensePriority	*Type:* boolean. Nullable Set this field to True if the application should automatically manage the priority of attached licenses.
IsMonitoringSessions	*Type:* boolean. Nullable Set this field to True if sessions are being monitored.
UsageSessions	*Type:* integer. Nullable An application is considered used if it is opened more than this many times within the monitoring period.
IsMonitoringActiveTime	*Type:* boolean. Nullable Set this field to True if active time is being monitored.
UsageActiveTime	*Type:* integer. Nullable An application is considered used if the application active time (time it is in the foreground) exceeds this value during the monitoring period.

Database Column	Details
UsagePeriod	*Type:* integer. Nullable The period in months over which to consider usage.
SoftwareTitleActionID	*Type:* integer. Key. Nullable A categorization for the application in the enterprise. Defaults to New.Foreign key to the SoftwareTitleAction table.
HasInstalls	*Type:* boolean. Nullable If this field is True this application has at least one installation. If False, the application has no installations.
HasCustomEndOfSupportLife	*Type:* boolean. Nullable Set this field to indicate custom end of support life for this application.
CustomEndOfSupportLifeDate	*Type:* datetime. Nullable Custom end of support life date.

SoftwareTitleFileEvidence Table

SoftwareTitleFileEvidence links software (application) titles to file evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 503: Database columns for SoftwareTitleFileEvidence table

Database Column	Details
SoftwareTitleID	*Type:* integer. Key The application title to which the file evidence is related. Foreign key to the SoftwareTitle table.
FileEvidenceID	*Type:* integer. Key The file evidence related to the software title. Foreign key to the FileEvidence table.
EvidenceExistenceRuleID	*Type:* integer The evidence existence rule related to the software title. Foreign key to the EvidenceExistenceRule table.

Database Column	Details
TrackUsage	*Type:* boolean If this field is True, the linked file evidence should be considered when calculating whether the application title is being used. If False, the file is not tracked for usage calculations.
IsLocal	*Type:* boolean If this field is False, the link has come from the ARL. If it is True, then the link has been created by an operator.
IsShared	*Type:* boolean

SoftwareTitleHierarchy Table

SoftwareTitleHierarchy records a hierarchy of applications. This table records relationships between Oracle database and component applications, between suites and their members, and between generic titles and more specific ones that will replace them.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 504: Database columns for SoftwareTitleHierarchy table

Database Column	Details
ParentSoftwareTitleID	*Type:* integer. Key The parent application. Foreign key to the SoftwareTitle table.
ChildSoftwareTitleID	*Type:* integer. Key The child application. Foreign key to the SoftwareTitle table.
IsLicensable	*Type:* boolean. Nullable This field is used for Oracle option titles. Set this field to True to indicate that the child application needs to be separately licensed. If this field is False, the child application does not need to be separately licensed when the parent application is present and licensed.
IsMandatory	*Type:* boolean. Nullable This field is used on component applications of software suites. When the value is True, the child application must be installed for the suite to be recognized as installed. Otherwise, the application may or may not be installed for the suite to be recognized.

Database Column	Details
`RemovalOfChild`	*Type:* boolean. Nullable This field is used to allow removal of titles when higher quality titles (with more specified evidence) are also found installed. When the value is `True`, the child application should be removed if evidence is found that both it and its parent title are installed. Otherwise, the child application is left in place.
`IsLocal`	*Type:* boolean If this field is `False`, the link has come from the ARL. If it is `True`, then the link has been created by an operator.
`IsMandatoryDefault`	*Type:* boolean. Nullable This field is used on component applications of software suites. This indicates the Default value of the Mandatory field and can be used to determine if this has been overridden by the user, in the case of an application with non-local membership to the suite (that is, the ARL specifies that the app belongs to the suite).
`IsShared`	*Type:* boolean

SoftwareTitleHierarchyEx Table

The `SoftwareTitleHierarchyEx` table contains additional information on the suite by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 505: Database columns for `SoftwareTitleHierarchyEx` table

Database Column	Details
`ParentSoftwareTitleID`	*Type:* integer. Key The parent application. Foreign key to the `SoftwareTitle` table.
`ChildSoftwareTitleID`	*Type:* integer. Key The child application. Foreign key to the `SoftwareTitle` table.
`IsMandatory`	*Type:* boolean. Nullable This field is used on component applications of software suites. When the value is `True`, the child application must be installed for the suite to be recognized as installed. Otherwise, the application may or may not be installed for the suite to be recognized.

SoftwareTitleInstallerEvidence Table

`SoftwareTitleInstallerEvidence` links software (application) titles to installer evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 506: Database columns for `SoftwareTitleInstallerEvidence` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The software title to which the installer evidence is related. Foreign key to the `SoftwareTitle` table.
`InstallerEvidenceID`	*Type:* integer. Key The installer evidence related to the software title. Foreign key to the `InstallerEvidence` table.
`IsLocal`	*Type:* boolean If this field is `False`, the link has come from the ARL. If it is `True`, then the link has been created by an operator.
`IsShared`	*Type:* boolean

SoftwareTitleLicense Table

The `SoftwareTitleLicense` table links software (application) titles to licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 507: Database columns for `SoftwareTitleLicense` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The application. Foreign key to the `SoftwareTitle` table.
`SoftwareLicenseID`	*Type:* integer. Key The license covering this application. Foreign key to the `SoftwareLicense` table.

Database Column	Details
`CompliancePriority`	*Type:* integer. Nullable Installations of this application will consume the linked licenses in this table in order of priority. When NULL, the default priority stored in `SoftwareLicenseType` table will be used.
`LicenseKeyValue`	*Type:* text (max 400 characters). Nullable The license (installation) key value to be used when this license covers an installation of this application.
`SoftwareTitleLicense ReasonID`	*Type:* integer The reason that this application has been added to this license. Foreign key to the `SoftwareTitleLicenseReason` table.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.

SoftwareTitleLicenseProposal Table

The `SoftwareTitleLicenseProposal` table is used to store a summary of application changes to a particular software license. The changes have been inferred from changes to the license definition used to create the software license.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 508: Database columns for `SoftwareTitleLicenseProposal` table

Database Column	Details
`SoftwareTitleLicense ProposalID`	*Type:* integer. Key. Generated ID Primary key for the `SoftwareTitleLicenseProposal` table.
`SoftwareTitleID`	*Type:* integer. Key Foreign key to the `SoftwareTitle` table.
`SoftwareTitleProductID`	*Type:* integer. Key. Nullable Foreign key to the `SoftwareTitleProduct` table.

Database Column	Details
`OldPrimarySoftwareTitleID`	*Type:* integer. Nullable The existing primary application of the license. This can be null if there is no primary application.
`Supplementary`	*Type:* boolean Whether this product will be added to this license as supplementary (counted for consumption) or not.
`SoftwareLicenseID`	*Type:* integer. Key Foreign key to the `SoftwareLicense` table.
`SoftwareLicense DefinitionID`	*Type:* integer. Key Foreign key to the `SoftwareLicenseDefinition` table.
`LicenseDefinitionVersion`	*Type:* integer The version of the license definition that has been used for these proposed changes.
`SoftwareTitleLicense ProposalActionID`	*Type:* integer. Key The proposed action for the software title on the software license.
`SoftwareLicense ProposalStatusID`	*Type:* integer The state of this software license change proposal.
`Conflicted`	*Type:* boolean Whether this license title change proposal conflicts with another for the same license.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was updated.

SoftwareTitleLicenseProposalAction Table

`SoftwareTitleLicenseProposalAction` is a static table listing all of the actions that can be proposed for a software title on a software license.

Table 509: Database columns for `SoftwareTitleLicenseProposalAction` table

Database Column	Details
`SoftwareTitleLicenseProposalActionID`	*Type:* integer. Key. Generated ID
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the SoftwareTitleLicenseProposalAction record. Foreign key to the ComplianceResourceString table.
`DefaultValue`	*Type:* text (max 256 characters) The text to display if the state resource string has no translation.

SoftwareTitleLicenseReason Table

`SoftwareTitleLicenseReason` is a static table listing valid reasons why a software title was added to a license.

Table 510: Database columns for `SoftwareTitleLicenseReason` table

Database Column	Details
`SoftwareTitleLicenseReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `SoftwareTitleLicenseReason`. Possible values and the corresponding default strings are: • 1 = Manual • 2 = Current • 3 = Edition Downgrade • 4 = Version Downgrade • 5 = Version Upgrade.
`ReasonResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing the reason a license was linked to a title. Foreign key to the `ComplianceResourceString` table.
`ReasonDefaultValue`	*Type:* text (max 100 characters) The text to display if the reason resource string has no translation.

SoftwareTitleOracle Table

The `SoftwareTitleOracle` table stores attributes of an application installation that are relevant to Oracle

applications only. These characteristics are important for Oracle licensing.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 511: Database columns for `SoftwareTitleOracle` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The Oracle application. Foreign key to the `SoftwareTitle` table.
`MaximumSockets`	*Type:* integer. Nullable The maximum number of sockets allowed on a computer where the application is installed.
`NUPProcessorMultiplier`	*Type:* integer. Nullable The multipler value to use when determining the minumum Named User Plus licenses for the application.
`OverrideSoftwareTitleTypeID`	*Type:* integer. Nullable If this is not null, then the application was initially created as non-Oracle, but the operator wants to license it as an Oracle title. Foreign key to the `SoftwareTitleType` table.
`IsShared`	*Type:* boolean

SoftwareTitleProduct Table

The “product”, unique for a given publisher, is the common name of a set of applications, independent of version or edition (for example, “Acrobat”).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 512: Database columns for `SoftwareTitleProduct` table

Database Column	Details
`SoftwareTitleProductID`	*Type:* integer. Key. Generated ID The unique identifier for a product.
`SoftwareTitlePublisherID`	*Type:* integer. Key. Nullable The publisher of this product. Foreign key to the `SoftwareTitlePublisher` table.

Database Column	Details
`ProductName`	*Type:* text (max 200 characters). Key The application's product name.
`IsLocal`	*Type:* boolean If this field is `False`, the product has come from the ARL. If it is `True`, then the product has been created by an operator.
`IsShared`	*Type:* boolean

SoftwareTitleProperty Table

`SoftwareTitleProperty` defines extra custom properties for all applications.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 513: Database columns for `SoftwareTitleProperty` table

Database Column	Details
`SoftwareTitlePropertyID`	*Type:* integer. Key. Generated ID The unique identifier for a software title property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the property.
`CustomPropertyDisplayXMLID`	*Type:* integer. Nullable Foreign key to a record in the `CustomPropertyDisplayXML` table, describing how to show the property on a property dialog.

SoftwareTitlePropertyValue Table

For each application, `SoftwareTitlePropertyValue` stores the values for the custom properties defined in `SoftwareTitleProperty`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 514: Database columns for `SoftwareTitlePropertyValue` table

Database Column	Details
`SoftwareTitleProperty ValueID`	*Type:* integer. Key. Generated ID A unique identifier for a property value.
`SoftwareTitleID`	*Type:* integer. Key The title for which the property is being stored. Foreign key to the `SoftwareTitle` table.
`SoftwareTitlePropertyID`	*Type:* integer. Key The property whose value is being stored. Foreign key to the `SoftwareTitleProperty` table.
`PropertyValue`	*Type:* text (max 4000 characters) The property value.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

SoftwareTitlePublisher Table

Publishers of software applications (for example, “Microsoft”). Note that only application records take the publisher name from this table. License and contract records take the publisher name from the `Vendor` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 515: Database columns for `SoftwareTitlePublisher` table

Database Column	Details
`SoftwareTitlePublisherID`	*Type:* integer. Key. Generated ID The unique identifier for a publisher.

Database Column	Details
`PublisherName`	*Type:* text (max 200 characters). Key The publisher name.
`IsLocal`	*Type:* boolean If this field is `False`, the publisher has come from the ARL. If it is `True`, then the publisher has been created by an operator.
`EOSLUrl`	*Type:* text (max 2083 characters). Nullable The publisher's end of support life URL.
`IsShared`	*Type:* boolean

SoftwareTitleRegistryEvidence Table

Reserved for future use.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 516: Database columns for `SoftwareTitleRegistryEvidence` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The software title to which the registry evidence is related. Foreign key to the `SoftwareTitle` table.
`RegistryEvidenceID`	*Type:* integer. Key The registry evidence related to the software title. Foreign key to the `RegistryEvidence` table.
`IsShared`	*Type:* boolean

SoftwareTitleSuite Table

For software that has been classed as a suite (because it has other applications linking to it as component applications), `SoftwareTitleSuite` identifies how many of its member applications must be present for the installation to count as a suite installation, using "application evidence" for suite recognition.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 517: Database columns for `SoftwareTitleSuite` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The suite. Foreign key to the `SoftwareTitle` table.
`MinNumberApps`	*Type:* integer The minimum number of member applications of the software suite that must be installed.
`MinNumberAppsDefault`	*Type:* integer. Nullable The original, default value of `MinNumberApps` before it was changed.
`IsShared`	*Type:* boolean

SoftwareTitleSuiteEx Table

The `SoftwareTitleSuiteEx` table contains additional information on the suite by FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 518: Database columns for `SoftwareTitleSuiteEx` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key A unique identifier for a software record.
`MinNumberApps`	*Type:* integer. Nullable The minimum number of member applications of the software suite that must be installed.

SoftwareTitleType Table

`SoftwareTitleType` is a static table listing possible types of software (application) titles. This is used particularly to identify types that need special processing. It is quite distinct from license types.

Table 519: Database columns for `SoftwareTitleType` table

Database Column	Details
`SoftwareTitleTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a `SoftwareTitleType`. Possible values and the corresponding default strings are: • 1 = General • 2 = Oracle Database • 3 = Oracle Option • 4 = Oracle Application • 5 = Oracle EBS Server • 6 = Oracle EBS.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a document type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.
`InstanceTypeID`	*Type:* integer The type of instance that can be created for this application. Foreign key to the `InstanceType` table.

SoftwareTitleVersion Table

A list of application versions, which must be unique for a given product. Examples include “6.4”, “XP”, “Vista” and “2003”.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 520: Database columns for `SoftwareTitleVersion` table

Database Column	Details
`SoftwareTitleVersionID`	*Type:* integer. Key. Generated ID The unique identifier for a version.

Database Column	Details
SoftwareTitleProductID	*Type:* integer. Key The version's product. Foreign key to the SoftwareTitleProduct table.
VersionName	*Type:* text (max 50 characters). Key The text for this application version.
VersionWeight	*Type:* decimal Version weight (for ordering, so we know which versions are upgrades/downgrades of other versions).
IsLocal	*Type:* boolean If this field is False, the version has come from the ARL. If it is True, then the version has been created by an operator.
IsShared	*Type:* boolean

SoftwareTitleVersionServicePack Table

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 521: Database columns for SoftwareTitleVersionServicePack table

Database Column	Details
SoftwareTitleVersionID	*Type:* integer. Key The application version. Foreign key to the SoftwareTitleVersion table.
ServicePackID	*Type:* integer. Key The service pack. Foreign key to the ServicePack table.

SoftwareTitleWMIEvidence Table

SoftwareTitleWMIEvidence links software titles to WMI evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 522: Database columns for `SoftwareTitleWMIEvidence` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key The software title to which the WMI evidence is related. Foreign key to the `SoftwareTitle` table.
`WMIEvidenceID`	*Type:* integer. Key The WMI evidence related to the software title. Foreign key to the `WMIEvidence` table.
`IsLocal`	*Type:* boolean If this field is `False`, the link has come from the ARL. If it is `True`, then the link has been created by an operator.
`IsShared`	*Type:* boolean

SoftwareUserLicensePointsConsumedData Table

`SoftwareUserLicensePointsConsumed` records how many software license entitlements have been consumed for a given license by a given end-user.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 523: Database columns for `SoftwareUserLicensePointsConsumedData` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key The end-user. Foreign key to the `ComplianceUserSnapshot` table.
`SoftwareLicenseID`	*Type:* integer. Key The license. Foreign key to the `SoftwareLicenseSnapshot` table.
`LicensesConsumed`	*Type:* integer The number of points (or entitlements) consumed for the license by the end-user.
`CalculatedConsumption`	*Type:* integer The calculated consumption value for this license assignment before exemptions or overrides are considered.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations that are actually being used.

Database Column	Details
LicenseMeasurementID	*Type:* integer. Key The license measurement ID. Foreign key to the LicenseMeasurement table.

SoftwareUserLicensePointsConsumedSuggested Table

SoftwareUserLicensePointsConsumedSuggested records how many software license entitlements would be consumed by an end-user for an optimized (suggested) license. Currently used to track optimized license usage suggested by FlexNet Manager for SAP.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 524: Database columns for SoftwareUserLicensePointsConsumedSuggested table

Database Column	Details
ComplianceUserID	*Type:* integer. Key The end-user. Foreign key to the ComplianceUser table.
SuggestedSoftwareLicenseID	*Type:* integer. Key The suggested or optimized license. Foreign key to the SoftwareLicense table.
LicensesConsumed	*Type:* integer The number of points (or entitlements) consumed for the license by the end-user.
LicensesUsed	*Type:* integer How many of the points consumed are for installations that are actually being used.
LicenseMeasurementID	*Type:* integer. Key The associated SAP license measurement snapshot. Foreign key to the LicenseMeasurement table.
LicensesCalculated	*Type:* integer The number of points (or entitlements) calculated for the license by the end-user.

SoftwareUserLicensePointsConsumedSuggestedHistory Table

SoftwareUserLicensePointsConsumedSuggestedHistory table records the history of suggested (optimised)

license consumption.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 525: Database columns for `SoftwareUserLicensePointsConsumedSuggestedHistory` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key The end-user. Foreign key to the `ComplianceUser` table.
`SuggestedSoftwareLicenseID`	*Type:* integer. Key The suggested or optimized license. Foreign key to the `SoftwareLicense` table.
`LicensesConsumed`	*Type:* integer The number of points (or entitlements) consumed for the license by the end-user.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations that are actually being used.
`LicenseMeasurementID`	*Type:* integer. Key The associated SAP license measurement snapshot. Foreign key to the `LicenseMeasurement` table.
`LicensesCalculated`	*Type:* integer The number of points (or entitlements) calculated for the license by the end-user.

SoftwareUserLicensePointsHistory Table

`SoftwareUserLicensePointsHistory` records history of license consumption by end-users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 526: Database columns for `SoftwareUserLicensePointsHistory` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key The end-user. Foreign key to the `ComplianceUser` table.

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key The license. Foreign key to the `SoftwareLicense` table.
`LicensesConsumed`	*Type:* integer The number of points (or entitlements) consumed for the license by an end-user.
`LicensesUsed`	*Type:* integer How many of the points consumed are for installations that are actually being used.
`LicenseMeasurementID`	*Type:* integer. Key The associated SAP license measurement snapshot. Foreign key to the `LicenseMeasurement` table.
`LicensesCalculated`	*Type:* integer The number of points (or entitlements) calculated for the license by the end-user.

Tag Table

Reserved for future development.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 527: Database columns for `Tag` table

Database Column	Details
`TagID`	*Type:* integer. Key. Generated ID The unique ID for this tag.
`Name`	*Type:* text (max 128 characters). Key The name of this tag.
`Description`	*Type:* text Description of this tag and its purpose.

TargetOperatingSystemType Table

TargetOperatingSystemType; is a static table listing all types of OSes that can be targeted by licensing.

Table 528: Database columns for `TargetOperatingSystemType` table

Database Column	Details
`TargetOperatingSystemTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `TargetOperatingSystemType`. Possible values and the corresponding default strings are: • 1 = All • 2 = Windows Server operating systems • 3 = Windows desktop operating systems • 4 = Non Windows Server operating systems
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an Operating System family. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

TemporalInstalledSoftwareSCD Table

`TemporalInstalledSoftwareSCD` is a table listing history of all the installations of an application (as defined in the `SoftwareTitle` table).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 529: Database columns for `TemporalInstalledSoftwareSCD` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The computer on which the software is installed. Foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key The software that is installed. Foreign key to the `SoftwareTitle` table.
`ValidFrom`	*Type:* datetime. Key Date from which these properties became valid.

Database Column	Details
`ValidTo`	*Type:* datetime. Key Date these properties were valid to, or 9999-12-31T23:59:59.997 if they are currently valid.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable LicenseID of license to which this installation has been assigned.

VDI Table

`VDI` is the list of VDI devices

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 530: Database columns for `VDI` table

Database Column	Details
`VDIID`	*Type:* integer. Key. Generated ID A unique identifier for a VDI device.
`ComputerName`	*Type:* text (max 256 characters). Key The computer name of the VDI.
`ComplianceDomainID`	*Type:* integer. Key. Nullable The domain the VDI is a member of. Foreign key to the `ComplianceDomain` table.
`VDIGroupID`	*Type:* integer. Key The VDI group the VDI device belongs to. Foreign key to the `VDIGroup` table.
`VDITemplateID`	*Type:* integer. Key The master VM template of the VDI. Foreign key to the `VDITemplate` table.
`RetiredDate`	*Type:* datetime. Key. Nullable The date the VDI device was deleted.
`ApplicationDeliveryOnly`	*Type:* boolean. Key Determines whether the VDI device is used only to server applications.

VDIEndPointAccess Table

`VDIEndPointAccess` is the list of endpoint devices that have accessed VDI devices

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 531: Database columns for `VDIEndPointAccess` table

Database Column	Details
`VDIEndPointAccessID`	*Type:* integer. Key. Generated ID A unique identifier for an endpoint device accessing a VDI.
`ComplianceComputerID`	*Type:* integer. Key. Nullable A unique identifier for the endpoint. Foreign key to the `ComplianceComputer` table.
`ComplianceUserID`	*Type:* integer. Key. Nullable A unique identifier for the endpoint user. Foreign key to the `ComplianceUser` table.
`VDIID`	*Type:* integer. Key A unique identifier for the VDI. Foreign key to the `VDI` table.
`LogonTime`	*Type:* datetime. Nullable The date the user logged on to the VDI.

VDIGroup Table

`VDIGroup` stores the list of available VDI groups in a VDI environment.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 532: Database columns for `VDIGroup` table

Database Column	Details
`VDIGroupID`	*Type:* integer. Key. Generated ID A unique identifier for a VDI Group record.
`GroupName`	*Type:* text (max 128 characters). Key The VDI Group name

Database Column	Details
VDISiteID	*Type:* integer. Key The VDI site ID
VDIGroupUUID	*Type:* unique identifier. Key. Nullable The UUID of the VDI group

VDISite Table

VDISite stores the list of available VDI sites.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 533: Database columns for VDISite table

Database Column	Details
VDISiteID	*Type:* integer. Key. Generated ID A unique identifier for a VDI site record.
SiteName	*Type:* text (max 256 characters). Key The VDI Group name
AccessModeID	*Type:* integer. Key The access mode of the VDI site. Foreign key to the AccessMode table.

VDITemplate Table

VDITemplate stores the list of available VDI groups in a VDI environment.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 534: Database columns for VDITemplate table

Database Column	Details
VDITemplateID	*Type:* integer. Key. Generated ID A unique identifier for a VDI Group record.

Database Column	Details
`TemplateName`	*Type:* text (max 256 characters). Key The VDI template name.
`VDISiteID`	*Type:* integer. Key. Nullable The VDI template's site ID
`ComplianceComputerID`	*Type:* integer. Key. Nullable The VDI template's ComplianceComputerID

VDIUser Table

`VDIUser` is the list of users that have access to VDI groups

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 535: Database columns for `VDIUser` table

Database Column	Details
`VDIUserID`	*Type:* integer. Key. Generated ID A unique identifier for a VDI User.
`VDIGroupID`	*Type:* integer. Key A unique identifier for the VDI Group. Foreign key to the `VDIGroup` table.
`ComplianceUserID`	*Type:* integer. Key. Nullable A unique identifier for the user with access to a VDI Group. Foreign key to the `ComplianceUser` table.

WMIEvidence Table

`WMIEvidence` lists WMI evidence that is used to identify that a particular item of software (defined in the `SoftwareTitle` table) has been installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 536: Database columns for `WMIEvidence` table

Database Column	Details
`WMIEvidenceID`	*Type:* integer. Key. Generated ID A unique identifier for a WMI evidence record.
`ClassName`	*Type:* text (max 50 characters). Key The WMI class name of the WMI evidence.
`PropertyName`	*Type:* text (max 50 characters). Key The WMI property name of the WMI evidence.
`PropertyValue`	*Type:* text (max 256 characters). Key The value of the WMI evidence property.
`Ignored`	*Type:* boolean Set this field to `True` if this WMI evidence is ignored for application recognition.
`IsShared`	*Type:* boolean

WMIEvidenceMatchCount Table

`WMIEvidenceMatchCount` tracks the number of times that each WMI evidence (rule) has been detected as installed and recorded in the data source. A separate count is kept for each WMI evidence rule, and for each data source.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 537: Database columns for `WMIEvidenceMatchCount` table

Database Column	Details
`WMIEvidenceMatchCountID`	*Type:* integer. Key. Generated ID A synthetic unique identifier is required, since `ComplianceConnectionID`, being nullable, cannot be included in the primary key.
`WMIEvidenceID`	*Type:* integer. Key WMI evidence rule being matched. Foreign key to the `WMIEvidence` table.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable Data source where the match is occurring. Foreign key to the `ComplianceConnection` table.

Database Column	Details
`MatchedCount`	*Type:* integer The number of installed WMI evidence records in this data source matching this WMI evidence rule.
`InstallCount`	*Type:* integer The number of physical application installations recognized in this data source using this WMI evidence rule.

Compliance.Logic.Structure Tables

The complete set of database tables documented here includes:

- ComplianceDomain table (see ComplianceDomain Table)
- GroupEx table (see GroupEx Table)
- GroupExPathCultureType table (see GroupExPathCultureType Table)
- GroupType table (see GroupType Table)
- MemberEx table (see MemberEx Table)
- Region table (see Region Table)
- RoleRight table (see RoleRight Table)

ComplianceDomain Table

Stores a list of domain names imported FlexNet Manager Suite.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 538: Database columns for `ComplianceDomain` table

Database Column	Details
`ComplianceDomainID`	*Type:* integer. Key. Generated ID A unique identifier for a domain.
`QualifiedName`	*Type:* text (max 100 characters) The fully qualified name of the domain.
`FlatName`	*Type:* text (max 32 characters) The flat name of the domain.

GroupEx Table

The `GroupEx` table stores information about enterprise groups and roles.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 539: Database columns for `GroupEx` table

Database Column	Details
`GroupID`	*Type:* integer. Key. Generated ID A unique identifier for a group.
`GroupTypeID`	*Type:* integer. Key Identifies the group type. Foreign key to the `GroupType` table.
`BusinessView`	*Type:* boolean. Key Set this to `True` if the group is a business view (that is, is a group heading like `Roles` or `Categories`.
`Path`	*Type:* text (max 500 characters) Complete path of the group.
`NextChild`	*Type:* integer The ID number for the next child to be created under this group. Internal use only: do not edit.
`GroupExID`	*Type:* text (max 128 characters). Key Unique string identifier for this extension record.
`BusinessPhoneNumber`	*Type:* text (max 30 characters). Nullable The business phone number of the group.
`FaxPhoneNumber`	*Type:* text (max 30 characters). Nullable The fax number of the group.
`Address_Street`	*Type:* text (max 200 characters). Nullable The street address of the group.
`Address_City`	*Type:* text (max 200 characters). Nullable The city of the group.
`Address_State`	*Type:* text (max 200 characters). Nullable The state of the group.

Database Column	Details
`Address_ZIP`	*Type:* text (max 20 characters). Nullable The ZIP or postal code of the group.
`Address_Country`	*Type:* text (max 100 characters). Nullable The country of the group.
`Email`	*Type:* text (max 200 characters). Nullable The email address of the group.
`Comments`	*Type:* text. Nullable Comments about the group.
`IsStockLocation`	*Type:* boolean For locations only. If this field is set to `True`, the location is considered to be a stock or storage location.
`ContactID`	*Type:* integer. Nullable A contact person for this group. This field is no longer in use in FlexNet Manager Suite
`ManagerID`	*Type:* integer. Nullable A manager for this group. This field is no longer in use in FlexNet Manager Suite
`GroupCN`	*Type:* text (max 256 characters). Nullable The common name for the group.
`NameResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing an enterprise group name (GroupCN). Foreign key to the `ComplianceResourceString` table.
`DescriptionResourceName`	*Type:* text (max 256 characters). Nullable The unique name of the localizable resource string representing an enterprise group description (Comments). Foreign key to the `ComplianceResourceString` table.
`ParentGroupExID`	*Type:* text (max 128 characters). Key. Nullable Unique string identifier for the parent record.
`TreeLevel`	*Type:* integer. Nullable The level of this group in the hierarchy.
`TreePath`	*Type:* text (max 4000 characters). Key. Nullable A generated path that can be used to sort groups in tree order.
`SpecifiedRegion`	*Type:* boolean RegionID specified explicitly by the user through the web UI.

Database Column	Details
`RegionID`	*Type:* integer. Key. Nullable RegionID for each location inherit from itself or parent.
`IsShared`	*Type:* boolean

GroupExPathCultureType Table

The `GroupExPathCultureType` table stores complete enterprise group paths per culture type for each enterprise group.

Table 540: Database columns for `GroupExPathCultureType` table

Database Column	Details
`GroupID`	*Type:* integer. Key The ID of the group the transalted path belongs to.
`CultureType`	*Type:* text (max 12 characters). Key A unique identifier for a culture type.
`Path`	*Type:* text (max 500 characters) The translated group path for the specific culture type.
`TreePath`	*Type:* text (max 4000 characters) A generated path that can be used to sort groups in tree order.

GroupType Table

The collection of types of enterprise groups, such as locations, departments, and cost centers.

Table 541: Database columns for `GroupType` table

Database Column	Details
`GroupTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `GroupType`. Possible values and the corresponding default strings are: • 1 = Location • 2 = Departments • 3 = Cost Center • 4 = Category • 5 = Role.
`Description`	*Type:* text (max 255 characters). Key A description of the type of enterprise group.
`ResourceName`	*Type:* text (max 256 characters). Key. Nullable The unique name of the localizable resource string representing a group type. Foreign key to the `ComplianceResourceString` table.

MemberEx Table

The `MemberEx` table stores the membership lists for every enterprise group or role.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 542: Database columns for `MemberEx` table

Database Column	Details
`GroupID`	*Type:* integer. Key The `GroupEx` to which the member belongs.

Database Column	Details
TargetTypeID	*Type:* integer. Key The TargetType. Possible values are: • 3 = Enterprise Group • 9 = Asset • 10 = Contract • 11 = Purchase Order • 12 = Software License • 13 = Software Title • 14 = Computer • 15 = User • 16 = Operator • 17 = SAP system landscapes • 18 = SAP systems • 19 = SAP rule sets
TargetID	*Type:* integer. Key The Asset, Contract, etc. identifier, depending on TargetType.

Region Table

The collection of region for IBM PVU license

Table 543: Database columns for Region table

Database Column	Details
RegionID	*Type:* integer. Key. Generated ID A unique identifier for each Region. Possible values and the corresponding default strings are: • 1 = North America and South America • 2 = Europe and Africa • 3 = Asia and Australia

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an end-user status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

RoleRight Table

Each action by FlexNet Manager Suite requires the role to have one or more `RoleRights` to perform an `ActionClass` over a given `Resource`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 544: Database columns for `RoleRight` table

Database Column	Details
`GroupID`	*Type:* integer. Key The role to whom the right is granted or denied.
`ResourceID`	*Type:* integer. Key The `Resource` to which the `RoleRight` applies.
`ActionClassID`	*Type:* integer. Key The action class which applies (read or modify).
`Denied`	*Type:* boolean When TRUE (1), indicates that the specified right is denied.
`ScopeGroupID`	*Type:* integer. Key. Nullable The enterprise group to which the right for this role applies, if applicable.

Compliance.Logic.Users Tables

The complete set of database tables documented here includes:

- ComplianceUser table (see ComplianceUser Table)
- ComplianceUserConnection table (see ComplianceUserConnection Table)
- ComplianceUserInventorySourceType table (see ComplianceUserInventorySourceType Table)

- ComplianceUserStatus table (see ComplianceUserStatus Table)
- EmploymentStatus table (see EmploymentStatus Table)
- UserSuffix table (see UserSuffix Table)
- UserTitle table (see UserTitle Table)

ComplianceUser Table

`ComplianceUser` stores information about end-users in the enterprise, including contact details, login details and inventory source details (if applicable). End-users in `ComplianceUser` will not be able to log in to FlexNet Manager Suite unless they have a corresponding record in the `ComplianceOperator` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 545: Database columns for `ComplianceUser` table

Database Column	Details
`ComplianceUserID`	*Type:* integer. Key. Generated ID A unique identifier for the end-user.
`UserName`	*Type:* text (max 512 characters). Nullable The end-user's full name. When creating a new end-user manually, defaults to a concatenation of title, first name, middle name, last name and suffix.
`SAMAccountName`	*Type:* text (max 64 characters). Key. Nullable The login name (SAM account name) of the end-user.
`ComplianceDomainID`	*Type:* integer. Key. Nullable Domain that the end-user belongs to. Foreign key to the `ComplianceDomain` table.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this end-user. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any corporate unit in the enterprise associated with this end-user. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any cost center in the enterprise associated with this end-user. Foreign key to the `GroupEx` table.

Database Column	Details
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable No longer in use, but retained for legacy systems. Any enterprise category associated with this end-user. Foreign key to the `GroupEx` table.
`EmployeeNumber`	*Type:* text (max 128 characters). Key. Nullable The employee number of the end-user (as defined in an organization's own HR system).
`UserTitleID`	*Type:* integer. Nullable The title of the end-user. Foreign key to the `UserTitle` table.
`FirstName`	*Type:* text (max 128 characters). Nullable The first name of the end-user.
`MiddleName`	*Type:* text (max 128 characters). Nullable The middle name(s) of the end-user.
`LastName`	*Type:* text (max 128 characters). Nullable The last name (surname) of the end-user.
`UserSuffixID`	*Type:* integer. Nullable The suffix to the name of the end-user. Foreign key to the `UserSuffix` table.
`JobTitle`	*Type:* text (max 128 characters). Nullable The job title of the end-user.
`BusinessPhoneNumber`	*Type:* text (max 30 characters). Nullable The work phone number of the end-user.
`MobilePhoneNumber`	*Type:* text (max 30 characters). Nullable The mobile phone number of the end-user.
`FaxPhoneNumber`	*Type:* text (max 30 characters). Nullable The fax number of the end-user.
`Address_Street`	*Type:* text (max 200 characters). Nullable The street address of the end-user.
`Address_City`	*Type:* text (max 200 characters). Nullable The city or suburb name of the end-user.
`Address_State`	*Type:* text (max 200 characters). Nullable The state or province of the end-user.
`Address_ZIP`	*Type:* text (max 20 characters). Nullable The ZIP or postal code of the end-user.

Database Column	Details
`Address_Country`	*Type:* text (max 100 characters). Nullable The country of the end-user.
`Email`	*Type:* text (max 200 characters). Key. Nullable The email address of the end-user.
`AlternateEmail`	*Type:* text (max 200 characters). Nullable The alternate email address of the end-user.
`Messenger`	*Type:* text (max 200 characters). Nullable The instant messenger address of the end-user.
`ManagerID`	*Type:* integer. Key. Nullable The manager of the end-user. Foreign key to another end-user in the `ComplianceUser` table.
`CurrencyID`	*Type:* integer. Nullable No longer in use - default currency is now stored in the `OperatorTenantSetting` table.
`UserStatusID`	*Type:* integer. Key The end-user's status. Foreign key to the `ComplianceUserStatus` table.
`EmploymentStatusID`	*Type:* integer. Nullable The end-user's employment status. Foreign key to the `EmploymentStatus` table.
`IsIncluded`	*Type:* boolean If `False`, the end-user's login name is in the list of excluded accounts, and this end-user will not consume licenses or be recorded as the last-logged-on or calculated end-user of a computer. This end-user will also not appear in many lists of end-users.
`CreationUser`	*Type:* text (max 128 characters). Nullable The operator who created the record.
`CreationDate`	*Type:* datetime The date the record was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The operator who last updated the record.
`UpdatedDate`	*Type:* datetime. Nullable The date the record was last updated.

Database Column	Details
ComplianceUser InventorySourceTypeID	*Type:* integer Whether this end-user has ever been reported in inventory, or has been manually created and maintained. Foreign key to the ComplianceUserInventorySourceType table.
InventoryAgent	*Type:* text (max 64 characters). Nullable If this end-user is reported in inventory, the name of the person or tool that performed the last inventory.
GeneratedFromEmail	*Type:* boolean If True, the ComplianceUser record was generated using the email address provided by a source connection.

ComplianceUserConnection Table

ComplianceUserConnection stores a link between end-users in ComplianceUser which have been reported in inventory, and external IDs which can be used to identify them in their inventory sources. End-users reported in multiple inventory sources will appear multiple times in this table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 546: Database columns for ComplianceUserConnection table

Database Column	Details
ComplianceUserID	*Type:* integer. Key A unique identifier for the end-user. Foreign key to the ComplianceUser table.
ComplianceConnectionID	*Type:* integer. Key The inventory source where the end-user was reported. Foreign key to the ComplianceConnection table.
ExternalID	*Type:* big integer. Key A (hopefully unique) identifier for the end-user in the external inventory source.

ComplianceUserInventorySourceType Table

ComplianceUserInventorySourceType is a static table used to define possible end-user inventory source values (that is, whether the end-user was created manually or reported by the compliance importer).

Table 547: Database columns for `ComplianceUserInventorySourceType` table

Database Column	Details
`ComplianceUser InventorySourceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceUserInventorySourceType`. Possible values and the corresponding default strings are: • 1 = Automatic (end-user was recently updated during an inventory import) • 2 = Manual (end-user was created manually by an operator, using FlexNet Manager Suite, and has never been updated by the compliance importer).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an inventory source. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the inventory source resource string has no translation.

ComplianceUserStatus Table

`ComplianceUserStatus` is a static table listing status values for end-user.

Table 548: Database columns for `ComplianceUserStatus` table

Database Column	Details
`ComplianceUserStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComplianceUserStatus`. Possible values and the corresponding default strings are: • 1 = Active • 2 = Inactive • 3 = Retired • 4 = On leave • 5 = Pending (perhaps for an employee just starting with the company).
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an end-user status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

Database Column	Details
`IsUserActive`	*Type:* boolean. Key If this field is set to `False`, any end-user with this status will not consume licenses or be recorded as the last-logged-on or calculated end-user of a computer. This end-user will also not appear in many lists of end-users.

EmploymentStatus Table

`EmploymentStatus` is a static table listing possible employment statuses values for end-users.

Table 549: Database columns for `EmploymentStatus` table

Database Column	Details
`EmploymentStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `EmploymentStatus`. Possible values and the corresponding default strings are: • 1 = Employee • 2 = Consultant • 3 = Temporary • 4 = Part time • 5 = Casual.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an employment status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the employment status resource string has no translation.

UserSuffix Table

`UserSuffix` is a static table listing possible name suffixes for end-users.

Table 550: Database columns for `UserSuffix` table

Database Column	Details
`UserSuffixID`	*Type:* integer. Key. Generated ID A unique identifier for each `UserSuffix`. Possible values and the corresponding default strings are: • 1 = Jr. • 2 = Sr. • 3 = I • 4 = II • 5 = III.
`ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an end-user name suffix. Foreign key to the `ComplianceResourceString` table.
`DefaultString`	*Type:* text (max 100 characters) The text to display if the suffix resource string has no translation.

UserTitle Table

`UserTitle` is a static table listing the possible titles of end-users.

Table 551: Database columns for `UserTitle` table

Database Column	Details
`UserTitleID`	*Type:* integer. Key. Generated ID A unique identifier for each `UserTitle`. Possible values and the corresponding default strings are: • 1 = Mr. • 2 = Miss • 3 = Mrs. • 4 = Ms. • 5 = Dr. • 6 = Prof.

Database Column	Details
`ResourceString`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing an end-user title. Foreign key to the `ComplianceResourceString` table.
`DefaultString`	*Type:* text (max 100 characters) The text to display if the title resource string has no translation.

Compliance.SAP Tables

The complete set of database tables documented here includes:

- SAPActivityCheckMultipleLogons table (see SAPActivityCheckMultipleLogons Table)
- SAPActivityCheckSummary table (see SAPActivityCheckSummary Table)
- SAPActivityCheckWorkTime table (see SAPActivityCheckWorkTime Table)
- SAPCompositeRole table (see SAPCompositeRole Table)
- SAPConnectivityDirectionType table (see SAPConnectivityDirectionType Table)
- SAPConnectivityType table (see SAPConnectivityType Table)
- SAPConsolidatedUser table (see SAPConsolidatedUser Table)
- SAPConsolidatedUserDuplicate table (see SAPConsolidatedUserDuplicate Table)
- SAPConsumption table (see SAPConsumption Table)
- SAPContentEngine table (see SAPContentEngine Table)
- SAPContentEngineRule table (see SAPContentEngineRule Table)
- SAPDuplicateUserRecommendation table (see SAPDuplicateUserRecommendation Table)
- SAPEngine table (see SAPEngine Table)
- SAPEngineConsumptionSummary table (see SAPEngineConsumptionSummary Table)
- SAPEngineMetric table (see SAPEngineMetric Table)
- SAPEngineMetricName table (see SAPEngineMetricName Table)
- SAPEngineName table (see SAPEngineName Table)
- SAPEnginePeriodType table (see SAPEnginePeriodType Table)
- SAPEngineSystemConsumption table (see SAPEngineSystemConsumption Table)
- SAPImportedInventoryFileDigest table (see SAPImportedInventoryFileDigest Table)
- SAPLicenseRatio table (see SAPLicenseRatio Table)

- SAPLicenseRecommendation table (see SAPLicenseRecommendation Table)
- SAPLicenseType table (see SAPLicenseType Table)
- SAPLicenseTypeHierarchy table (see SAPLicenseTypeHierarchy Table)
- SAPLicenseTypeName table (see SAPLicenseTypeName Table)
- SAPModule table (see SAPModule Table)
- SAPMultipleLogon table (see SAPMultipleLogon Table)
- SAPObject table (see SAPObject Table)
- SAPObjectType table (see SAPObjectType Table)
- SAPRFCConnection table (see SAPRFCConnection Table)
- SAPRFCConnectionSummary table (see SAPRFCConnectionSummary Table)
- SAPRecommendationAdjustmentReason table (see SAPRecommendationAdjustmentReason Table)
- SAPRecommendationProcessedStatus table (see SAPRecommendationProcessedStatus Table)
- SAPRecommendationSet table (see SAPRecommendationSet Table)
- SAPRecommendationSetStatus table (see SAPRecommendationSetStatus Table)
- SAPRecommendationSetSummary table (see SAPRecommendationSetSummary Table)
- SAPRole table (see SAPRole Table)
- SAPRoleConsumption table (see SAPRoleConsumption Table)
- SAPRoleTransactionCode table (see SAPRoleTransactionCode Table)
- SAPRule table (see SAPRule Table)
- SAPRuleAlgorithm table (see SAPRuleAlgorithm Table)
- SAPRuleCategory table (see SAPRuleCategory Table)
- SAPRuleMapping table (see SAPRuleMapping Table)
- SAPRuleSet table (see SAPRuleSet Table)
- SAPRuleSetMapping table (see SAPRuleSetMapping Table)
- SAPRuleType table (see SAPRuleType Table)
- SAPSecurityUser table (see SAPSecurityUser Table)
- SAPSystem table (see SAPSystem Table)
- SAPSystemActivityCheckSummary table (see SAPSystemActivityCheckSummary Table)
- SAPSystemClass table (see SAPSystemClass Table)
- SAPSystemEngineMetric table (see SAPSystemEngineMetric Table)

- SAPSystemEnvironment table (see SAPSystemEnvironment Table)
- SAPSystemGroup table (see SAPSystemGroup Table)
- SAPSystemLandscape table (see SAPSystemLandscape Table)
- SAPSystemLandscapeEngine table (see SAPSystemLandscapeEngine Table)
- SAPSystemLandscapeEngineMapping table (see SAPSystemLandscapeEngineMapping Table)
- SAPSystemLandscapeLicenseType table (see SAPSystemLandscapeLicenseType Table)
- SAPSystemLandscapeLicenseTypeHierarchy table (see SAPSystemLandscapeLicenseTypeHierarchy Table)
- SAPSystemLandscapeLicenseTypeImport table (see SAPSystemLandscapeLicenseTypeImport Table)
- SAPSystemLandscapeStatus table (see SAPSystemLandscapeStatus Table)
- SAPSystemMigrationStatus table (see SAPSystemMigrationStatus Table)
- SAPSystemModule table (see SAPSystemModule Table)
- SAPSystemObject table (see SAPSystemObject Table)
- SAPSystemPriceList table (see SAPSystemPriceList Table)
- SAPSystemPriceListName table (see SAPSystemPriceListName Table)
- SAPSystemRFCConnectionSummary table (see SAPSystemRFCConnectionSummary Table)
- SAPSystemRoleType table (see SAPSystemRoleType Table)
- SAPSystemType table (see SAPSystemType Table)
- SAPTransactionProfile table (see SAPTransactionProfile Table)
- SAPTransactionProfileObject table (see SAPTransactionProfileObject Table)
- SAPUser table (see SAPUser Table)
- SAPUserRole table (see SAPUserRole Table)
- SAPUserType table (see SAPUserType Table)

SAPActivityCheckMultipleLogons Table

This table stores SAP activity check data related to work time.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 552: Database columns for `SAPActivityCheckMultipleLogons` table

Database Column	Details
`SAPActivityCheck MultipleLogonsID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP activity check multiple login data.
`SAPActivityCheck MultipleLogonsUID`	*Type:* text (max 32 characters). Key The SAP unique identifier for the SAP activity check multiple login data.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to the system landscape that the SAP activity check multiple login data belongs to.
`MeasurementDate`	*Type:* datetime The date that the SAP activity check multiple login data was measured.
`MeasurementPeriodStart Date`	*Type:* datetime The start date that the SAP activity check multiple login data was measured from.
`MeasurementPeriodEndDate`	*Type:* datetime The end date that the SAP activity check multiple login data was measured to.
`NumberOfMultipleLogons`	*Type:* integer The number of logons the user account has made from different systems at the same time during the measurement period.
`MultipleLogonsPeakDate`	*Type:* datetime The date where the number of logons the user account has made from different systems at the same time during the measurement period reached its peak value.

SAPActivityCheckSummary Table

This table stores SAP activity check summary data.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 553: Database columns for `SAPActivityCheckSummary` table

Database Column	Details
`SAPActivityCheckSummaryID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP activity check summary.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to the system landscape that the SAP activity check work time data belongs to.
`HasExceededBreakDuration`	*Type:* boolean Indicates whether or not the user has exceeded the minimum required break duration.
`HasMultipleLogons`	*Type:* boolean Indicates whether or not the user has multiple logons.
`IsHidden`	*Type:* boolean Is this record marked as hidden in the UI.

SAPActivityCheckWorkTime Table

This table stores SAP activity check data related to work time.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 554: Database columns for `SAPActivityCheckWorkTime` table

Database Column	Details
`SAPActivityCheckWork TimeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP activity check work time data.
`SAPActivityCheckWork TimeUID`	*Type:* text (max 32 characters). Key The SAP unique identifier for the SAP activity check work time data.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user.

Database Column	Details
SAPSystemLandscapeID	*Type:* integer. Key Foreign key to the system landscape that the SAP activity check work time data belongs to.
MeasurementDate	*Type:* datetime The date that the SAP activity check work time data was measured.
MeasurementPeriodStart Date	*Type:* datetime The start date that the SAP activity check work time data was measured from.
MeasurementPeriodEndDate	*Type:* datetime The end date that the SAP activity check work time data was measured to.
BreakDurationSetting	*Type:* integer The minimum number of seconds that a user must not be running any transactions in a 24 hour period.
TableName	*Type:* text (max 256 characters). Key The name of the SAP table that was accessed during the minimum required break period.
BreakDurationResult	*Type:* integer The number of days that the user has not meet the minimum break duration setting during the measurement period.

SAPCompositeRole Table

This table stores SAP composite roles.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 555: Database columns for `SAPCompositeRole` table

Database Column	Details
SAPCompositeRoleID	*Type:* integer. Key. Generated ID A unique identifier for SAP composite role.
CompositeRoleID	*Type:* integer. Key Foreign key to SAP role which contain one or more single roles.
SingleRoleID	*Type:* integer. Key Foreign key to SAP role that is a member if the composite role.

SAPConnectivityDirectionType Table

This table stores SAP connectivity direction type.

Table 556: Database columns for `SAPConnectivityDirectionType` table

Database Column	Details
`SAPConnectivity DirectionTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP connectivity direction type.
`TypeName`	*Type:* text (max 64 characters). Key A unique lookup for each `SAPConnectivityDirectionType`. Possible values and the corresponding default strings are: • Out • In • InOut
`ResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP connectivity type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the SAP connectivity type resource string has no translation.

SAPConnectivityType Table

This table stores SAP connectivity type.

Table 557: Database columns for `SAPConnectivityType` table

Database Column	Details
`SAPConnectivityTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP connectivity type.
`TypeName`	*Type:* text (max 64 characters). Key A unique lookup for each `SAPConnectivityType`. Possible values and the corresponding default strings are: • Interactive • Batch

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP connectivity type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the SAP connectivity type resource string has no translation.

SAPConsolidatedUser Table

This table stores the data specific to an SAP consolidated user.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 558: Database columns for `SAPConsolidatedUser` table

Database Column	Details
`SAPConsolidatedUserID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP consolidated user.
`UserUID`	*Type:* text (max 128 characters). Key A globally unique identifier for the SAP license recommendation.
`SAPRecommendationSetID`	*Type:* integer. Key Foreign key to the SAP recommendation set that the duplicate user recommendation belongs to.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user that the duplicate user recommendation belongs to.
`UserName`	*Type:* text The user name of the user that the duplicate user recommendation belongs to.
`DuplicateGroupNum`	*Type:* integer The unique identifier showing which users are duplicates of one another.
`LicenseType`	*Type:* text (max 2 characters). Nullable The license code originally assigned to the user.
`IsConsolidatedBySAP`	*Type:* boolean Whether or not this user is consolidated by SAP.

Database Column	Details
`OptimalLicenseType`	*Type:* text (max 2 characters). Nullable The license code recommended the user be assigned ignoring license ratios and rebalancing.

SAPConsolidatedUserDuplicate Table

This table stores the data specific to an SAP consolidated user duplicate.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 559: Database columns for `SAPConsolidatedUserDuplicate` table

Database Column	Details
`SAPConsolidatedUser DuplicateID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP consolidated duplicate user.
`SAPRecommendationSetID`	*Type:* integer. Key Foreign key to the SAP recommendation set that the duplicate user belongs to.
`SAPConsolidatedUserID`	*Type:* integer. Key Foreign key to the SAP consolidated user linked to an SAP user.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user that the duplicate user belongs to.
`IsConsolidatedBySAP`	*Type:* boolean. Key Whether or not this user is consolidated by SAP.

SAPConsumption Table

This table stores the data related to the definition of SAP consumption data.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 560: Database columns for `SAPConsumption` table

Database Column	Details
`SAPConsumptionID`	*Type:* integer. Key. Generated ID The unique identifier for the SAP consumption.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user that the consumption belongs to.
`TimePeriodStartDate`	*Type:* datetime. Key The date and time of the consumption
`AccountObject`	*Type:* text (max 40 characters). Key The account object
`AccountObjectDetails`	*Type:* text (max 40 characters). Key The account object details
`EntryType`	*Type:* text (max 1 characters). Key The consumption entry type
`TaskType`	*Type:* text (max 2 characters). Key The consumption task type
`CPUTime`	*Type:* decimal. Key. Nullable CPU usage in seconds
`MemoryUsed`	*Type:* big integer. Nullable Memory used
`PrivateMemoryUsed`	*Type:* big integer. Nullable Private memory used
`AccessCount`	*Type:* integer. Nullable Number of times the object has been used/accessed.

SAPContentEngine Table

This table stores an engine from downloadable content.

Table 561: Database columns for `SAPContentEngine` table

Database Column	Details
`SAPContentEngineID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP content engine table.

Database Column	Details
EngineContentUID	*Type:* text (max 128 characters). Key A global unique identifier for the engine.
EngineName	*Type:* text (max 128 characters) Name of engine.
EngineDescription	*Type:* text. Nullable Description of engine.
Comments	*Type:* text. Nullable Comments from factory.
ApplicationID	*Type:* integer. Nullable SAP internal application ID
ConsumptionUnit	*Type:* text. Nullable Unit description to describe the consumption amount.
CreationDate	*Type:* datetime The data and time the engine was created.
UpdatedDate	*Type:* datetime The date and time the engine was last updated.

SAPContentEngineRule Table

This table stores an engine rule from downloadable content.

Table 562: Database columns for SAPContentEngineRule table

Database Column	Details
SAPContentEngineRuleID	*Type:* integer. Key. Generated ID A unique identifier for the SAP content engine rule table.
EngineContentUID	*Type:* text (max 128 characters) A global unique identifier for the engine.
RuleContentUID	*Type:* text (max 128 characters). Key A global unique identifier for the engine rule.
RuleName	*Type:* text (max 128 characters) Name of engine rule.
RuleDefinition	*Type:* text. Nullable Rule definition for calculating consumption of an engine.

Database Column	Details
IsDefault	*Type:* boolean Is this formula the default for created packages.
CreationDate	*Type:* datetime The data and time the engine rule was created.
UpdatedDate	*Type:* datetime The date and time the engine rule was last updated.

SAPDuplicateUserRecommendation Table

This table stores the data specific to an SAP duplicate user recommendation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 563: Database columns for `SAPDuplicateUserRecommendation` table

Database Column	Details
SAPDuplicateUser RecommendationID	*Type:* integer. Key. Generated ID A unique identifier for the SAP duplicate user recommendation.
RecommendationUID	*Type:* text (max 128 characters). Key A globally unique identifier for the SAP license recommendation.
SAPRecommendationSetID	*Type:* integer. Key Foreign key to the SAP recommendation set that the duplicate user recommendation belongs to.
DuplicateGroupNum	*Type:* integer The unique identifier showing which users are duplicates of one another.
SAPUserID	*Type:* integer. Key. Nullable The unique identifier of the user that the duplicate user recommendation belongs to.
UserName	*Type:* text The user name of the user that the duplicate user recommendation belongs to.
SystemID	*Type:* text The ID of the system that the duplicate user recommendation belongs to.

Database Column	Details
ClientID	*Type:* text The ID of the client that the duplicate user recommendation belongs to.
IsConsolidatedBySAP	*Type:* boolean Whether or not this duplicate is consolidated by SAP.
SAPRuleID	*Type:* integer. Nullable The unique identifier of the rule used to produce the duplicate user recommendation.
RuleSetName	*Type:* text. Nullable The name of the rule set used to produce the duplicate user recommendation.
RuleName	*Type:* text. Nullable The name of the rule used to produce the duplicate user recommendation.
RuleSequenceNumber	*Type:* integer. Nullable The sequence number of the rule used to produce the duplicate user recommendation.
RuleMessage	*Type:* text. Nullable The message produced given by the rule used to produce the duplicate user recommendation.
SAPRecommendation ProcessedStatusID	*Type:* integer Foreign key to the SAP recommendation processed status of the duplicate user recommendation.
RuleMessageResourceName	*Type:* text (max 256 characters). Nullable The resource name of the message produced given by the rule used to produce the duplicate user recommendation.
RuleMessageParameters	*Type:* text (max 256 characters). Nullable The parameters used by the message produced given by the rule used to produce the duplicate user recommendation.

SAPEngine Table

This table stores the application engines used in SAP.

Table 564: Database columns for SAPEngine table

Database Column	Details
SAPEngineID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine table.

Database Column	Details
ApplicationID	*Type:* integer. Key The unique identifier given to the application engine by SAP.
SAPEnginePeriodTypeID	*Type:* integer A unique identifier for the SAP engine period type.

SAPEngineConsumptionSummary Table

This table stores the total consumption of SAP package consumption recommendation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 565: Database columns for `SAPEngineConsumptionSummary` table

Database Column	Details
SAPEngineConsumption SummaryID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine consumption.
LandscapeUID	*Type:* text (max 128 characters) A global unique identifier for the system landscape the summary belongs to.
RecommendationSetUID	*Type:* text (max 128 characters) A global unique identifier for the SAP recommendation set the summary belongs to.
SAPRecommendationSet StatusID	*Type:* integer The status of the recommendation set.
SAPSystemLandscape EngineID	*Type:* integer. Key. Nullable A unique identifier for the SAP system landscape engine table.
EngineUID	*Type:* text (max 128 characters) A global unique identifier for the SAP engine in a system landscape.
EngineName	*Type:* text (max 128 characters) Name of engine.
Consumed	*Type:* decimal. Nullable The number of consumed units for the package (null = indeterminate)
ConsumptionUnit	*Type:* text. Nullable Unit description to describe the consumption amount.

Database Column	Details
ReasonMessage	*Type:* text. Nullable And optional message detailing the reason for the consumed result.
EntitlementsPurchased	*Type:* integer Total number of purchased license entitlements.
EngineUnitPrice	*Type:* currency. Nullable The unit price of a license entitlement.
EngineUnitPriceRateID	*Type:* integer. Nullable The unit price rate of a license entitlement.
CalculationDate	*Type:* datetime The date of the license postion calculation.
SystemMeasurementDate	*Type:* datetime The date the system measurement calculation was performed.

SAPEngineMetric Table

This table stores the application engine metrics used in SAP.

Table 566: Database columns for SAPEngineMetric table

Database Column	Details
SAPEngineMetricID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine metric table.
MetricID	*Type:* integer. Key The unique identifier given to the application engine metric by SAP.

SAPEngineMetricName Table

This table stores the name of applications engine metrics in different languages.

Table 567: Database columns for SAPEngineMetricName table

Database Column	Details
SAPEngineMetricNameID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine metric name table.

Database Column	Details
SAPEngineMetricID	*Type:* integer. Key The unique identifier of an SAP engine metric.
EngineMetricName	*Type:* text (max 128 characters). Key. Nullable The name of the application engine metric.
Language	*Type:* text (max 4 characters). Key A unique code to identify the language.

SAPEngineName Table

This table stores the name of applications engines in different languages.

Table 568: Database columns for SAPEngineName table

Database Column	Details
SAPEngineNameID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine name table.
SAPEngineID	*Type:* integer. Key The unique identifier of an SAP engine.
EngineName	*Type:* text (max 128 characters). Key. Nullable The name of the application engine.
Language	*Type:* text (max 4 characters). Key A unique code to identify the language.

SAPEnginePeriodType Table

This table stores the types of SAP applications engine measurement periods.

Table 569: Database columns for SAPEnginePeriodType table

Database Column	Details
SAPEnginePeriodTypeID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine period type table.

Database Column	Details
PeriodTypeCode	*Type:* text (max 1 characters). Key A unique lookup for each SAPEnginePeriodType. Possible values and the corresponding default strings are: • Y = Last year • C = Last calendar year • T = Year to date • M = This month • Q = This quarter • 6 = Last six months • U = Undefined
ResourceName	*Type:* text (max 256 characters) A localizable resource string representing an SAP system type. Foreign key to the ComplianceResourceString table.
DefaultValue	*Type:* text (max 100 characters) The text to display if the system type resource string has no translation.

SAPEngineSystemConsumption Table

This table stores the per-system consumption of SAP package consumption recommendation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 570: Database columns for SAPEngineSystemConsumption table

Database Column	Details
SAPEngineSystem ConsumptionID	*Type:* integer. Key. Generated ID A unique identifier for the SAP engine consumption.
SAPRecommendationSetID	*Type:* integer. Key Foreign key to the SAP recommendation set that the license recommendation belongs to.
SAPSystemLandscape EngineID	*Type:* integer. Key. Nullable A unique identifier for the SAP system landscape engine table.

Database Column	Details
EngineUID	*Type:* text (max 128 characters) A global unique identifier for the SAP engine in a system landscape.
EngineName	*Type:* text (max 128 characters) Name of engine.
SAPSystemID	*Type:* integer. Key. Nullable The unique identifier of the system that the consumed count belongs to.
SystemID	*Type:* text The ID of the system that the license recommendation belongs to.
ClientID	*Type:* text The ID of the client that the license recommendation belongs to.
Consumed	*Type:* decimal. Nullable The number of consumed units for the package (null = indeterminate)
ReasonMessage	*Type:* text. Nullable And optional message detailing the reason for the consumed result.
SystemMeasurementDate	*Type:* datetime. Nullable The date the system measurement calculation was performed.

SAPImportedInventoryFileDigest Table

This table stores digests of imported SAP inventory files.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 571: Database columns for `SAPImportedInventoryFileDigest` table

Database Column	Details
SAPImportedInventory FileDigestID	*Type:* integer. Key. Generated ID A unique identifier for the SAP impoted inventory file digest.
LandscapeUID	*Type:* text (max 128 characters). Key A global unique identifier for the system landscape.
SystemID	*Type:* text (max 64 characters). Key The System ID that is used to identify the SAP system.

Database Column	Details
`ClientID`	*Type:* text (max 32 characters). Key The Client ID that is to be used when connecting to the SAP system.
`SystemNumber`	*Type:* text (max 32 characters). Key. Nullable The SAP system number. This value will be used by the RFC connection.
`MD5Hash`	*Type:* text (max 64 characters). Key MD5 hash of imported SAP inventory file content.
`CreationDate`	*Type:* datetime The data and time the digest record was created.

SAPLicenseRatio Table

This table stores SAP license ratios used for recommending optimizations for SAP.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 572: Database columns for `SAPLicenseRatio` table

Database Column	Details
`SAPLicenseRatioID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP license ratio.
`LeftLicenseType`	*Type:* text (max 2 characters) The type of license assigned to the left side of the license ratio.
`LeftValue`	*Type:* integer The value belonging to the left side of the license ratio.
`RightLicenseType`	*Type:* text (max 2 characters) The type of license assigned to the right side of the license ratio.
`RightValue`	*Type:* integer The value belonging to the right side of the license ratio.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to the system landscape that the license ratio belongs to.
`IsActive`	*Type:* boolean Whether or not this license ratio is used to automatically optimize SAP license assignments.

Database Column	Details
`CreationUser`	*Type:* text (max 256 characters) The user who created the license ratio.
`CreationDate`	*Type:* datetime The data and time the license ratio was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the license ratio.
`UpdatedDate`	*Type:* datetime The date and time the license ratio was last updated.

SAPLicenseRecommendation Table

This table stores the data specific to an SAP license recommendation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 573: Database columns for `SAPLicenseRecommendation` table

Database Column	Details
`SAPLicense RecommendationID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP license recommendation.
`RecommendationUID`	*Type:* text (max 128 characters). Key A globally unique identifier for the SAP license recommendation.
`SAPRecommendationSetID`	*Type:* integer. Key Foreign key to the SAP recommendation set that the license recommendation belongs to.
`SAPUserID`	*Type:* integer. Key. Nullable The unique identifier of the user that the license recommendation belongs to.
`UserName`	*Type:* text The user name of the user that the license recommendation belongs to.
`SystemID`	*Type:* text The ID of the system that the license recommendation belongs to.
`ClientID`	*Type:* text The ID of the client that the license recommendation belongs to.

Database Column	Details
`OriginalLicenseType`	*Type:* text (max 2 characters). Nullable The license code originally assigned to the user.
`RecommendedLicenseType`	*Type:* text (max 2 characters). Nullable The license code recommended the user be assigned.
`SAPRuleID`	*Type:* integer. Nullable The unique identifier of the rule used to produce the license recommendation.
`RuleSetName`	*Type:* text. Nullable The name of the rule set used to produce the license recommendation.
`RuleName`	*Type:* text. Nullable The name of the rule used to produce the license recommendation.
`RuleSequenceNumber`	*Type:* integer. Nullable The sequence number of the rule used to produce the license recommendation.
`RuleMessage`	*Type:* text. Nullable The message produced given by the rule used to produce the license recommendation.
`SAPRecommendation ProcessedStatusID`	*Type:* integer Foreign key to the SAP recommendation processed status of the license recommendation.
`OptimalLicenseType`	*Type:* text (max 2 characters). Nullable The license code recommended the user be assigned ignoring license ratios and rebalancing.
`SAPRecommendation AdjustmentReasonID`	*Type:* integer. Nullable The unique identifier of the reason why the license recommendation differs from optimal.
`RuleMessageResourceName`	*Type:* text (max 256 characters). Nullable The resource name of the message produced given by the rule used to produce the license recommendation.
`RuleMessageParameters`	*Type:* text (max 256 characters). Nullable The parameters used by the message produced given by the rule used to produce the license recommendation.

SAPLicenseType Table

This table stores the SAP license type on SAP systems.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 574: Database columns for `SAPLicenseType` table

Database Column	Details
`SAPLicenseTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP license type.
`SAPSystemID`	*Type:* integer. Key Foreign key to the system that the SAP license type belongs to.
`Identifier`	*Type:* text (max 2 characters). Key SAP license type identifier
`SAPSpecialVersionID`	*Type:* integer. Key. Nullable SAP special version ID
`Active`	*Type:* boolean Indicates whether the SAP license type is active or not active.
`SpecialVersionAssignment`	*Type:* boolean. Nullable Indicates whether the SAP license type is affected by special version.
`SSCR_Allow`	*Type:* boolean. Nullable
`IsDeleted`	*Type:* boolean Indicated whether the SAP license type has been deleted or not.

SAPLicenseTypeHierarchy Table

This table stores SAP license type hierarchy.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 575: Database columns for `SAPLicenseTypeHierarchy` table

Database Column	Details
`SAPLicenseTypeHierarchyID`	*Type:* integer. Key. Generated ID The unique identifer for the SAP license type hierarchy.

Database Column	Details
SAPLicenseTypeID	*Type:* integer. Key Parent license type. Foreign key to the SAP license type.
ChildSAPLicenseTypeID	*Type:* integer. Key Child license type. Foreign key to SAP license type.

SAPLicenseTypeName Table

This table stores SAP license types in various languages.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 576: Database columns for `SAPLicenseTypeName` table

Database Column	Details
SAPLicenseTypeNameID	*Type:* integer. Key. Generated ID Unique identifier for SAP license type name.
SAPLicenseTypeID	*Type:* integer. Key Foreign key to the SAP license type.
Language	*Type:* text (max 4 characters) The two letter language code.
ShortName	*Type:* text (max 128 characters). Nullable SAP license type short name.
LongName	*Type:* text (max 256 characters). Nullable SAP license type long name

SAPModule Table

This table stores the modules used in SAP.

Table 577: Database columns for `SAPModule` table

Database Column	Details
SAPModuleID	*Type:* integer. Key. Generated ID A unique identifier for the SAP module table.

Database Column	Details
`ModuleName`	*Type:* text (max 64 characters). Key The name of the module.
`SubModuleName`	*Type:* text (max 64 characters). Key. Nullable The name of the sub module.
`ObjectName`	*Type:* text (max 40 characters). Key. Nullable The name of the object linked to the SAP system module.

SAPMultipleLogon Table

This table stores logon metrics for SAP users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 578: Database columns for `SAPMultipleLogon` table

Database Column	Details
`SAPMultipleLogonID`	*Type:* integer. Key. Generated ID A unique identifier for the user's logon metrics
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user.
`Year`	*Type:* text (max 4 characters) The year of the logon metrics
`NumberOfMultipleLogon`	*Type:* integer. Nullable Number of multiple concurrent logon
`MaxMultipleLogon`	*Type:* integer. Nullable Maximum number of concurrent logon

SAPObject Table

This table stores the SAP object

Table 579: Database columns for `SAPObject` table

Database Column	Details
`SAPObjectID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP object.
`ObjectName`	*Type:* text (max 40 characters). Key Name of the object
`SAPObjectTypeID`	*Type:* integer. Key Foreign key to the SAP object type that identifies the object type.

SAPObjectType Table

This tables stores SAP object types

Table 580: Database columns for `SAPObjectType` table

Database Column	Details
`SAPObjectTypeID`	*Type:* integer. Key. Generated ID • 1 = Transaction • 2 = Report • 3 = Job • 4 = NonSAP
`TypeName`	*Type:* text (max 64 characters). Key A unique name for the SAP object type.
`ResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP object type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the object type resource string has no translation.

SAPRFCConnection Table

This table stores RFC connections made to the SAP system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 581: Database columns for `SAPRFCConnection` table

Database Column	Details
`SAPRFCConnectionID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP RFC consumption.
`SAPUserID`	*Type:* integer. Key SAP user performing the RFC connection. Foreign key to the `SAPUser` table
`TimePeriodStartDate`	*Type:* datetime. Key The date and time of the RFC connection
`RemoteSystem`	*Type:* text (max 128 characters). Key Remote system name connecting to the SAP system.
`ProgramName`	*Type:* text (max 40 characters). Key Program Name associated to the function name.
`FunctionName`	*Type:* text (max 40 characters). Key The function executed by the RFC calls
`TaskType`	*Type:* text (max 2 characters). Key. Nullable Task type.
`RFCDestination`	*Type:* text (max 128 characters). Key The RFC destination string value.
`TotalExecutionCount`	*Type:* integer The number of times the function is executed.
`TotalExecutionTime`	*Type:* decimal Total execution time.
`TotalCallTime`	*Type:* decimal Total call time.
`TotalDataSent`	*Type:* big integer Total data sent by the RFC calls.
`TotalDataReceived`	*Type:* big integer Total data received b the RFC calls.

SAPRFCConnectionSummary Table

This table stores the remote RFC consumption summary. It only includes Non-SAP remote system

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 582: Database columns for `SAPRFCConnectionSummary` table

Database Column	Details
`SAPRFCConnectionSummaryID`	*Type:* integer. Key. Generated ID A unique identifier for the remote SAP RFC connection summary.
`RemoteSystem`	*Type:* text (max 128 characters). Key Remote system making the RFC calls.
`NumberOfSAPSystems`	*Type:* integer Number of SAP systems the Remote System is connecting to.
`NumberOfDialogUsers`	*Type:* integer Number of Dialog SAP users making the RFC call to the SAP system.
`NumberOfNonDialogUsers`	*Type:* integer Number of Service SAP users making the RFC call to the SAP system.
`NumberOfExecutedPrograms`	*Type:* integer The number of executed programs
`NumberOfExecutedFunctions`	*Type:* integer The number of executed functions
`TotalExecutionCount`	*Type:* integer The total excution count of all functions.
`TotalExecutionTime`	*Type:* decimal Total execution time.
`TotalCallTime`	*Type:* decimal Total call time.
`TotalDataSent`	*Type:* big integer Total data sent by the RFC calls.
`TotalDataReceived`	*Type:* big integer Total data received b the RFC calls.
`IsHidden`	*Type:* boolean Is this record marked as hidden in the UI.

SAPRecommendationAdjustmentReason Table

This table stores SAP Recommendation adjustment reasons.

Table 583: Database columns for `SAPRecommendationAdjustmentReason` table

Database Column	Details
`SAPRecommendationAdjustmentReasonID`	*Type:* integer. Key. Generated ID A unique identifier for each `SAPRecommendationAdjustmentReason`. Possible values and the corresponding default strings are: • 1 = License ratio enforced • 2 = Excess purchase(s) of covering license type applied.
`ResourceName`	*Type:* text (max 256 characters). Key A localizable resource string representing an SAP recommendation adjustment reason. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the adjustment reason resource string has no translation.

SAPRecommendationProcessedStatus Table

This table stores SAP Recommendation Processed status.

Table 584: Database columns for `SAPRecommendationProcessedStatus` table

Database Column	Details
`SAPRecommendationProcessedStatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `SAPRecommendationProcessedStatus`. Possible values and the corresponding default strings are: • 1 = Pending • 2 = Accepted • 3 = Rejected
`ResourceName`	*Type:* text (max 256 characters). Key A localizable resource string representing an SAP recommendation processed status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

SAPRecommendationSet Table

This table stores data specific to the definition of a recommendation set.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 585: Database columns for `SAPRecommendationSet` table

Database Column	Details
`SAPRecommendationSetID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP recommendation set.
`RecommendationSetUID`	*Type:* text (max 128 characters). Key A global unique identifier for the SAP recommendation set.
`RecommendationSetName`	*Type:* text (max 128 characters) Name of recommendation set.
`RecommendationSet Description`	*Type:* text. Nullable Description of recommendation set.
`LandscapeUID`	*Type:* text (max 128 characters) A global unique identifier for the system landscape the recommendation set belongs to.
`SAPRecommendationSet StatusID`	*Type:* integer. Key The status of the recommendation set.
`CalculationDate`	*Type:* datetime. Nullable The date of the license postion calculation.
`CreationUser`	*Type:* text (max 256 characters) The user who created the recommendation set.
`CreationDate`	*Type:* datetime The data and time the recommendation set was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the recommendation set.
`UpdatedDate`	*Type:* datetime The date and time the recommendation set was last updated.
`ReviewedUser`	*Type:* text (max 256 characters). Nullable The user who reviewed the recommendation set.

Database Column	Details
`ReviewedDate`	*Type:* datetime. Nullable The date and time the recommendation set was reviewed.
`ReleasedUser`	*Type:* text (max 256 characters). Nullable The user who released the recommendation set.
`ReleasedDate`	*Type:* datetime. Nullable The date and time the recommendation set was released.
`Uploaded`	*Type:* boolean Indicates whether the recommendation set was oploaded by FNM-SAP
`UploadedDate`	*Type:* datetime. Nullable The date the recommendation set was oploaded by FNM-SAP

SAPRecommendationSetStatus Table

This table stores SAP Recommendation Set status.

Table 586: Database columns for `SAPRecommendationSetStatus` table

Database Column	Details
`SAPRecommendationSet StatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `SAPRecommendationSetStatus`. Possible values and the corresponding default strings are: • 1 = In Review • 2 = Archived • 3 = Released • 4 = New License Position • 5 = Rejected • 6 = Simulation • 7 = Creating • 8 = Previous License Position.
`ResourceName`	*Type:* text (max 256 characters). Key A localizable resource string representing an SAP recommendation set status. Foreign key to the `ComplianceResourceString` table.

Database Column	Details
DefaultValue	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

SAPRecommendationSetSummary Table

This table stores a history of SAP license positions.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 587: Database columns for SAPRecommendationSetSummary table

Database Column	Details
SAPRecommendationSet SummaryID	*Type:* integer. Key. Generated ID A unique identifier for the SAP license recommendation summary.
LandscapeUID	*Type:* text (max 128 characters). Key A global unique identifier for the system landscape the summary belongs to.
RecommendationSetUID	*Type:* text (max 128 characters). Key A global unique identifier for the SAP recommendation set the summary belongs to.
SAPRecommendationSet StatusID	*Type:* integer The status of the recommendation set.
LicenseType	*Type:* text (max 2 characters). Key. Nullable The license code to which the position applies.
EntitlementsPurchased	*Type:* integer Total number of purchased license entitlements.
EntitlementsOriginal	*Type:* integer Total number of consumed license entitlements.
EntitlementsRecommended	*Type:* integer Total number of recommended license entitlements.
LicenseTypeUnitPrice	*Type:* currency. Nullable The unit price of a license entitlement.
LicenseTypeUnitPrice RateID	*Type:* integer. Nullable The unit price rate of a license entitlement.

Database Column	Details
`CalculationDate`	*Type:* datetime The date of the license postion calculation.
`EntitlementsOptimal`	*Type:* integer Total number of recommended license entitlements without license ratio constraints.

SAPRole Table

This table stores SAP roles

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 588: Database columns for `SAPRole` table

Database Column	Details
`SAPRoleID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP role.
`SAPSystemID`	*Type:* integer. Key Foreign key to the system that the role belongs to.
`RoleName`	*Type:* text (max 30 characters) The name of the role.
`NumberOfTransactionCodes`	*Type:* integer. Nullable Total number of transaction codes allowed to be executed by this role.
`LicenseType`	*Type:* text (max 2 characters). Nullable License type associated to this role

SAPRoleConsumption Table

This table stores SAP roles and its link to SAP consumption.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 589: Database columns for `SAPRoleConsumption` table

Database Column	Details
`SAPRoleConsumptionID`	*Type:* big integer. Key. Generated ID A unique identifier for for SAP role consumption.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user that the role consumption belongs to.
`CompositeRoleID`	*Type:* integer. Key. Nullable Foreign key to SAP role.
`SingleRoleID`	*Type:* integer. Key Foreign key to SAP role.
`SingleRoleTransaction CodeID`	*Type:* integer. Key Foreign key to SAP transaction code.
`SAPConsumptionID`	*Type:* integer. Key Foreign key to SAP consumption.

SAPRoleTransactionCode Table

This table stores list of roles and its transaction codes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 590: Database columns for `SAPRoleTransactionCode` table

Database Column	Details
`SAPRoleTransactionCodeID`	*Type:* integer. Key. Generated ID A unique identifier for the roles and its transaction codes.
`SAPRoleID`	*Type:* integer. Key Foreign to the SAP Roles where transaction codes belong to.
`TCodeLow`	*Type:* text (max 80 characters). Key. Nullable Lower range of the transaction code.
`TCodeHigh`	*Type:* text (max 40 characters). Nullable Upper range of the transaction code.

SAPRule Table

This table stores SAP rules used for recommending optimizations for SAP.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 591: Database columns for `SAPRule` table

Database Column	Details
`SAPRuleID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule.
`RuleName`	*Type:* text (max 128 characters) Name of the rule.
`SAPRuleTypeID`	*Type:* integer. Key Foreign key to the rule type of the SAP rule.
`SAPRuleSetID`	*Type:* integer. Key Foreign key to the rule set that the SAP rule belongs to.
`RuleDefinition`	*Type:* text The rule definition XML used to build the rule statement used by the SAP rules engine.
`SequenceNumber`	*Type:* integer The sequence number used to designate the order of the rules within the rule set.
`SAPRuleCategoryID`	*Type:* integer. Key Foreign key to the rule category of the SAP rule.
`IsActive`	*Type:* boolean Whether or not this rule is active for execution.
`UseRuleSetMapping`	*Type:* boolean Whether or not to use mapping from the SAP rule set
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.

Database Column	Details
UpdatedDate	*Type:* datetime The date and time the system landscape was last updated.

SAPRuleAlgorithm Table

This table stores the availble SAP rule algorithms used by SAP rules.

Table 592: Database columns for SAPRuleAlgorithm table

Database Column	Details
SAPRuleAlgorithmID	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule algorithm.
AlgorithmName	*Type:* text (max 100 characters). Key A unique name for the SAP category.
SAPRuleCategoryID	*Type:* integer. Key Foreign key to the rule category of the SAP rule algorithm.
TitleResourceName	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP rule algorithm. Foreign key to the ComplianceResourceString table.
TitleDefaultValue	*Type:* text (max 100 characters) The text to display if the rule type resource string has no translation.
AlgorithmType	*Type:* text Type associated with this algorithm
AlgorithmData	*Type:* text. Nullable Data associated with this algorithm, such as a custom SQL query to run.

SAPRuleCategory Table

This table stores the different rule categories used in recommending optimizations for SAP.

Table 593: Database columns for SAPRuleCategory table

Database Column	Details
SAPRuleCategoryID	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule category.

Database Column	Details
`CategoryName`	*Type:* text (max 100 characters). Key A unique name for the SAP category.

SAPRuleMapping Table

This table stores mapping between SAP rule to either System Landscapes, System Groups or SAP systems.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 594: Database columns for `SAPRuleMapping` table

Database Column	Details
`SAPRuleMappingID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule set mapping.
`SAPRuleID`	*Type:* integer. Key Foreign key to SAP Rule ID
`SAPSystemLandscapeID`	*Type:* integer. Key. Nullable Foreign key to System Landscape ID
`SAPSystemGroupID`	*Type:* integer. Key. Nullable Foreign key to System Group ID.
`SAPSystemID`	*Type:* integer. Key. Nullable Foreign key to the SAP system.
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.
`UpdatedDate`	*Type:* datetime The date and time the system landscape was last updated.

SAPRuleSet Table

This table stores SAP rule sets used for recommending optimizations for SAP.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 595: Database columns for `SAPRuleSet` table

Database Column	Details
`SAPRuleSetID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule set.
`RuleSetName`	*Type:* text (max 128 characters) Name of rule set.
`RuleSetDescription`	*Type:* text. Nullable Description of rule set.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to the system landscape that the SAP rule set belongs to.
`IsActive`	*Type:* boolean Whether or not this rule set is used to automatically optimize SAP license assignments.
`SequenceNumber`	*Type:* integer The sequence number used to designate the order of the rule sets within the landscape.
`SAPRuleCategoryID`	*Type:* integer. Key Foreign key to the rule category of the SAP rule set.
`NumberOfConsumptionMonth`	*Type:* integer
`ConsumptionMonthEndDate`	*Type:* datetime. Nullable End date of consumption period used for recommending optiomizations. If null,
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.

Database Column	Details
UpdatedDate	*Type:* datetime The date and time the system landscape was last updated.
SecurityTypeID	*Type:* integer Security type for this object. Foreign key to the SecurityType table.

SAPRuleSetMapping Table

This table stores mapping between SAP rule sets to either System Landscapes, System Groups or SAP systems.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 596: Database columns for SAPRuleSetMapping table

Database Column	Details
SAPRuleSetMappingID	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule set mapping.
SAPRuleSetID	*Type:* integer. Key Foreign key to SAP Rule Set ID
SAPSystemLandscapeID	*Type:* integer. Key. Nullable Foreign key to System Landscape ID
SAPSystemGroupID	*Type:* integer. Key. Nullable Foreign key to System Group ID.
SAPSystemID	*Type:* integer. Key. Nullable Foreign key to the SAP system.
CreationUser	*Type:* text (max 256 characters) The user who created the system landscape.
CreationDate	*Type:* datetime The data and time the system landscape was created.
UpdatedUser	*Type:* text (max 256 characters) The last user who update the system landscape.
UpdatedDate	*Type:* datetime The date and time the system landscape was last updated.

SAPRuleType Table

This table stores the available SAP rule types used for recommending optimizations for SAP.

Table 597: Database columns for `SAPRuleType` table

Database Column	Details
`SAPRuleTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP rule type.
`TypeName`	*Type:* text (max 100 characters). Key A unique name for the SAP rule type.
`SAPRuleCategoryID`	*Type:* integer. Key Foreign key to the rule category of the SAP rule.
`TitleResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP rule type. Foreign key to the `ComplianceResourceString` table.
`TitleDefaultValue`	*Type:* text (max 100 characters) The text to display if the rule type resource string has no translation.
`RuleTemplate`	*Type:* text The template used to build a rule for the SAP rules engine.
`DefaultRuleDefinition`	*Type:* text. Nullable Default rule definition for newly created SAP rule

SAPSecurityUser Table

This table stores the operators allowed to access SAP objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 598: Database columns for `SAPSecurityUser` table

Database Column	Details
`SAPSecurityUserID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP security user table.
`TargetTypeID`	*Type:* integer. Key Target type of object with restricted access.

Database Column	Details
SAPSystemLandscapeID	*Type:* integer. Key. Nullable The unique identifier of an SAP system landscape.
SAPSystemID	*Type:* integer. Key. Nullable The unique identifier of an SAP system.
SAPRuleSetID	*Type:* integer. Key. Nullable The unique identifier of an SAP rule set.
ResourceID	*Type:* integer The unique identifier of a security resource.
ActionClassID	*Type:* integer The unique identifier of a security action class.
ComplianceOperatorID	*Type:* integer. Key The unique identifier of an operator.

SAPSystem Table

This table stores the data specific to the definition of SAP systems.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 599: Database columns for `SAPSystem` table

Database Column	Details
SAPSystemID	*Type:* integer. Key. Generated ID A unique identifier for the SAP system.
SystemName	*Type:* text (max 128 characters) The name of the SAP system.
SystemDescription	*Type:* text. Nullable A more detailed description of the SAP system.
SAPSystemLandscapeID	*Type:* integer. Key Foreign key to the system landscape that the SAP system belongs to.
SAPSystemGroupID	*Type:* integer. Key. Nullable Foreign key to the system group that the SAP system belongs to.

Database Column	Details
SAPSystemEnvironmentID	*Type:* integer. Key. Nullable The type of environment for the SAP system.
SystemID	*Type:* text (max 64 characters) The System ID that is used to identify the SAP system.
ClientID	*Type:* text (max 32 characters) The Client ID that is to be used when connecting to the SAP system.
ServerName	*Type:* text (max 256 characters). Nullable The DNS name of the SAP system. This value will be used by the RFC connection. This field can also store the SAP System's IP address.
SystemNumber	*Type:* text (max 32 characters). Nullable The SAP system number. This value will be used by the RFC connection.
Username	*Type:* text (max 256 characters). Nullable The user name that is to be used when connecting to the SAP system.
Password	*Type:* binary. Nullable The password that is to be used when connecting to the SAP system.
IsOfflineSystem	*Type:* boolean Indicates whether an SAP system is offline.
IsPortalSystem	*Type:* boolean Indicates whether the system is a portal system.
SystemStatus	*Type:* text (max 128 characters). Nullable The status of the SAP system.
UsersControlledByCUA	*Type:* boolean Identifies whether the uses on the SAP system are controlled by a CUA.
ModelView	*Type:* text (max 128 characters). Nullable Further clarification required.
CUACentralSystem	*Type:* boolean The status of the SAP system.
CUACentralSystemID	*Type:* text (max 128 characters). Nullable The System ID of the CUA system that this SAP system is controlled by.
FNMSAPRelease	*Type:* text (max 128 characters). Nullable The version of FNM for SAP installed on the SAP system.

Database Column	Details
LAWVersion	*Type:* text (max 128 characters). Nullable The version of the License Assignment Workbench module installed on the SAP system.
SAPRelease	*Type:* text (max 128 characters). Nullable The version of SAP installed on the SAP system.
SAPPatchRelease	*Type:* text (max 128 characters). Nullable The SAP patch version
STPIRelease	*Type:* text (max 128 characters). Nullable The ST-PI version
DBSystem	*Type:* text (max 128 characters). Nullable The database system running on the SAP system.
HardwareKey	*Type:* text (max 128 characters). Nullable The hardware key of the SAP system.
InstallationNumber	*Type:* text (max 128 characters). Nullable The SAP system installation number
LastChangedOn	*Type:* datetime. Nullable The date and time the SAP system data was last refreshed.
SupportPackage	*Type:* text (max 128 characters). Nullable The support package of the SAP system.
HRSystem	*Type:* text (max 128 characters). Nullable The SAP system which contains the HR data.
SystemType	*Type:* text (max 128 characters). Nullable Indicates whether the SAP system is an ABAP or JAVA based system.
DefaultLicenseType	*Type:* text (max 2 characters). Nullable Default license type for the SAP system.
ContactFirstName	*Type:* text (max 128 characters). Nullable First name of the contact for this system.
ContactLastName	*Type:* text (max 128 characters). Nullable Last name of the contact for this system.
ContactBusinessPhone Number	*Type:* text (max 30 characters). Nullable Business phone number of the contact for this system.

Database Column	Details
`ContactMobilePhoneNumber`	*Type:* text (max 30 characters). Nullable Mobile phone number of the contact for this system.
`ContactEmail`	*Type:* text (max 200 characters). Nullable Email address of the contact for this system.
`Location`	*Type:* text (max 128 characters). Nullable Location of this system.
`InventoryDate`	*Type:* datetime. Nullable The date and time the SAP system data was collected by SAP Reader.
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.
`UpdatedDate`	*Type:* datetime The date and time the system landscape was last updated.
`SecurityTypeID`	*Type:* integer. Key Security type for this object. Foreign key to the `SecurityType` table.
`AccessToModuleData`	*Type:* boolean Indicates whether the system has access to module data.
`SAPSystemTypeID`	*Type:* integer The type of system for the system. Foreign key to the `SAPSystemType` table.
`SAPConnectivityTypeID`	*Type:* integer. Nullable The type of connectivity for the SAP system. Foreign key to the `SAPConnectivityType` table.
`SAPConnectivity DirectionTypeID`	*Type:* integer. Nullable The type of SAP connectivity direction for the SAP system. Foreign key to the `SAPConnectivityDirectionType` table.
`BeaconUID`	*Type:* unique identifier. Key. Nullable The inventory beacon where this connection is defined.
`SAPSystemRoleTypeID`	*Type:* integer. Nullable The type of SAP SystemRole for the SAP system. Foreign key to `SAPSystemRoleType` Table

Database Column	Details
MasterSAPSystemID	*Type:* integer. Nullable A unique identifier for the Master SAP system.
SAPSystemClassID	*Type:* integer. Key. Nullable The class of SAP system. Foreign key to the SAPPlatformType table.
SAPSystemMigration StatusID	*Type:* integer. Nullable Migration status of the SAP system. Foreign key to the SAPSystemMigrationStatus table.

SAPSystemActivityCheckSummary Table

This table stores the link between SAP System and SAP Activity Check Summary data.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 600: Database columns for SAPSystemActivityCheckSummary table

Database Column	Details
SAPSystemActivityCheck SummaryID	*Type:* integer. Key. Generated ID A unique identifier.
SAPSystemID	*Type:* integer. Key The Non-SAP system foreign key.
SAPActivityCheckSummaryID	*Type:* integer. Key The SAP Activity Check Summary data foreign key.

SAPSystemClass Table

This table stores classes of SAP systems e.g. S/4HANA, Classic Business Suite

Table 601: Database columns for SAPSystemClass table

Database Column	Details
SAPSystemClassID	*Type:* integer. Key. Generated ID A unique identifier for the SAP system class.

Database Column	Details
`ClassName`	*Type:* text (max 64 characters). Key A unique lookup for each `SAPSystemClass`. Possible values and the corresponding default strings are: • ClassicBusinessSuite • S4Hana
`ResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing a SAP system class. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the SAP system class resource string has no translation.

SAPSystemEngineMetric Table

This table stores the value of applications engine metrics per system.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 602: Database columns for `SAPSystemEngineMetric` table

Database Column	Details
`SAPSystemEngineMetricID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP system engine metric name table.
`SAPEngineID`	*Type:* integer. Key The unique identifier of an SAP engine.
`SAPEngineMetricID`	*Type:* integer. Key The unique identifier of an SAP engine metric.
`SAPSystemID`	*Type:* integer. Key The unique identifier of an SAP system.
`SAPEnginePeriodTypeID`	*Type:* integer. Key A unique identifier for the SAP engine period type.
`MetricValue`	*Type:* decimal The value of the application engine metric.

Database Column	Details
`PeriodStartDate`	*Type:* datetime. Key. Nullable The start date of the SAP application engine metric calculation period.
`PeriodEndDate`	*Type:* datetime. Key. Nullable The end date of the SAP application engine metric calculation period.
`CalculationDate`	*Type:* datetime. Key. Nullable The date the SAP application engine metric calculation was performed.

SAPSystemEnvironment Table

This table stores SAP System Environment.

Table 603: Database columns for `SAPSystemEnvironment` table

Database Column	Details
`SAPSystemEnvironmentID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP System Environment.
`EnvironmentCode`	*Type:* text (max 1 characters). Key A unique lookup for each `SAPSystemEnvironment`. Possible values and the corresponding default strings are: • P = Production • T = Test • C = Customizing • D = Demo • E = Training/Education • S = SAP reference
`ResourceName`	*Type:* text (max 256 characters) A localizable resource string representing an SAP system environment name. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the SAP system environment resource string has no translation.

SAPSystemGroup Table

This table stores the data specific to the definition of SAP system groups.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 604: Database columns for `SAPSystemGroup` table

Database Column	Details
`SAPSystemGroupID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP system group.
`GroupName`	*Type:* text (max 128 characters). Key The name of the SAP system group.
`GroupDescription`	*Type:* text. Nullable A more detailed description of the SAP system group.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to the system landscape that the SAP system group belongs to.
`ParentSAPSystemGroupID`	*Type:* integer. Key. Nullable Foreign key to the SAP system group that is its parent. This field will be null if the SAP system group is itself a top level SAP system group.
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.
`UpdatedDate`	*Type:* datetime The date and time the system landscape was last updated.

SAPSystemLandscape Table

This table stores the data specific to the definition of system landscapes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 605: Database columns for `SAPSystemLandscape` table

Database Column	Details
`SAPSystemLandscapeID`	*Type:* integer. Key. Generated ID A unique identifier for the system landscape.
`LandscapeUID`	*Type:* text (max 128 characters). Key A global unique identifier for the system landscape.
`LandscapeName`	*Type:* text (max 128 characters) A unique identifier for the system landscape.
`LandscapeDescription`	*Type:* text. Nullable A more detailed description of the SAP system group.
`SAPSystemLandscape StatusID`	*Type:* integer Identifies whether this system landscape is actively being used in the license optimization process.
`LocationID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise location associated with this landscape. Foreign key to the `GroupEx` table.
`BusinessUnitID`	*Type:* text (max 128 characters). Key. Nullable Any corporate unit in the enterprise associated with this landscape. Foreign key to the `GroupEx` table.
`CostCenterID`	*Type:* text (max 128 characters). Key. Nullable Any cost center in the enterprise associated with this landscape. Foreign key to the `GroupEx` table.
`CategoryID`	*Type:* text (max 128 characters). Key. Nullable Any enterprise category associated with this landscape. Foreign key to the `GroupEx` table.
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.
`UpdatedDate`	*Type:* datetime The date and time the system landscape was last updated.

Database Column	Details
SecurityTypeID	*Type:* integer Security type for this object. Foreign key to the SecurityType table.
CanRebalanceLicenseTypes	*Type:* boolean Indicates whether license types can be rebalanced to use excess purchases of higher license types.

SAPSystemLandscapeEngine Table

This table stores an engine in the system landscape.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 606: Database columns for SAPSystemLandscapeEngine table

Database Column	Details
SAPSystemLandscape EngineID	*Type:* integer. Key. Generated ID A unique identifier for the SAP system landscape engine table.
SAPSystemLandscapeID	*Type:* integer. Key Foreign key to the system landscape that the SAP engine belongs to.
EngineUID	*Type:* text (max 128 characters). Key A global unique identifier for the SAP engine in a system landscape.
EngineName	*Type:* text (max 128 characters) Name of engine.
EngineDescription	*Type:* text. Nullable Description of engine.
ApplicationID	*Type:* integer. Nullable SAP internal application ID
IsActive	*Type:* boolean Whether or not the engine is active for inclusion in license position.
NumberPurchased	*Type:* integer. Nullable
UnitPrice	*Type:* currency. Nullable
UnitPriceRateID	*Type:* integer. Nullable

Database Column	Details
SAPContentEngineID	*Type:* integer. Key. Nullable A unique identifier for the SAP content engine table.
SAPContentEngineRuleID	*Type:* integer. Key. Nullable A unique identifier for the SAP content engine rule table.
CustomRuleDefinition	*Type:* text. Nullable Custom rule definition for calculating consumption of an engine.
CustomTotalConsumption	*Type:* integer. Nullable Self-declared total consumption.
UseCustomTotalConsumption	*Type:* boolean Use CustomTotalConsumption
ConsumptionUnit	*Type:* text. Nullable Unit description to describe the consumption amount.
CreationUser	*Type:* text (max 256 characters) The user who created the system landscape.
CreationDate	*Type:* datetime The data and time the system landscape was created.
UpdatedUser	*Type:* text (max 256 characters) The last user who update the system landscape.
UpdatedDate	*Type:* datetime The date and time the system landscape was last updated.

SAPSystemLandscapeEngineMapping Table

This table stores mapping between SAP system landscape engines to either System Landscapes, System Groups or SAP systems.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 607: Database columns for `SAPSystemLandscapeEngineMapping` table

Database Column	Details
SAPSystemLandscape EngineMappingID	*Type:* integer. Key. Generated ID A unique identifier for the SAP system landscape engine mapping.

Database Column	Details
`SAPSystemLandscape EngineID`	*Type:* integer. Key Foreign key to SAPSystemLandscapeEngine ID
`SAPSystemLandscapeID`	*Type:* integer. Key. Nullable Foreign key to System Landscape ID
`SAPSystemGroupID`	*Type:* integer. Key. Nullable Foreign key to System Group ID.
`SAPSystemID`	*Type:* integer. Key. Nullable Foreign key to the SAP system.
`CreationUser`	*Type:* text (max 256 characters) The user who created the system landscape.
`CreationDate`	*Type:* datetime The data and time the system landscape was created.
`UpdatedUser`	*Type:* text (max 256 characters) The last user who update the system landscape.
`UpdatedDate`	*Type:* datetime The date and time the system landscape was last updated.

SAPSystemLandscapeLicenseType Table

This table stores SAP license types belonging to SAP system landscapes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 608: Database columns for `SAPSystemLandscapeLicenseType` table

Database Column	Details
`SAPSystemLandscape LicenseTypeID`	*Type:* integer. Key. Generated ID A unique identifier for SAP system landscape license type.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to SAP system landscape.
`Identifier`	*Type:* text (max 2 characters). Key The SAP license type identifier.

Database Column	Details
`ShortName`	*Type:* text (max 128 characters). Nullable The SAP license type short name.
`LongName`	*Type:* text (max 256 characters). Nullable The SAP license type long name.
`Active`	*Type:* boolean Indicate whether the SAP license is active or not.
`IsS4HanaRecommended`	*Type:* boolean Indicates whether the license type is included in Recommended license types for S/4HANA in License Assignment Rule.
`NumberPurchased`	*Type:* integer. Nullable Number purchased.
`UnitPrice`	*Type:* currency. Nullable Unit price of an SAP license type.
`UnitPriceRateID`	*Type:* integer. Nullable The unit price rate of an SAP license type.
`CreationUser`	*Type:* text (max 128 characters). Nullable The user who created the SAP license type.
`CreationDate`	*Type:* datetime The data and time the SAP license type was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The last user who update the SAP license type.
`UpdatedDate`	*Type:* datetime The date and time the SAP license type was last updated.
`AllowLicenseBalancing`	*Type:* boolean Indicates whether license types can be rebalanced to use excess purchases of higher license types.

SAPSystemLandscapeLicenseTypeHierarchy Table

This table stores the SAP license hierarchy for an SAP system landscape.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 609: Database columns for `SAPSystemLandscapeLicenseTypeHierarchy` table

Database Column	Details
`SAPSystemLandscape LicenseTypeHierarchyID`	*Type:* integer. Key. Generated ID A unique identifier for the license type hierarchy
`SAPSystemLandscape LicenseTypeID`	*Type:* integer. Key A parent system landscape license type. Foreign key to SAP system landscape license type.
`ChildSAPSystem LandscapeLicenseTypeID`	*Type:* integer. Key A child system landscape license type. Foreign key to SAP system landscape license type.

SAPSystemLandscapeLicenseTypeImport Table

This table stores the imported SAP license type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 610: Database columns for `SAPSystemLandscapeLicenseTypeImport` table

Database Column	Details
`SAPSystemLandscape LicenseTypeImportID`	*Type:* integer. Key. Generated ID A unique identifier for the imported SAP license type.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to SAP system landscape.
`SAPSystemID`	*Type:* integer. Key Foreign key to SAP system
`SystemName`	*Type:* text (max 128 characters). Nullable The SAP system name.
`ImportUser`	*Type:* text (max 128 characters). Nullable The user who imported the SAP license type
`ImportDate`	*Type:* datetime The data and time the SAP license type was imported

SAPSystemLandscapeStatus Table

This table stores SAP System Landscape status.

Table 611: Database columns for `SAPSystemLandscapeStatus` table

Database Column	Details
`SAPSystemLandscape StatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `SAPSystemLandscapeStatus`. Possible values and the corresponding default strings are: • 1 = Inactive • 2 = Active • 3 = Archived • 4 = Simulation
`ResourceName`	*Type:* text (max 256 characters). Key A localizable resource string representing an SAP System Landscape status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

SAPSystemMigrationStatus Table

This table stores SAP system migration status.

Table 612: Database columns for `SAPSystemMigrationStatus` table

Database Column	Details
`SAPSystemMigration StatusID`	*Type:* integer. Key. Generated ID A unique identifier for each `SAPSystemMigrationStatusID`. Possible values and the corresponding default strings are: • 1 = NotPlanned • 2 = Planned • 3 = InProgress • 4 = Completed

Database Column	Details
`ResourceName`	*Type:* text (max 256 characters). Key A localizable resource string representing a SAP system migration status. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the status resource string has no translation.

SAPSystemModule Table

This table stores the modules used in SAP and the system they are used on.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 613: Database columns for `SAPSystemModule` table

Database Column	Details
`SAPSystemModuleID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP system module table.
`SAPSystemID`	*Type:* integer. Key The unique identifier of an SAP system.
`SAPModuleID`	*Type:* integer. Key The unique identifier of an SAP module.

SAPSystemObject Table

This table stores objects belonging to SAP systems

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 614: Database columns for `SAPSystemObject` table

Database Column	Details
`SAPSystemObjectID`	*Type:* integer. Generated ID A unique identifier for the SAP system object

Database Column	Details
`SAPSystemID`	*Type:* integer. Key Foreign key to the SAP system that the object belongs to.
`SAPObjectID`	*Type:* integer. Key Foreign key to the SAP object.

SAPSystemPriceList Table

This table stores the SAP system price list.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 615: Database columns for `SAPSystemPriceList` table

Database Column	Details
`SAPSystemPriceListID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP system price list.
`SAPSystemID`	*Type:* integer. Key Foreign key to the system that the price list belongs to.
`PriceListID`	*Type:* text (max 2 characters). Key SAP Price List ID
`DefaultLicenseType`	*Type:* text (max 2 characters). Nullable LicenseType associated to this price list
`IsActive`	*Type:* boolean Indicates whether the price list is active or not active.
`Surcharge`	*Type:* boolean Indicates whether the price list affected by surcharge.

SAPSystemPriceListName Table

This table stores the SAP system price name in multiple languages.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 616: Database columns for `SAPSystemPriceListName` table

Database Column	Details
`SAPSystemPriceListNameID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP system price list name.
`SAPSystemPriceListID`	*Type:* integer. Key Foreign key to the SAP price list.
`Language`	*Type:* text (max 4 characters) A unique code to identify the language.
`PriceListName`	*Type:* text (max 128 characters). Nullable The name of the SAP price list.

SAPSystemRFCConnectionSummary Table

This table stores the link between SAP System and RFC Consumption.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 617: Database columns for `SAPSystemRFCConnectionSummary` table

Database Column	Details
`SAPSystemRFCConnection SummaryID`	*Type:* integer. Key. Generated ID A unique identifier.
`SAPSystemID`	*Type:* integer. Key The Non-SAP system
`SAPRFCConnectionSummaryID`	*Type:* integer. Key The RFC consumption.

SAPSystemRoleType Table

This table stores SAP System Role Type.

Table 618: Database columns for `SAPSystemRoleType` table

Database Column	Details
`SAPSystemRoleTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP System Role type.
`TypeName`	*Type:* text (max 64 characters). Key A unique lookup for each `SAPSystemRoleType`. Possible values and the corresponding default strings are: • AdminModule • IndependentSAPSystem • DependentSAPSystem
`ResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP System Role type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the SAP System Role type resource string has no translation.

SAPSystemType Table

This table stores SAP system type.

Table 619: Database columns for `SAPSystemType` table

Database Column	Details
`SAPSystemTypeID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP system type.
`TypeName`	*Type:* text (max 64 characters). Key A unique lookup for each `SAPSystemType`. Possible values and the corresponding default strings are: • SAP • NonSAP
`ResourceName`	*Type:* text (max 256 characters). Nullable A localizable resource string representing an SAP system type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the SAP system type resource string has no translation.

SAPTransactionProfile Table

This table stores SAP transaction profiles.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 620: Database columns for `SAPTransactionProfile` table

Database Column	Details
`SAPTransactionProfileID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP transaction profile.
`SAPSystemLandscapeID`	*Type:* integer. Key Foreign key to SAP system landscapes the SAP transaction profile belongs to.
`TransactionProfileName`	*Type:* text (max 128 characters) Name of the SAP transaction profile
`Description`	*Type:* text. Nullable Description of the SAP transaction profile
`CreationUser`	*Type:* text (max 128 characters). Nullable The user who created the SAP transaction profile.
`CreationDate`	*Type:* datetime The data and time the SAP transaction profile was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The last user who update the SAP transaction profile.
`UpdatedDate`	*Type:* datetime The date and time the SAP transaction profile was last updated.

SAPTransactionProfileObject Table

This table stores the linking between SAP transaction profile and SAP object.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 621: Database columns for `SAPTransactionProfileObject` table

Database Column	Details
`SAPTransactionProfileObjectID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP transaction profile object
`SAPTransactionProfileID`	*Type:* integer. Key Foreign key to an SAP transaction profile.
`ObjectName`	*Type:* text (max 128 characters) The SAP object name
`Description`	*Type:* text. Nullable The SAP object description
`IsTransaction`	*Type:* boolean Indicates whether the object is of type Transaction
`IsReport`	*Type:* boolean Indicates whether the object is of type Report
`IsJob`	*Type:* boolean Indicates whether the object is of type Job
`IsExcludedFromProfile`	*Type:* boolean Indicates whether the object is marked as excluded from this profile.
`CreationUser`	*Type:* text (max 128 characters). Nullable The user who created the profile and object link.
`CreationDate`	*Type:* datetime The data and time the profile and object link was created.
`UpdatedUser`	*Type:* text (max 128 characters). Nullable The last user who update the profile and object link.
`UpdatedDate`	*Type:* datetime The date and time the profile and object link was last updated.
`IsNonSAP`	*Type:* boolean Indicates whether the object is of type Non-SAP

SAPUser Table

This table stores the data specific to the definition of SAP users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 622: Database columns for `SAPUser` table

Database Column	Details
`SAPUserID`	*Type:* integer. Key. Generated ID A unique identifier for the SAP user.
`SAPSystemID`	*Type:* integer. Key Foreign key to the system that the user belongs to.
`UserName`	*Type:* text (max 256 characters). Key The SAP user's username.
`FirstName`	*Type:* text (max 40 characters). Nullable The SAP user's first name.
`LastName`	*Type:* text (max 40 characters). Nullable The SAP user's last name.
`ValidFrom`	*Type:* datetime. Nullable The date that the SAP user is valid from on the SAP system.
`ValidTo`	*Type:* datetime. Nullable The date that the SAP user is valid to on the SAP system.
`UserType`	*Type:* text (max 1 characters). Nullable The type of user the SAP user is.
`LicenseType`	*Type:* text (max 2 characters). Nullable The type of license assigned to the SAP user.
`UserGroup`	*Type:* text (max 12 characters). Nullable The user group the SAP user belongs to.
`LastLogonDate`	*Type:* datetime. Nullable The date when the SAP user last logged on to the SAP system.
`IsDeveloper`	*Type:* boolean Indicates whether the SAP user is a developer or not.
`UserCreationDate`	*Type:* datetime. Nullable The date the SAP user was created.
`EmailAddress`	*Type:* text (max 128 characters). Nullable The SAP user's email address.

Database Column	Details
`TelephoneNumber`	*Type:* text (max 30 characters). Nullable The SAP user's telephone number.
`TelephoneExtension`	*Type:* text (max 10 characters). Nullable The SAP user's telephone extension.
`AccountID`	*Type:* text (max 12 characters). Nullable The SAP user's account ID.
`CostCenter`	*Type:* text (max 8 characters). Nullable The cost center the SAP user belongs to.
`CompanyName1`	*Type:* text (max 40 characters). Nullable The name of the company the SAP user belongs to.
`CompanyName2`	*Type:* text (max 40 characters). Nullable The name of a second company the SAP user belongs to.
`Department`	*Type:* text (max 40 characters). Nullable The department the SAP user belongs to.
`UserFunction`	*Type:* text (max 40 characters). Nullable
`UserLockStatus`	*Type:* integer. Nullable User lock status.
`SpecialVersionAssignment`	*Type:* text (max 2 characters). Nullable
`CountrySurcharge`	*Type:* text (max 4 characters). Nullable
`RepresentativeFromDate`	*Type:* datetime. Nullable
`RepresentativeToDate`	*Type:* datetime. Nullable
`IsDeleted`	*Type:* boolean Indicated whether the SAP user has been deleted or not.
`ChargeableUserClient`	*Type:* text (max 32 characters). Nullable
`ChargeableUserSysID`	*Type:* text (max 32 characters). Nullable
`ChargeableUserName`	*Type:* text (max 12 characters). Nullable
`RemoteServerUserName`	*Type:* text (max 64 characters). Nullable Remote server user name

SAPUserRole Table

This table stores SAP users and its SAP role memberships

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 623: Database columns for `SAPUserRole` table

Database Column	Details
`SAPUserRoleID`	*Type:* integer. Key. Generated ID A unique identifier for SAP user role.
`SAPUserID`	*Type:* integer. Key Foreign key to the SAP user that the role belongs to.
`SAPRoleID`	*Type:* integer. Key Foreign key to SAP role.
`ValidFrom`	*Type:* datetime. Nullable The date that the SAP role is valid from.
`ValidTo`	*Type:* datetime. Nullable The date that the SAP role is valid to.

SAPUserType Table

This table stores SAP User type.

Table 624: Database columns for `SAPUserType` table

Database Column	Details
`SAPUserTypeID`	*Type:* integer. Key. Generated ID

Database Column	Details
`UserTypeCode`	*Type:* text (max 1 characters). Key A unique lookup for each `SAPUserType`. Possible values and the corresponding default strings are: • A = Dialog • B = System • C = Communication Data • D = BDC • L = Reference • S = Service
`ResourceName`	*Type:* text (max 256 characters) A localizable resource string representing an SAP user type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the system type resource string has no translation.

ManageSoft Tables

The complete set of database tables documented here includes:

- DatabaseConfiguration table (see DatabaseConfiguration Table)

DatabaseConfiguration Table

The `DatabaseConfiguration` table contains configuration properties for the FlexNet Manager Suite database tables, which are used for ongoing maintenance of the database.

Table 625: Database columns for `DatabaseConfiguration` table

Database Column	Details
`Property`	*Type:* text (max 32 characters). Key The name of the property.
`Value`	*Type:* text (max 256 characters) The value of the property.
`Created`	*Type:* datetime The date and time the property was created.

Database Column	Details
LastUpdate	*Type:* datetime The date and time the property was last updated.

ReferenceData Tables

The complete set of database tables documented here includes:

- Country table (see Country Table)
- Language table (see Language Table)
- Locale table (see Locale Table)

Country Table

Stores country information, including their ISO country code and English names.

Table 626: Database columns for Country table

Database Column	Details
CountryCode	*Type:* text (max 2 characters). Key The two letter country code.
Name	*Type:* text (max 128 characters). Key The english name of the country.

Language Table

Stores language information, including their English names, and various forms of language id.

Table 627: Database columns for Language table

Database Column	Details
LangCode3	*Type:* text (max 3 characters). Key The three letter language code.
LangCode2	*Type:* text (max 2 characters). Nullable The two letter language code.
EnglishName	*Type:* text (max 128 characters). Key The english name of the language.

Database Column	Details
`LocalName`	*Type:* text (max 128 characters). Nullable The name of the language, written in the local language.
`MSLanguageID`	*Type:* integer. Nullable The Microsoft language id, as specified in winnt.h in the Platform SDK.

Locale Table

Stores locale information, which consists of country and language combinations. Use the `LocaleCode` column as the foreign key into this table.

Table 628: Database columns for `Locale` table

Database Column	Details
`LocaleCode`	*Type:* text (max 6 characters). Key A combination of the language code and country code, separated by a hyphen. If there is no country code, then there will be no hyphen added. This column MUST have the correct value when inserted, based on the values of the language and country codes.
`LangCode3`	*Type:* text (max 3 characters). Key The three letter language code.
`CountryCode`	*Type:* text (max 2 characters). Key. Nullable The two letter country code.
`LocaleName`	*Type:* text (max 128 characters) The name of the locale. For example, “English (United States)”.
`MSLocaleID`	*Type:* integer. Nullable The Microsoft identifier for the locale. For example, 1033 for English (United States).

Rights Tables

The complete set of database tables documented here includes:

- ActionClass table (see ActionClass Table)
- PartitionType table (see PartitionType Table)
- Resource table (see Resource Table)

ActionClass Table

The types of action on a `Resource` for which rights may be granted or denied.

Table 629: Database columns for `ActionClass` table

Database Column	Details
`ActionClassID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`ActionClassName`	*Type:* text (max 16 characters). Key The name of the `ActionClass`.

PartitionType Table

Some secured `Resource`s may be partitioned. Partitions are used to grant rights to one part of a `Resource` excluding other parts, for example limiting rights so that the operator can access only certain distribution servers, organizational units, or areas in the software library. There are three types of partitioning, defined by entries in this table.

Table 630: Database columns for `PartitionType` table

Database Column	Details
`PartitionTypeID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`PartitionTypeName`	*Type:* text (max 32 characters). Key Name of the `PartitionType`.

Resource Table

Access rights are granted to the `Resource`s defined in this table.

Table 631: Database columns for `Resource` table

Database Column	Details
`ResourceID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`ResourceName`	*Type:* text (max 16 characters). Key Name of the `Resource`.

Database Column	Details
`PartitionTypeID`	*Type:* integer. Nullable If not NULL, the type of partitioning used with this `Resource`.

Targeting Tables

The complete set of database tables documented here includes:

- TargetType table (see TargetType Table)

TargetType Table

The `TargetType` table contains a row for each type of object that can be targeted in FlexNet Manager Suite.

Table 632: Database columns for `TargetType` table

Database Column	Details
`TargetTypeID`	*Type:* integer. Key. Generated ID The ID for the target type: • Computers • Users • Group • DistributionLocation • DistributionServer • Organization • Assets • Contracts • Purchase orders • Software licenses • Software titles • Compliance computers • Compliance users • Operators • SAP system landscapes • SAP systems • SAP rule sets • Discovered devices • Beacon • Vendor • Device • Rule • Inventory connection • FNMP Server • Fast Import • OLE DB Connection • ORACLE Connection

Database Column	Details
	• XML • Intermediate File • ADSI Connection • Web Service • SQL Connection • Software Title Evidence • FNMEA Agent • Installed Software • Baseline Import • Available Package • Client ARL
`TargetTypeName`	*Type:* text (max 256 characters). Key The name of the target type.

Tenants Tables

The complete set of database tables documented here includes:

- FlexeraLicense table (see FlexeraLicense Table)
- Tenant table (see Tenant Table)

FlexeraLicense Table

The `FlexeraLicense` table contains the encoded contents of the Flexera Software licenses required for the tenants in the system. This table is also used by the system in the single-tenant setup where there is only one tenant.

Table 633: Database columns for `FlexeraLicense` table

Database Column	Details
`TenantUID`	*Type:* text (max 40 characters). Key The unique identifier of a tenant. A reference to the `Tenant` to which this license is attached.

Database Column	Details
License	*Type:* text The encoded contents of the Flexera Software license attached to a particular Tenant.
LicenseChecksum	*Type:* integer. Key The check sum of the license.
LicenseDetails	*Type:* XML. Nullable XML definition of the license details

Tenant Table

The Tenant table contains the details of each tenant in multitenant FlexNet Manager Suite database tables.

Table 634: Database columns for Tenant table

Database Column	Details
TenantID	*Type:* integer. Key. Generated ID The tenant ID in a multi-tenant database.
TenantUID	*Type:* text (max 40 characters). Key The unique identifier of a tenant. This identifier is used to identify the tenant in environments where tenant information is stored on multiple databases.
TenantName	*Type:* text (max 256 characters). Key The name of the tenant.
TenantDomain	*Type:* text (max 20 characters). Nullable The sub-domain to use for the tenant.
Comments	*Type:* text. Nullable Operator comments about this tenant record.
CreationUser	*Type:* text (max 128 characters). Nullable The operator who created the tenant record.
CreationDate	*Type:* datetime The date the tenant record was created.
UpdatedUser	*Type:* text (max 128 characters). Nullable The name of the operator who last updated the tenant record.
UpdatedDate	*Type:* datetime. Nullable The date the tenant record was last updated.

Database Column	Details
`OrganizationID`	*Type:* integer. Nullable The unique Flexera organization ID.

3

Dashboard Database Schema

This chapter describes the schema used for customized dashboards using the Analytics feature (powered by Cognos) of FlexNet Manager Suite.

There are three separate data models related to IBM Cognos within FlexNet Manager Suite:

- A model for use when customizing dashboards for FlexNet Manager Suite, which is the subject of this chapter
- An operational model for reporting on live data (this model is not separately documented)
- The dimensional data model for reporting on data that changes over time (see DataWarehouse Database Schema).

Dashboard Tables

The complete set of database tables documented here includes:

- ApplicationAction_CODE table (see ApplicationAction_CODE Table)
- ApplicationCategory_CODE table (see ApplicationCategory_CODE Table)
- ApplicationClassification_CODE table (see ApplicationClassification_CODE Table)
- Application_DIM table (see Application_DIM Table)
- Application_Measurement_FACT table (see Application_Measurement_FACT Table)
- AssetStatus_CODE table (see AssetStatus_CODE Table)
- AssetType_CODE table (see AssetType_CODE Table)
- Asset_Activity_FACT table (see Asset_Activity_FACT Table)
- Asset_FACT table (see Asset_FACT Table)
- ContractStatus_CODE table (see ContractStatus_CODE Table)
- ContractType_CODE table (see ContractType_CODE Table)
- Contract_DIM table (see Contract_DIM Table)

- Contract_FACT table (see Contract_FACT Table)
- CurrencyCurrentConversion_FACT table (see CurrencyCurrentConversion_FACT Table)
- Currency_DIM table (see Currency_DIM Table)
- DiscoveredDevices_Activity_FACT table (see DiscoveredDevices_Activity_FACT Table)
- Installation_Activity_FACT table (see Installation_Activity_FACT Table)
- Inventory_DuplicateHostName_FACT table (see Inventory_DuplicateHostName_FACT Table)
- Inventory_DuplicateSerialNumber_FACT table (see Inventory_DuplicateSerialNumber_FACT Table)
- Inventory_VirtualizationType_FACT table (see Inventory_VirtualizationType_FACT Table)
- LicenseComplianceStatus_CODE table (see LicenseComplianceStatus_CODE Table)
- LicenseStatus_CODE table (see LicenseStatus_CODE Table)
- LicenseType_CODE table (see LicenseType_CODE Table)
- License_DIM table (see License_DIM Table)
- License_Position_FACT table (see License_Position_FACT Table)
- PurchaseStatus_CODE table (see PurchaseStatus_CODE Table)
- PurchaseType_CODE table (see PurchaseType_CODE Table)
- Purchase_DIM table (see Purchase_DIM Table)
- Purchase_Latest_FACT table (see Purchase_Latest_FACT Table)
- Purchase_Spend_FACT table (see Purchase_Spend_FACT Table)
- Purchase_Unprocessed_FACT table (see Purchase_Unprocessed_FACT Table)
- ResourceString_CODE table (see ResourceString_CODE Table)
- VMType_CODE table (see VMType_CODE Table)

ApplicationAction_CODE Table

`ApplicationAction_CODE` is an enumerated code table for application action status.

Table 635: Database columns for `ApplicationAction_CODE` table

Database Column	Details
`ActionStatusID`	*Type:* integer. Key A unique identifier for application action status. Possible values and the corresponding default strings are: • 1 = Unmanaged (recently created application, not yet categorized) • 2 = Authorized (application is authorized for use in the enterprise) • 3 = Unauthorized (application is not authorized for use) • 4 = Ignored (application will not be tracked by the enterprise) • 5 = Inactive (application is not in use in the enterprise). • 6 = Deferred (application installed in enterprise but marked for later attention).
`ActionStatus_en`	*Type:* text (max 1000 characters) Action status of the application in English.
`ActionStatus_de`	*Type:* text (max 1000 characters) Action status of the application in German.
`ActionStatus_fr`	*Type:* text (max 1000 characters) Action status of the application in French.
`ActionStatus_ja`	*Type:* text (max 1000 characters) Action status of the application in Japanese.
`ActionStatus_es`	*Type:* text (max 1000 characters). Nullable Action status of the application in Spanish.

ApplicationCategory_CODE Table

`ApplicationCategory_CODE` is an enumerated code table for UNSPSC categories.

Table 636: Database columns for `ApplicationCategory_CODE` table

Database Column	Details
`CategoryID`	*Type:* integer. Key Primary key of the category.
`Category_en`	*Type:* text (max 1000 characters) Category (UNSPSC) of the application in English.

Database Column	Details
`Category_de`	*Type:* text (max 1000 characters) Category (UNSPSC) of the application in German.
`Category_fr`	*Type:* text (max 1000 characters) Category (UNSPSC) of the application in French.
`Category_ja`	*Type:* text (max 1000 characters) Category (UNSPSC) of the application in Japanese.
`Category_es`	*Type:* text (max 1000 characters). Nullable Category (UNSPSC) of the application in Spanish.

ApplicationClassification_CODE Table

`ApplicationClassification_CODE` is an enumerated code table for application classifications.

Table 637: Database columns for `ApplicationClassification_CODE` table

Database Column	Details
`ClassificationID`	*Type:* integer. Key A unique identifier for application classifications. Possible values and the corresponding default strings are: • 1 = Shareware • 2 = Freeware • 3 = Commercial • 4 = Update • 5 = Malware • 6 = Beta • 7 = XRated • 8 = None • 9 = Component
`Classification_en`	*Type:* text (max 1000 characters) Classification of the application in English.
`Classification_de`	*Type:* text (max 1000 characters) Classification of the application in German.

Database Column	Details
`Classification_fr`	*Type:* text (max 1000 characters) Classification of the application in French.
`Classification_ja`	*Type:* text (max 1000 characters) Classification of the application in Japanese.
`Classification_es`	*Type:* text (max 1000 characters). Nullable Classification of the application in Spanish.

Application_DIM Table

`Application_DIM` is a dimension table storing applications (specific edition and version of a product).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 638: Database columns for `Application_DIM` table

Database Column	Details
`ApplicationID`	*Type:* integer. Key Primary key of the application.
`FullName`	*Type:* text (max 512 characters). Key Full name of the application.
`PublisherName`	*Type:* text (max 200 characters). Key Publisher of the application.
`ProductName`	*Type:* text (max 200 characters). Key Product name of the application.
`EditionName`	*Type:* text (max 50 characters) Edition of the application.
`VersionName`	*Type:* text (max 50 characters) Version of the application.
`CategoryID`	*Type:* integer. Key. Nullable Category ID (UNSPSC) of the application.
`ClassificationID`	*Type:* integer. Key Classification ID of the application.

Database Column	Details
`ActionStatusID`	*Type:* integer. Key Action status ID of the application.
`IsManaged`	*Type:* boolean Whether the application is a managed or unmanaged application.
`StartOfLifeDate`	*Type:* datetime. Key. Nullable Start of life Date.
`ReleaseDate`	*Type:* datetime. Key. Nullable The date the application was released.
`EndOfSalesDate`	*Type:* datetime. Key. Nullable End of sales Date.
`SupportedUntil`	*Type:* datetime. Key. Nullable The date the application will be supported.
`ExtendedSupportUntil`	*Type:* datetime. Key. Nullable The date the application will be supported, in extended case.
`EndOfLifeDate`	*Type:* datetime. Key. Nullable End of life Date.

Application_Measurement_FACT Table

`Application_Measurement_FACT` is a fact table storing application measurements by operator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 639: Database columns for `Application_Measurement_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`ApplicationID`	*Type:* integer. Key Foreign key to the application.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable Foreign key to SoftwareLicense_DIM.

Database Column	Details
`InstallCount`	*Type:* big integer Number of installations of this application covered by this license.
`UnlicensedInstallCount`	*Type:* big integer Number of installations of this application not covered by this license.

AssetStatus_CODE Table

`AssetStatus_CODE` is an enumerated code table for Asset status.

Table 640: Database columns for `AssetStatus_CODE` table

Database Column	Details
`AssetStatusID`	*Type:* integer. Key A unique identifier for Asset status. Possible values and the corresponding default strings are: • 1 = Purchased • 2 = In Storage • 3 = Installed • 4 = Retired • 5 = Disposed • 6 = Other.
`AssetStatus_en`	*Type:* text (max 1000 characters) Status of the Asset in English.
`AssetStatus_de`	*Type:* text (max 1000 characters) Status of the Asset in German.
`AssetStatus_fr`	*Type:* text (max 1000 characters) Status of the Asset in French.
`AssetStatus_ja`	*Type:* text (max 1000 characters) Status of the Asset in Japanese.
`AssetStatus_es`	*Type:* text (max 1000 characters). Nullable Status of the Asset in Spanish.

AssetType_CODE Table

`AssetType_CODE` is an enumerated code table for Asset type.

Table 641: Database columns for `AssetType_CODE` table

Database Column	Details
`AssetTypeID`	*Type:* integer. Key A unique identifier for Asset type. Possible values and the corresponding default strings are: • 1 = Workstation • 2 = Server • 3 = Monitor • 4 = Desk • 5 = Chair • 6 = Printer • 7 = Router • 8 = Switch • 9 = Telephone • 10 = Cell phone • 11 = Laptop. • 12 = Mobile Device.
`AssetType_en`	*Type:* text (max 1000 characters) Type of the Asset in English.
`AssetType_de`	*Type:* text (max 1000 characters) Type of the Asset in German.
`AssetType_fr`	*Type:* text (max 1000 characters) Type of the Asset in French.
`AssetType_ja`	*Type:* text (max 1000 characters) Type of the Asset in Japanese.
`AssetType_es`	*Type:* text (max 1000 characters). Nullable Type of the Asset in Spanish.

Asset_Activity_FACT Table

`Asset_Activity_FACT` is a fact table storing assets that had activity in the last 90 days. Row count : 90 (days) * combination of AssetStatusID and AssetTypeID rows.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 642: Database columns for `Asset_Activity_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`AgeInDay`	*Type:* integer. Key The days relative to date when the table is last updated.
`AssetStatusID`	*Type:* integer. Key Asset status ID of the asset.
`AssetTypeID`	*Type:* integer. Key Asset type ID of the asset.
`ActivityDate`	*Type:* datetime The date the activity occurred.
`NewCount`	*Type:* integer Number of new assets created on this date.
`LastReportedCount`	*Type:* integer Number of assets when its inventory is last reported on this date.
`OracleDBLastReportedCount`	*Type:* integer Number of Oracle database assets when retired or disposed inventory is reported on this date.

Asset_FACT Table

`Asset_FACT` is a fact table storing the asset count by operator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 643: Database columns for `Asset_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`AssetStatusID`	*Type:* integer. Key Asset status ID.
`AssetTypeID`	*Type:* integer. Key Asset type ID.
`TotalCount`	*Type:* integer Number of assets.

ContractStatus_CODE Table

`ContractStatus_CODE` is a dimension table storing contract status information.

Table 644: Database columns for `ContractStatus_CODE` table

Database Column	Details
`ContractStatusID`	*Type:* integer. Key A unique identifier for Contract status. Possible values and the corresponding default strings are: • 1 = Active • 2 = Archived • 3 = Draft • 4 = Suspended • 5 = Cancelled • 6 = Expired • 7 = Completed
`ContractStatus_en`	*Type:* text (max 1000 characters) Compliance status of the license in English.
`ContractStatus_de`	*Type:* text (max 1000 characters) Compliance status of the license in German.
`ContractStatus_fr`	*Type:* text (max 1000 characters) Compliance status of the license in French.

Database Column	Details
`ContractStatus_ja`	*Type:* text (max 1000 characters) Compliance status of the license in Japanese.
`ContractStatus_es`	*Type:* text (max 1000 characters) Compliance status of the license in Spanish.

ContractType_CODE Table

`ContractType_CODE` is a dimension table storing contract type information.

Table 645: Database columns for `ContractType_CODE` table

Database Column	Details
`ContractTypeID`	*Type:* integer. Key A unique identifier for Contract type. Possible values and the corresponding default strings are: • 1 = General • 2 = Lease • 3 = Hardware Maintenance and Support • 4 = Software License • 5 = Software Maintenance and Support • 6 = Blanket purchase • 7 = Consulting services • 8 = Insurance • 9 = Rent • 10 = Subscription • 11 = Microsoft Business and Services Agreement • 12 = Microsoft Select License Agreement • 13 = Microsoft Select Plus Agreement • 14 = Microsoft Select License Enrollment • 15 = Microsoft Select Plus Affiliate • 16 = Microsoft Enterprise Agreement • 17 = Microsoft Enterprise Subscription Agreement

Database Column	Details
ContractType_en	*Type:* text (max 1000 characters) Contract Type of the Contract in English.
ContractType_de	*Type:* text (max 1000 characters) Contract Type of the Contract in German.
ContractType_fr	*Type:* text (max 1000 characters) Contract Type of the Contract in French.
ContractType_ja	*Type:* text (max 1000 characters) Contract Type of the Contract in Japanese.
ContractType_es	*Type:* text (max 1000 characters) Contract Type of the Contract in Spanish.

Contract_DIM Table

Contract_DIM is a dimension table storing contracts information.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 646: Database columns for Contract_DIM table

Database Column	Details
ContractID	*Type:* integer. Key Primary key to the contract.
ContractNo	*Type:* text (max 60 characters) Contract No of the contract.
ContractName	*Type:* text (max 100 characters) Contract Name of the contract.
VendorName	*Type:* text (max 64 characters). Nullable Vendor Name of the contract.
IsEvergreen	*Type:* boolean If the contract never expires of the contract.
StartDate	*Type:* datetime. Nullable Start date of the contract.

Database Column	Details
`ExpiryDate`	*Type:* datetime. Nullable Expiry date of the contract.
`ReviewDate`	*Type:* datetime. Nullable Review date of the contract.
`RenewalDate`	*Type:* datetime. Nullable Renewal date of the contract.
`IsLinkedToLicense`	*Type:* boolean If the contract is linked to a license.

Contract_FACT Table

`Contract_FACT` is a fact table storing contracts information per ComplianceOperator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 647: Database columns for `Contract_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`ContractID`	*Type:* integer. Key Foreign key of the contract.
`ContractTypeID`	*Type:* integer Contract type ID of the contract.
`ContractStatusID`	*Type:* integer. Nullable Contract status ID of the contract.
`CurrencyID`	*Type:* integer Currency ID (total value id) of the contract.
`GlobalAmount`	*Type:* currency. Nullable Global amount (total value) of the contract.

CurrencyCurrentConversion_FACT Table

`CurrencyCurrentConversion_FACT` is a fact table storing current currency conversion rate.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 648: Database columns for `CurrencyCurrentConversion_FACT` table

Database Column	Details
`FromCurrencyID`	*Type:* integer. Key Foreign key to the Currency.
`ToCurrencyID`	*Type:* integer. Key Currency to convert to. Foreign key to the Currency.
`ExchangeRate`	*Type:* decimal Current exchange rate.

Currency_DIM Table

`Currency_DIM` is a dimension table storing latest currency exchange rates.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 649: Database columns for `Currency_DIM` table

Database Column	Details
`CurrencyID`	*Type:* integer. Key Primary key of the currency.
`CurrencyCode`	*Type:* text (max 32 characters). Key Code assigned to the currency.
`LongPrefix`	*Type:* text (max 32 characters) Long prefix to display in front of the money value.
`LongSuffix`	*Type:* text (max 32 characters) Long suffix to display after the money value.

Database Column	Details
LongFormat	*Type:* text (max 80 characters) Long format of the currency.
ShortPrefix	*Type:* text (max 32 characters) Short prefix to display in front of the money value.
ShortSuffix	*Type:* text (max 32 characters) Short suffix to display after the money value.
ShortFormat	*Type:* text (max 80 characters) Short format of the currency.
Currency_en	*Type:* text (max 1000 characters) Currency name in English.
Currency_de	*Type:* text (max 1000 characters) Currency name in German.
Currency_fr	*Type:* text (max 1000 characters) Currency name in French.
Currency_ja	*Type:* text (max 1000 characters) Currency name in Japanese.

DiscoveredDevices_Activity_FACT Table

DiscoveredDevices_FACT is a table containing devices discovered in the last 90 days but have no inventory.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 650: Database columns for DiscoveredDevices_Activity_FACT table

Database Column	Details
ComplianceOperatorID	*Type:* integer. Key Foreign key to the operator.
AgeInDay	*Type:* integer. Key The days relative to date when the table is last updated.
ActivityDate	*Type:* datetime The date the activity occurred.

Database Column	Details
`MissingInventoryCount`	*Type:* big integer Number of discovered devices on this date that are missing inventory.

Installation_Activity_FACT Table

`Installation_Activity_FACT` is a fact table storing application installations that have been discovered in the last 90 days. Row count : 90 (days) * combination of ActionStatusID and ClassificationID rows.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 651: Database columns for `Installation_Activity_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`AgeInDay`	*Type:* integer. Key The days relative to date when the table is last updated.
`ActionStatusID`	*Type:* integer. Key Action status ID of the application.
`ClassificationID`	*Type:* integer. Key Classification ID of the application.
`ActivityDate`	*Type:* datetime The date the activity occurred.
`InstallCount`	*Type:* big integer Number of installations on this date.

Inventory_DuplicateHostName_FACT Table

`Inventory_DuplicateHostName_FACT` is a fact table storing duplicate host name and its duplicate count.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 652: Database columns for `Inventory_DuplicateHostName_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`HostName`	*Type:* text (max 256 characters). Key Host name.
`DuplicateCount`	*Type:* integer Duplicate count.

Inventory_DuplicateSerialNumber_FACT Table

`Inventory_DuplicateSerialNumber_FACT` is a fact table storing duplicate serial number and its duplicate count.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 653: Database columns for `Inventory_DuplicateSerialNumber_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`SerialNo`	*Type:* text (max 100 characters). Key Serial number.
`DuplicateCount`	*Type:* integer Duplicate count.

Inventory_VirtualizationType_FACT Table

`Inventory_VirtualizationType_FACT` is a fact table storing the number of virtual machines and Oracle Database servers by type.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 654: Database columns for `Inventory_VirtualizationType_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`VMTypeID`	*Type:* integer. Key Foreign key to the VM type ID.
`VMCount`	*Type:* integer Number of virtual machines by type.
`OracleDBCount`	*Type:* integer Number of Oracle database servers by type.

LicenseComplianceStatus_CODE Table

`LicenseComplianceStatus_CODE` is an enumerated code table for compliance status for a license.

Table 655: Database columns for `LicenseComplianceStatus_CODE` table

Database Column	Details
`ComplianceStatusID`	*Type:* integer. Key A unique identifier for license compliance status. Possible values and the corresponding default strings are: • 1 = Compliant • 2 = In Breach • 3 = Unknown • 4 = Not Tracked.
`ComplianceStatus_en`	*Type:* text (max 1000 characters) Compliance status of the license in English.
`ComplianceStatus_de`	*Type:* text (max 1000 characters) Compliance status of the license in German.
`ComplianceStatus_fr`	*Type:* text (max 1000 characters) Compliance status of the license in French.
`ComplianceStatus_ja`	*Type:* text (max 1000 characters) Compliance status of the license in Japanese.

Database Column	Details
`ComplianceStatus_es`	*Type:* text (max 1000 characters). Nullable Compliance status of the license in Spanish.

LicenseStatus_CODE Table

`LicenseStatus_CODE` is an enumerated code table for license status for a license.

Table 656: Database columns for `LicenseStatus_CODE` table

Database Column	Details
`LicenseStatusID`	*Type:* integer. Key A unique identifier for license status. Possible values and the corresponding default strings are: • 1 = Active • 2 = Retired • 3 = In Stock • 4 = Purchased • 5 = Received
`LicenseStatus_en`	*Type:* text (max 1000 characters) License status of the license in English.
`LicenseStatus_de`	*Type:* text (max 1000 characters) License status of the license in German.
`LicenseStatus_fr`	*Type:* text (max 1000 characters) License status of the license in French.
`LicenseStatus_ja`	*Type:* text (max 1000 characters) License status of the license in Japanese.
`LicenseStatus_es`	*Type:* text (max 1000 characters). Nullable License status of the license in Spanish.

LicenseType_CODE Table

`LicenseType_CODE` is an enumerated code table for software license types.

Table 657: Database columns for `LicenseType_CODE` table

Database Column	Details
`LicenseTypeID`	*Type:* integer. Key A unique identifier for license type. Possible values and the corresponding default strings are: • 1 = Enterprise • 2 = Device • 3 = Node-Locked • 4 = User • 5 = Concurrent User • 6 = Appliance • 7 = Client Server • 8 = OEM • 9 = Evaluation • 10 = Run-Time • 11 = Device (Processor-Limited) • 12 = Site • 13 = Named User • 14 = Device (Core-Limited) • 15 = Core Points • 16 = Oracle Processor • 17 = Oracle Named User Plus • 18 = Processor Points • 19 = Oracle Legacy • 20 = Enterprise Agreement • 21 = SAP Named User • 22 = Microsoft Server Processor • 23 = CAL Legacy • 24 = Tiered Device • 25 = IBM Processor Value Unit • 26 = IBM Authorized User

Database Column	Details
	• 27 = IBM Concurrent User • 28 = IBM Floating User • 29 = Custom Metric • 30 = Processor • 31 = IBM Resource Value Unit • 32 = IBM User Value Unit • 33 = Microsoft Server Core • 34 = Oracle User • 35 = SAP Package • 36 = Microsoft SCCM Client Device • 37 = Microsoft SCCM Client User • 38 = Microsoft Developer Network • 39 = Microsoft Device CAL • 40 = Microsoft User CAL
`LicenseType_en`	*Type:* text (max 1000 characters) License type of the license in English.
`LicenseType_de`	*Type:* text (max 1000 characters) License type of the license in German.
`LicenseType_fr`	*Type:* text (max 1000 characters) License type of the license in French.
`LicenseType_ja`	*Type:* text (max 1000 characters) License type of the license in Japanese.
`LicenseType_es`	*Type:* text (max 1000 characters). Nullable License type of the license in Spanish.

License_DIM Table

`License_DIM` is a dimension table storing software licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 658: Database columns for `License_DIM` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key Primary key of the license.
`LicenseName`	*Type:* text (max 256 characters). Key Name of the license.
`ProductName`	*Type:* text (max 256 characters). Key Product name of the primary application.
`PublisherName`	*Type:* text (max 256 characters). Key Publisher of the primary application.
`EditionName`	*Type:* text (max 60 characters) Edition of the license.
`VersionName`	*Type:* text (max 60 characters) Version of the license.
`LicenseTypeID`	*Type:* integer. Key License type ID of the license.
`ClassificationID`	*Type:* integer. Key Classification ID of the primary application.
`ActionStatusID`	*Type:* integer. Key Action status ID of the primary application.
`IsBundle`	*Type:* boolean Whether this license is a bundle license (contain multiple primary applications).

License_Position_FACT Table

`License_Position_FACT` is a fact table storing license positions by operator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 659: Database columns for `License_Position_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`SoftwareLicenseID`	*Type:* integer. Key Foreign key to the license.
`Entitlements`	*Type:* big integer Total number of entitlements.
`Consumption`	*Type:* big integer Number of entitlements consumed.
`Installs`	*Type:* big integer Number of installations.
`ComplianceStatusID`	*Type:* integer Compliance status ID of the license.
`RiskCount`	*Type:* big integer Number of entitlements at risk (aka in breach).
`OriginalCurrencyID`	*Type:* integer Currency ID of the OriginalCurrencyRiskAmount.
`OriginalCurrencyUnitPrice`	*Type:* decimal Unit price in original currency.
`OriginalCurrencyRisk Amount`	*Type:* decimal Value at risk in original currency.
`SystemCurrencyID`	*Type:* integer Currency ID of the SystemCurrencyRiskAmount.
`SystemCurrencyUnitPrice`	*Type:* decimal Unit price in system currency.
`SystemCurrencyRiskAmount`	*Type:* decimal Value at risk in system currency.
`RiskPercent`	*Type:* decimal Percentage at risk.
`UtilizationPercent`	*Type:* decimal. Nullable Percentage utilization.

Database Column	Details
LicenseStatusID	*Type:* integer. Nullable License status ID of the license.

PurchaseStatus_CODE Table

PurchaseStatus_CODE is a dimension table storing purchase status information.

Table 660: Database columns for PurchaseStatus_CODE table

Database Column	Details
PurchaseStatusID	*Type:* integer. Key A unique identifier for Purchase status. Possible values and the corresponding default strings are: • 1 = New • 2 = Pending • 3 = Completed • 4 = Cancelled
PurchaseStatus_en	*Type:* text (max 1000 characters) Purchase status of the license in English.
PurchaseStatus_de	*Type:* text (max 1000 characters) Purchase status of the license in German.
PurchaseStatus_fr	*Type:* text (max 1000 characters) Purchase status of the license in French.
PurchaseStatus_ja	*Type:* text (max 1000 characters) Purchase status of the license in Japanese.
PurchaseStatus_es	*Type:* text (max 1000 characters) Purchase status of the license in Spanish.

PurchaseType_CODE Table

PurchaseType_CODE is a dimension table storing purchase type information.

Table 661: Database columns for `PurchaseType_CODE` table

Database Column	Details
`PurchaseTypeID`	*Type:* integer. Key A unique identifier for purchase order detail type. Possible values and the corresponding default strings are: • 1 = NotSet • 2 = Software • 3 = Hardware • 4 = Service • 5 = Other • 6 = Software Upgrade • 7 = Software Maintenance • 8 = DiskKit • 9 = Hardware Maintenance • 10 = Software Baseline • 11 = Software Subscription
`PurchaseType_en`	*Type:* text (max 1000 characters) Purchase Type of the purchase in English.
`PurchaseType_de`	*Type:* text (max 1000 characters) Purchase Type of the purchase in German.
`PurchaseType_fr`	*Type:* text (max 1000 characters) Purchase Type of the purchase in French.
`PurchaseType_ja`	*Type:* text (max 1000 characters) Purchase Type of the purchase in Japanese.
`PurchaseType_es`	*Type:* text (max 1000 characters) Purchase Type of the purchase in Spanish.

Purchase_DIM Table

`Purchase_DIM` is a dimension table storing all purchase information.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 662: Database columns for `Purchase_DIM` table

Database Column	Details
`PurchaseOrderDetailID`	*Type:* integer. Key Foreign key of the purchaseorder.
`PurchaseNo`	*Type:* text (max 50 characters). Nullable Purchase No of the purchase.
`PurchaseDescription`	*Type:* text (max 250 characters) Purchase Description of the purchase.
`EffectiveQuantity`	*Type:* integer. Nullable Effective Quantity of the purchase.
`PurchaseDate`	*Type:* datetime Purchase date of the purchase.
`CreationDate`	*Type:* datetime Creation date of the purchase.
`PublisherName`	*Type:* text (max 64 characters). Nullable Publisher name of the purchase.
`VendorName`	*Type:* text (max 64 characters). Nullable Vendor name of the purchase.

Purchase_Latest_FACT Table

`Purchase_Latest_FACT` is a fact table storing latest purchase information per ComplianceOperator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 663: Database columns for `Purchase_Latest_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`PurchaseOrderDetailID`	*Type:* integer. Key Foreign key of the purchaseorder.

Database Column	Details
`PurchaseTypeID`	*Type:* integer Purchase type ID of the purchase.
`PurchaseStatusID`	*Type:* integer Purchase status ID of the purchase.
`CurrencyID`	*Type:* integer Currency ID (total price id) of the purchase.
`TotalPrice`	*Type:* currency. Nullable Total price of the purchase.

Purchase_Spend_FACT Table

`Purchase_Spend_FACT` is a fact table storing all purchase information per ComplianceOperator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 664: Database columns for `Purchase_Spend_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`PurchaseOrderDetailID`	*Type:* integer. Key Foreign key of the purchaseorder.
`PurchaseTypeID`	*Type:* integer Purchase type ID of the purchase.
`CurrencyID`	*Type:* integer Currency ID (total price id) of the purchase.
`TotalPrice`	*Type:* currency. Nullable Total price of the purchase.

Purchase_Unprocessed_FACT Table

`Purchase_Unprocessed_FACT` is a fact table storing unprocessed purchase information per ComplianceOperator.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 665: Database columns for `Purchase_Unprocessed_FACT` table

Database Column	Details
`ComplianceOperatorID`	*Type:* integer. Key Foreign key to the operator.
`PurchaseOrderDetailID`	*Type:* integer. Key Foreign key of the purchaseorder.
`AvailableEntitlements`	*Type:* integer Available Entitlements of the purchase.

ResourceString_CODE Table

`ResourceString_CODE` is a lookup table for localized text.

Table 666: Database columns for `ResourceString_CODE` table

Database Column	Details
`ResourceKey`	*Type:* text (max 256 characters). Key Primary key of the resource string.
`ResourceString_en`	*Type:* text (max 1000 characters) Resource string in English.
`ResourceString_de`	*Type:* text (max 1000 characters) Resource string in German.
`ResourceString_fr`	*Type:* text (max 1000 characters) Resource string in French.
`ResourceString_ja`	*Type:* text (max 1000 characters) Resource string in Japanese.
`ResourceString_es`	*Type:* text (max 1000 characters). Nullable Resource string in Spanish.

VMType_CODE Table

`VMType_CODE` is an enumerated code table for VM type.

Table 667: Database columns for `VMType_CODE` table

Database Column	Details
`VMTypeID`	*Type:* integer. Key A unique identifier for VM type. Possible values and the corresponding default strings are: • 1 = VMware • 2 = Hyper-V • 3 = LPAR • 4 = WPAR • 5 = nPar • 6 = vPar • 7 = SRP • 8 = Zone • 9 = Unknown. • 10 = Oracle VM
`VMType_en`	*Type:* text (max 1000 characters) VM type in English.
`VMType_de`	*Type:* text (max 1000 characters) VM type in German.
`VMType_fr`	*Type:* text (max 1000 characters) VM type in French.
`VMType_ja`	*Type:* text (max 1000 characters) VM type in Japanese.
`VMType_es`	*Type:* text (max 1000 characters). Nullable VM type in Spanish.

4

DataWarehouse Database Schema

This chapter describes a schema for the dimensional data model available for reporting using the Flexera Analytics (powered by Cognos).

There are three separate data models related to IBM Cognos within FlexNet Manager Suite:

- A model for use when customizing dashboards for FlexNet Manager Suite (see Dashboard Tables)
- An operational model for reporting on live data (this model is not separately documented)
- The dimensional data model for reporting on data that changes over time, which is the subject of this chapter.

DataWarehouseTables Tables

The complete set of database tables documented here includes:

- AssignmentData table (see AssignmentData Table)
- CategoryData table (see CategoryData Table)
- ConsumptionData table (see ConsumptionData Table)
- CorporateUnitData table (see CorporateUnitData Table)
- CostCenterData table (see CostCenterData Table)
- DataWarehouseSetting table (see DataWarehouseSetting Table)
- InstallationData table (see InstallationData Table)
- LocationData table (see LocationData Table)
- PurchaseData table (see PurchaseData Table)
- PurchaseDateData table (see PurchaseDateData Table)
- SnapshotData table (see SnapshotData Table)
- SoftwareLicenseData table (see SoftwareLicenseData Table)

- SoftwareTitleData table (see SoftwareTitleData Table)
- VendorData table (see VendorData Table)
- VendorPurchaseData table (see VendorPurchaseData Table)

AssignmentData Table

Stores all assignment information required by the external Consumption Fact.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 668: Database columns for `AssignmentData` table

Database Column	Details
`AssignmentDataID`	*Type:* integer. Generated ID A unique identifier for this assignment.
`SnapshotID`	*Type:* integer. Key The snapshot to which this assignment data pertains. Reference to the snapshot dimension.
`SoftwareLicenseID`	*Type:* integer. Key The license to which this assignment data pertains. Reference to the software license dimension.
`LocationID`	*Type:* integer. Key The location where the assignments were made.
`CorporateUnitID`	*Type:* integer. Key The corporate unit where the assignments were made.
`CostCenterID`	*Type:* integer. Key The cost center where the assignments were made.
`CategoryID`	*Type:* integer. Key The category which classifies this license assignment.
`AssignedCount`	*Type:* integer The number of licenses that have been assigned or the number of licenses that have been consumed as a result of group assignment.

CategoryData Table

This table stores each of the categories known to FNMP. This maps directly to the External Category Dimension.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 669: Database columns for `CategoryData` table

Database Column	Details
`CategoryID`	*Type:* integer. Key Unique identifier for this category from the FNMP database.
`GroupExID`	*Type:* text (max 128 characters) Internal identifier for this category.
`Level1`	*Type:* integer Parsed GroupExID, first level
`Level1Name`	*Type:* text (max 500 characters) Parsed Path, first level
`Level2`	*Type:* integer. Nullable Parsed GroupExID, second level
`Level2Name`	*Type:* text (max 500 characters). Nullable Parsed Path, second level
`Level3`	*Type:* integer. Nullable Parsed GroupExID, third level
`Level3Name`	*Type:* text (max 500 characters). Nullable Parsed Path, third level
`Level4`	*Type:* integer. Nullable Parsed GroupExID, fourth level
`Level4Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fourth level
`Level5`	*Type:* integer. Nullable Parsed GroupExID, fifth level
`Level5Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fifth level

Database Column	Details
`Level6`	*Type:* integer. Nullable Parsed GroupExID, sixth level
`Level6Name`	*Type:* text (max 500 characters). Nullable Parsed Path, sixth level
`Level7`	*Type:* integer. Nullable Parsed GroupExID, seventh level
`Level7Name`	*Type:* text (max 500 characters). Nullable Parsed Path, seventh level
`Level8`	*Type:* integer. Nullable Parsed GroupExID, eighth level
`Level8Name`	*Type:* text (max 500 characters). Nullable Parsed Path, eighth level
`Level9`	*Type:* integer. Nullable Parsed GroupExID, ninth level
`Level9Name`	*Type:* text (max 500 characters). Nullable Parsed Path, ninth level
`Level10`	*Type:* integer. Nullable Parsed GroupExID, tenth level
`Level10Name`	*Type:* text (max 500 characters). Nullable Parsed Path, tenth level
`CategoryPath`	*Type:* text (max 500 characters) The full path to this category.
`CategoryName`	*Type:* text (max 256 characters) The full name of this category.

ConsumptionData Table

Stores all consumption information required by the external Consumption Fact.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 670: Database columns for `ConsumptionData` table

Database Column	Details
`ConsumptionDataID`	*Type:* integer. Generated ID A unique identifier for this consumption data.
`SnapshotID`	*Type:* integer. Key The snapshot to which this consumption data pertains. Reference to the snapshot dimension.
`SoftwareLicenseID`	*Type:* integer. Key The license to which this consumption data pertains. Reference to the software license dimension.
`LocationID`	*Type:* integer. Key The location which has consumed this license.
`CorporateUnitID`	*Type:* integer. Key The corporate unit which has consumed this license.
`CostCenterID`	*Type:* integer. Key The cost center which has consumed this license.
`CategoryID`	*Type:* integer. Key The category which classifies this license consumption.
`InstalledCount`	*Type:* integer. Nullable Number of installed software records, linked to the license. It is not a number of application installations.
`ConsumedCount`	*Type:* integer The number of licenses consumed.
`UsedCount`	*Type:* integer The number of license consumptions that were used.
`SecondUseCount`	*Type:* integer The number of installations which are not consuming a license as a result of second use rights.
`DowngradeCount`	*Type:* integer The number of licenses consumed which are a result of downgrade rights.
`VirtualEnvironmentCount`	*Type:* integer The number of installations which are not consuming a license as a result of virtual machine product use rights.

Database Column	Details
`VMNonConsumedCount`	*Type:* integer. Nullable Count that is not consumed because of VM second use rights.
`ExemptCount`	*Type:* integer The number of installations which are exempt from consuming a license.
`LicensedCores`	*Type:* integer. Nullable The number of processor cores that are covered by a license.

CorporateUnitData Table

This table will store each of the corporate units known to FNMP. This will map directly to the External Corporate Unit Dimension.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 671: Database columns for `CorporateUnitData` table

Database Column	Details
`CorporateUnitID`	*Type:* integer. Key Unique identifier for this corporate unit from the FNMP database.
`GroupExID`	*Type:* text (max 128 characters) Internal identifier for this corporate unit.
`Level1`	*Type:* integer Parsed GroupExID, first level
`Level1Name`	*Type:* text (max 500 characters) Parsed Path, first level
`Level2`	*Type:* integer. Nullable Parsed GroupExID, second level
`Level2Name`	*Type:* text (max 500 characters). Nullable Parsed Path, second level
`Level3`	*Type:* integer. Nullable Parsed GroupExID, third level
`Level3Name`	*Type:* text (max 500 characters). Nullable Parsed Path, third level

Database Column	Details
`Level4`	*Type:* integer. Nullable Parsed GroupExID, fourth level
`Level4Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fourth level
`Level5`	*Type:* integer. Nullable Parsed GroupExID, fifth level
`Level5Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fifth level
`Level6`	*Type:* integer. Nullable Parsed GroupExID, sixth level
`Level6Name`	*Type:* text (max 500 characters). Nullable Parsed Path, sixth level
`Level7`	*Type:* integer. Nullable Parsed GroupExID, seventh level
`Level7Name`	*Type:* text (max 500 characters). Nullable Parsed Path, seventh level
`Level8`	*Type:* integer. Nullable Parsed GroupExID, eighth level
`Level8Name`	*Type:* text (max 500 characters). Nullable Parsed Path, eighth level
`Level9`	*Type:* integer. Nullable Parsed GroupExID, ninth level
`Level9Name`	*Type:* text (max 500 characters). Nullable Parsed Path, ninth level
`Level10`	*Type:* integer. Nullable Parsed GroupExID, tenth level
`Level10Name`	*Type:* text (max 500 characters). Nullable Parsed Path, tenth level
`CorporateUnitPath`	*Type:* text (max 500 characters) The full path to this corporate unit.
`CorporateUnitName`	*Type:* text (max 256 characters) The name of this corporate unit.

CostCenterData Table

This table stores each of the cost centers known to FNMP. This will map directly to the External Cost Center Dimension.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 672: Database columns for `CostCenterData` table

Database Column	Details
`CostCenterID`	*Type:* integer. Key Unique identifier fro this cost center from the FNMP database.
`GroupExID`	*Type:* text (max 128 characters) Internal identifier for this cost center.
`Level1`	*Type:* integer Parsed GroupExID, first level
`Level1Name`	*Type:* text (max 500 characters) Parsed Path, first level
`Level2`	*Type:* integer. Nullable Parsed GroupExID, second level
`Level2Name`	*Type:* text (max 500 characters). Nullable Parsed Path, second level
`Level3`	*Type:* integer. Nullable Parsed GroupExID, third level
`Level3Name`	*Type:* text (max 500 characters). Nullable Parsed Path, third level
`Level4`	*Type:* integer. Nullable Parsed GroupExID, fourth level
`Level4Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fourth level
`Level5`	*Type:* integer. Nullable Parsed GroupExID, fifth level
`Level5Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fifth level

Database Column	Details
`Level6`	*Type:* integer. Nullable Parsed GroupExID, sixth level
`Level6Name`	*Type:* text (max 500 characters). Nullable Parsed Path, sixth level
`Level7`	*Type:* integer. Nullable Parsed GroupExID, seventh level
`Level7Name`	*Type:* text (max 500 characters). Nullable Parsed Path, seventh level
`Level8`	*Type:* integer. Nullable Parsed GroupExID, eighth level
`Level8Name`	*Type:* text (max 500 characters). Nullable Parsed Path, eighth level
`Level9`	*Type:* integer. Nullable Parsed GroupExID, ninth level
`Level9Name`	*Type:* text (max 500 characters). Nullable Parsed Path, ninth level
`Level10`	*Type:* integer. Nullable Parsed GroupExID, tenth level
`Level10Name`	*Type:* text (max 500 characters). Nullable Parsed Path, tenth level
`CostCenterPath`	*Type:* text (max 500 characters) The full path to this cost center.
`CostCenterName`	*Type:* text (max 256 characters) The name of this cost center.

DataWarehouseSetting Table

Stores settings for data warehouse: currency, currency symbol

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 673: Database columns for `DataWarehouseSetting` table

Database Column	Details
`DataWarehouseSettingID`	*Type:* integer. Key. Generated ID A unique identifier for this assignment.
`ComplianceOperatorID`	*Type:* integer. Key. Nullable For future use
`CurrencyName`	*Type:* text (max 128 characters) Currency name
`CurrencySymbol`	*Type:* text (max 128 characters) Currency symbol

InstallationData Table

Stores all of the installation information. Installation records are scoped as per the scoping rules in FNMP.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 674: Database columns for `InstallationData` table

Database Column	Details
`InstallationDataID`	*Type:* integer. Key. Generated ID A unique identifier for this installation data.
`SnapshotID`	*Type:* integer. Key The snapshot to which this installation data pertains. Reference to the snapshot dimension.
`SoftwareTitleID`	*Type:* integer. Key The software title that is installed. Reference to the software title dimension.
`LocationID`	*Type:* integer. Key The location where these installs occurred.
`CorporateUnitID`	*Type:* integer. Key The corporate unit where these installs occurred.
`CostCenterID`	*Type:* integer. Key The cost center where these installs occurred.

Database Column	Details
`CategoryID`	*Type:* integer. Key The category that classifies these installs.
`InstalledCount`	*Type:* integer. Key The number of installs.
`LicensableInstalledCount`	*Type:* integer The number of licensable installs.
`LicensedCount`	*Type:* integer The number of installs which are covered by a license.
`UsedCount`	*Type:* integer. Key The number of installations which have usage exceeding the defined levels for the installation to be deemed used.
`VirtualEnvironmentCount`	*Type:* integer The number of installs which are on virtual machines.

LocationData Table

This table will store each of the locations known to FNMP. This will map directly to the External Location Dimension.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 675: Database columns for `LocationData` table

Database Column	Details
`LocationID`	*Type:* integer. Key Unique identifier for this location from the FNMP database.
`GroupExID`	*Type:* text (max 128 characters) Internal identifier for this location.
`Level1`	*Type:* integer Parsed GroupExID, first level
`Level1Name`	*Type:* text (max 500 characters) Parsed Path, first level
`Level2`	*Type:* integer. Nullable Parsed GroupExID, second level

Database Column	Details
`Level2Name`	*Type:* text (max 500 characters). Nullable Parsed Path, second level
`Level3`	*Type:* integer. Nullable Parsed GroupExID, third level
`Level3Name`	*Type:* text (max 500 characters). Nullable Parsed Path, third level
`Level4`	*Type:* integer. Nullable Parsed GroupExID, fourth level
`Level4Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fourth level
`Level5`	*Type:* integer. Nullable Parsed GroupExID, fifth level
`Level5Name`	*Type:* text (max 500 characters). Nullable Parsed Path, fifth level
`Level6`	*Type:* integer. Nullable Parsed GroupExID, sixth level
`Level6Name`	*Type:* text (max 500 characters). Nullable Parsed Path, sixth level
`Level7`	*Type:* integer. Nullable Parsed GroupExID, seventh level
`Level7Name`	*Type:* text (max 500 characters). Nullable Parsed Path, seventh level
`Level8`	*Type:* integer. Nullable Parsed GroupExID, eighth level
`Level8Name`	*Type:* text (max 500 characters). Nullable Parsed Path, eighth level
`Level9`	*Type:* integer. Nullable Parsed GroupExID, ninth level
`Level9Name`	*Type:* text (max 500 characters). Nullable Parsed Path, ninth level
`Level10`	*Type:* integer. Nullable Parsed GroupExID, tenth level

Database Column	Details
`Level10Name`	*Type:* text (max 500 characters). Nullable Parsed Path, tenth level
`LocationPath`	*Type:* text (max 500 characters) The full path of this location.
`LocationName`	*Type:* text (max 256 characters) The name of this location.

PurchaseData Table

The Purchases table will store all purchase information that is required for the External Consumption Fact.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 676: Database columns for `PurchaseData` table

Database Column	Details
`PurchasesDataID`	*Type:* integer. Generated ID A unique identifier for this purchase information.
`SnapshotID`	*Type:* integer. Key The snapshot to which this purchase data pertains. Reference to the snapshot dimension.
`SoftwareLicenseID`	*Type:* integer. Key The license to which these purchases pertain. Reference to software license dimension.
`LocationID`	*Type:* integer. Key The location which has made these purchases.
`CorporateUnitID`	*Type:* integer. Key The corporate unit which has made these purchases.
`CostCenterID`	*Type:* integer. Key The cost center which has made these purchases.
`CategoryID`	*Type:* integer. Key The category which classifies this license purchases.

Database Column	Details
`PurchasedCount`	*Type:* integer. Nullable The number of license entitlements purchased.
`PurchasedCost`	*Type:* currency. Nullable The purchase cost for these license entitlements.
`LastPurchaseDate`	*Type:* datetime. Nullable The last date on which a purchase of entitlements for this license was made.

PurchaseDateData Table

This table stores purchase dates known to FNMP. This maps directly to the External Purchase Date Dimension. This is not a slowly changing dimension, but a snapshot of current and deleted purchase order dates.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 677: Database columns for `PurchaseDateData` table

Database Column	Details
`PurchaseDateID`	*Type:* integer. Key. Generated ID Unique identifier for this purchase date.
`Month`	*Type:* integer. Key The month for this purchase date.
`Year`	*Type:* integer. Key The year for this purchase date.

SnapshotData Table

Stores information to uniquely identify each individual snapshot. Whilst the scheduled snapshot generation will only happen on a daily or weekly basis, there is no restriction that multiple snapshots on the same day could not be generated.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 678: Database columns for `SnapshotData` table

Database Column	Details
`SnapshotID`	*Type:* integer. Key. Generated ID A unique identifier for a snapshot.
`SnapshotYear`	*Type:* integer. Key The year in which the snapshot was created.
`SnapshotMonth`	*Type:* integer. Key The month in which the snapshot was created.
`SnapshotDay`	*Type:* integer. Key The day on which the snapshot was created.
`SnapshotDate`	*Type:* datetime The date and time the snapshot was created or last updated.
`SnapshotDescription`	*Type:* text (max 500 characters) A description of this snapshot.

SoftwareLicenseData Table

This table stores each of the Software Licenses known to FNMP. This maps directly to the External Software License Dimension. This is not a slowly changing dimension, but a snapshot of current and deleted license information. If license properties change from one snapshot to the next, the information in this table will be updated.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 679: Database columns for `SoftwareLicenseData` table

Database Column	Details
`SoftwareLicenseID`	*Type:* integer. Key Unique identifier for this license from the FNMP database.
`ProductName`	*Type:* text (max 1024 characters) Product name of the primary application of this license.
`PublisherName`	*Type:* text (max 64 characters). Nullable The name of the publisher
`LicenseName`	*Type:* text (max 256 characters) The name of this license.

Database Column	Details
`LicenseVersion`	*Type:* text (max 60 characters). Nullable The version of this license.
`LicenseEdition`	*Type:* text (max 60 characters). Nullable The edition of this license.
`LicenseTypeID`	*Type:* integer ID of the type of this license.
`LicenseType`	*Type:* text (max 256 characters) The type of this license.
`GrantsSecondUse`	*Type:* boolean Whether this license offers second use rights.
`GrantsDowngrade`	*Type:* boolean Whether this license offers downgrade rights.
`IsTrueUp`	*Type:* boolean Whether this license provides True Up functionality.
`UnlimitedConsumption`	*Type:* boolean Whether this license provides unlimited consumption.
`EstimatedUnitPrice`	*Type:* currency. Nullable Estimated Unit price for the license
`GrantsVirtualEnvironment`	*Type:* boolean Whether installs of this license on a virtual machine host covers installations on virtual machines hosted by that host.
`UseInSecondUseRights`	*Type:* boolean A Boolean field that states whether product use rights apply to this license type.
`NumberPurchased`	*Type:* integer The quantity of purchased license entitlements.
`LocationID`	*Type:* integer. Key The location which owns this license.
`CorporateUnitID`	*Type:* integer. Key The corporate unit which owns this license.
`CostCenterID`	*Type:* integer. Key The cost center which owns this license.

Database Column	Details
`CategoryID`	*Type:* integer. Key The category of this license.
`LicenseStatusID`	*Type:* integer ID of the status of this license
`LicenseStatus`	*Type:* text (max 256 characters) License Status of the license
`ComplianceStatusID`	*Type:* integer ID of the compliance status of this license
`ComplianceStatus`	*Type:* text (max 256 characters) Compliance Status of the license
`DurationID`	*Type:* integer ID of duration of this license
`Duration`	*Type:* text (max 256 characters) The name of the resource string containing the text to display on the user interface.
`ExpiryDate`	*Type:* datetime. Nullable The date this license expires. A NULL value means the license does not expire.

SoftwareTitleData Table

This table stores each of the software titles known to FNMP. This maps directly to the External Software Title Dimension. This is not a slowly changing dimension, but a snapshot of current and deleted software titles. If title properties change from one snapshot to the next, the information in this table will be updated.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 680: Database columns for `SoftwareTitleData` table

Database Column	Details
`SoftwareTitleID`	*Type:* integer. Key Unique identifier for this software title from the FNMP database.
`SoftwareTitleName`	*Type:* text (max 512 characters) The name of this software title.

Database Column	Details
`PublisherName`	*Type:* text (max 200 characters) The publisher of this software title.
`ProductName`	*Type:* text (max 1024 characters) The product represented by this software title.
`VersionName`	*Type:* text (max 50 characters) The version of this software title.
`VersionWeight`	*Type:* decimal. Nullable A numeric value used to sort various versions of a software title.
`EditionName`	*Type:* text (max 50 characters) The edition of this software title.
`EditionWeight`	*Type:* decimal. Nullable A numeric value used to sort various editions of a software title.
`Classification`	*Type:* text (max 50 characters) The classification of this software title.
`ClassificationID`	*Type:* integer. Nullable The ID of the classification of this software title.
`Action`	*Type:* text (max 50 characters) The action of this software title.
`ActionID`	*Type:* integer The ID of the action of this software title.
`IsLicensed`	*Type:* boolean 1 if the SoftwareTitle is linked to any license
`OperatorManageStateID`	*Type:* integer The management responsibility for this software title. Part of the unique key for a software title in the FNMP database.

VendorData Table

This table stores each of the vendors known to FNMP. This maps directly to the External Vendor Dimension. This is not a slowly changing dimension, but a snapshot of current and deleted vendors. If vendor properties change from one snapshot to the next, the information in this table will be updated.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 681: Database columns for `VendorData` table

Database Column	Details
`VendorID`	*Type:* integer. Key Unique identifier for this vendor from the FNMP database.
`VendorName`	*Type:* text (max 64 characters) The name of this vendor.

VendorPurchaseData Table

Stores all of the vendor purchase information. Purchase records are scoped as per the scoping rules in FNMP.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 682: Database columns for `VendorPurchaseData` table

Database Column	Details
`VendorPurchaseID`	*Type:* integer. Generated ID Unique identifier for this vendor purchase.
`VendorID`	*Type:* integer. Key The vendor to which this purchase data pertains. Reference to the vendor dimension.
`PurchaseDateID`	*Type:* integer. Key The date to which this vendor purchase data pertains. Reference to the purchase date dimension.
`LocationID`	*Type:* integer. Key The location where these purchases occurred.
`CorporateUnitID`	*Type:* integer. Key The corporate unit where these purchases occurred.
`CostCenterID`	*Type:* integer. Key The cost center where these purchases occurred.
`CategoryID`	*Type:* integer. Key The category that classifies these purchases.

Database Column	Details
`HardwareCost`	*Type:* currency The cost of hardware purchased from this vendor on this date.
`HardwareMaintenanceCost`	*Type:* currency The cost of hardware maintenance purchased from this vendor on this date.
`SoftwareCost`	*Type:* currency The cost of software purchased from this vendor on this date.
`SoftwareMaintenanceCost`	*Type:* currency The cost of software maintenance purchased from this vendor on this date.
`OtherCost`	*Type:* currency The cost of other items purchased from this vendor on this date.

5

Compliance Reader Database Schema

This chapter describes the schema for the staging tables used by the importer (`ComplianceReader.exe`) in the process of importing data into the main FlexNet Manager Suite. Imports through these tables may come from many sources, including (but not limited to) the inventory data collected by the FlexNet inventory agent and rationalized in the inventory database (see Inventory Database Schema).

For each data source, data in these tables is over-written as each import.

Information Structure

The following information is provided about database tables. Items appear only when relevant to the database column, and are suppressed where they do not apply. Two of these items (shown bold) are columns in the following pages, and the remainder are displayed within the **Details**.

Item	Comment
Database Column	The name of the column in the SQL table.
Type	The data type of the contents of the database column.
Size	For types that have a maximum capacity, the upper limit is provided in parentheses.
Key	The word “Key” appears when a column is a unique key field within the table. It is possible for several database columns to be part of the key, so that this indicator may appear for several columns in a table.
Generated ID	This indicates that a numeric ID is assigned by the database.
Nullable	If this indicator is present, the database column permits nulls.
Computed	This indicator appears for columns that are automatically computed by the database.
Default	If a column has a default value declared in the schema, this is specified at the end of the first set of details for the column.

Item	Comment
Details	Describes the data stored in the database column, including many of the indicators described above.

Compliance.InventoryReader.Logic Tables

The complete set of database tables documented here includes:

- ExpiredImportedComputer table (see ExpiredImportedComputer Table)
- ImportedARSLicense table (see ImportedARSLicense Table)
- ImportedAccessingDevice table (see ImportedAccessingDevice Table)
- ImportedAccessingUser table (see ImportedAccessingUser Table)
- ImportedActiveDirectoryComputer table (see ImportedActiveDirectoryComputer Table)
- ImportedActiveDirectoryDomain table (see ImportedActiveDirectoryDomain Table)
- ImportedActiveDirectoryExternalMember table (see ImportedActiveDirectoryExternalMember Table)
- ImportedActiveDirectoryGroup table (see ImportedActiveDirectoryGroup Table)
- ImportedActiveDirectoryMember table (see ImportedActiveDirectoryMember Table)
- ImportedActiveDirectoryUser table (see ImportedActiveDirectoryUser Table)
- ImportedActiveSyncDevice table (see ImportedActiveSyncDevice Table)
- ImportedAttributeMapping table (see ImportedAttributeMapping Table)
- ImportedClientAccessEvidence table (see ImportedClientAccessEvidence Table)
- ImportedClientAccessEvidenceMapping table (see ImportedClientAccessEvidenceMapping Table)
- ImportedClientAccessedAccessEvidence table (see ImportedClientAccessedAccessEvidence Table)
- ImportedClientAccessedAccessOccurrence table (see ImportedClientAccessedAccessOccurrence Table)
- ImportedCluster table (see ImportedCluster Table)
- ImportedClusterGroup table (see ImportedClusterGroup Table)
- ImportedClusterGroupMember table (see ImportedClusterGroupMember Table)
- ImportedClusterHostAffinityRule table (see ImportedClusterHostAffinityRule Table)
- ImportedClusterNode table (see ImportedClusterNode Table)
- ImportedComputer table (see ImportedComputer Table)
- ImportedComputerCustomProperty table (see ImportedComputerCustomProperty Table)
- ImportedComputerScriptResult table (see ImportedComputerScriptResult Table)

- ImportedCustomPropertyName table (see ImportedCustomPropertyName Table)
- ImportedDomain table (see ImportedDomain Table)
- ImportedEvidenceAttribute table (see ImportedEvidenceAttribute Table)
- ImportedFNMEAFeature table (see ImportedFNMEAFeature Table)
- ImportedFNMEAProduct table (see ImportedFNMEAProduct Table)
- ImportedFNMEAUsageStatus table (see ImportedFNMEAUsageStatus Table)
- ImportedFileEvidence table (see ImportedFileEvidence Table)
- ImportedFileEvidenceMapping table (see ImportedFileEvidenceMapping Table)
- ImportedGuidMapping table (see ImportedGuidMapping Table)
- ImportedILMTPVUCounts table (see ImportedILMTPVUCounts Table)
- ImportedILMTPVUCreatedLicenses table (see ImportedILMTPVUCreatedLicenses Table)
- ImportedILMTVMMapping table (see ImportedILMTVMMapping Table)
- ImportedInstalledFileEvidence table (see ImportedInstalledFileEvidence Table)
- ImportedInstalledFileEvidencePath table (see ImportedInstalledFileEvidencePath Table)
- ImportedInstalledFileEvidenceUsage table (see ImportedInstalledFileEvidenceUsage Table)
- ImportedInstalledInstallerEvidence table (see ImportedInstalledInstallerEvidence Table)
- ImportedInstalledInstallerEvidenceAttribute table (see ImportedInstalledInstallerEvidenceAttribute Table)
- ImportedInstalledInstallerEvidenceUsage table (see ImportedInstalledInstallerEvidenceUsage Table)
- ImportedInstalledWMIEvidence table (see ImportedInstalledWMIEvidence Table)
- ImportedInstallerEvidence table (see ImportedInstallerEvidence Table)
- ImportedInstallerEvidenceMapping table (see ImportedInstallerEvidenceMapping Table)
- ImportedInstallerEvidenceRepackageMapping table (see ImportedInstallerEvidenceRepackageMapping Table)
- ImportedInstance table (see ImportedInstance Table)
- ImportedInstanceUser table (see ImportedInstanceUser Table)
- ImportedMissingComputer table (see ImportedMissingComputer Table)
- ImportedMissingLicenseUser table (see ImportedMissingLicenseUser Table)
- ImportedMissingUser table (see ImportedMissingUser Table)
- ImportedPVUVirtualMachineLayer table (see ImportedPVUVirtualMachineLayer Table)
- ImportedProductCodeEvidenceMapping table (see ImportedProductCodeEvidenceMapping Table)
- ImportedRelatedInstalledInstallerEvidence table (see ImportedRelatedInstalledInstallerEvidence Table)

- ImportedRemoteApplication table (see ImportedRemoteApplication Table)
- ImportedRemoteApplicationAccess table (see ImportedRemoteApplicationAccess Table)
- ImportedRemoteApplicationInstallerData table (see ImportedRemoteApplicationInstallerData Table)
- ImportedRemoteApplicationServer table (see ImportedRemoteApplicationServer Table)
- ImportedRemoteServerFileEvidenceMapping table (see ImportedRemoteServerFileEvidenceMapping Table)
- ImportedRemoteUsage table (see ImportedRemoteUsage Table)
- ImportedRemoteUserToApplicationAccess table (see ImportedRemoteUserToApplicationAccess Table)
- ImportedSite table (see ImportedSite Table)
- ImportedSiteSubnet table (see ImportedSiteSubnet Table)
- ImportedSoftwareLicense table (see ImportedSoftwareLicense Table)
- ImportedSoftwareLicenseAllocation table (see ImportedSoftwareLicenseAllocation Table)
- ImportedSoftwareTitle table (see ImportedSoftwareTitle Table)
- ImportedSoftwareTitleAccessEvidence table (see ImportedSoftwareTitleAccessEvidence Table)
- ImportedSoftwareTitleLicense table (see ImportedSoftwareTitleLicense Table)
- ImportedStringMapping table (see ImportedStringMapping Table)
- ImportedStringMappingLatin1CS table (see ImportedStringMappingLatin1CS Table)
- ImportedTDSComputer table (see ImportedTDSComputer Table)
- ImportedUser table (see ImportedUser Table)
- ImportedVDI table (see ImportedVDI Table)
- ImportedVDIEndPointAccess table (see ImportedVDIEndPointAccess Table)
- ImportedVDITemplate table (see ImportedVDITemplate Table)
- ImportedVDIUser table (see ImportedVDIUser Table)
- ImportedVMHostDatastore table (see ImportedVMHostDatastore Table)
- ImportedVMHostManagedBySoftware table (see ImportedVMHostManagedBySoftware Table)
- ImportedVMHostProperty table (see ImportedVMHostProperty Table)
- ImportedVMPool table (see ImportedVMPool Table)
- ImportedVirtualMachine table (see ImportedVirtualMachine Table)
- ImportedWMIEvidence table (see ImportedWMIEvidence Table)
- ImportedWMIEvidenceRuleMapping table (see ImportedWMIEvidenceRuleMapping Table)
- ImporterValueMapping table (see ImporterValueMapping Table)

- InstalledApplications table (see InstalledApplications Table)
- RelatedInstalledApplications table (see RelatedInstalledApplications Table)

ExpiredImportedComputer Table

The `ExpiredImportedComputer` table holds all of the computers which have been retrieved from the source connections and are expired.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 683: Database columns for `ExpiredImportedComputer` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer.
`ComputerName`	*Type:* text (max 256 characters). Nullable The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by `GetComputerName()`. For UNIX, it is the host name of the machine, as returned by `gethostname(2)`.
`Domain`	*Type:* text (max 100 characters). Nullable The domain of the computer.
`OperatingSystem`	*Type:* text (max 128 characters). Nullable The operating system of the computer.
`ServicePack`	*Type:* text (max 128 characters). Nullable The service pack installed for the operating system.
`NumberOfProcessors`	*Type:* integer. Nullable The number of processors in the computer.
`ProcessorType`	*Type:* text (max 256 characters). Nullable The type of processor in the computer.
`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed of the fastest processor in the computer.

Database Column	Details
`NumberOfCores`	*Type:* integer. Nullable The number of cores in the computer.
`TotalMemory`	*Type:* big integer. Nullable The total RAM in the computer, in bytes.
`ChassisType`	*Type:* text (max 128 characters). Nullable The type of case of the computer. The value must be a (case insensitive) exact match for one of the values shown. Note that some license types use this information to optimize the licensing position, particularly with desktop and laptop computers.
`NumberOfHardDrives`	*Type:* integer. Nullable The number of hard drives in the computer.
`TotalDiskSpace`	*Type:* big integer. Nullable The total size of all hard drives in the computer.
`NumberOfNetworkCards`	*Type:* integer. Nullable The number of network cards in the computer.
`NumberOfDisplayAdapters`	*Type:* integer. Nullable The number of graphics cards in the computer.
`IPAddress`	*Type:* text (max 256 characters). Nullable The IP address of the computer.
`MACAddress`	*Type:* text (max 256 characters). Nullable The MAC address of the computer.

Database Column	Details
`Manufacturer`	*Type:* text (max 128 characters). Nullable The manufacturer of the computer hardware. Some examples include: • On Windows, the SMBios manufacturer (the WMI Manufacturer property of the 'Win32_ComputerSystem' class). • On Linux, 'Manufacturer' in the 'System Information' section resulting from the 'dmidecode' command. Sample command: 'dmidecode -s system-manufacturer' • On Solaris x86, as for Linux, with failovers first to 'sysinfo SI_HW_PROVIDER' and then to 'ModelNo'. • On Solaris SPARC, the 'sysinfo SI_HW_PROVIDER'. Typically this value is 'Sun_Microsystems' or, more recently, 'Oracle Corporation'. Failover to the 'ModelNo'. • On HP-UX, the string literal 'HP'. • On AIX, the 'modelname' system attribute preceding the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use 'IBM'. This value is typically 'IBM'.
`ModelNo`	*Type:* text (max 128 characters). Nullable The model number of the computer.

Database Column	Details
`SerialNo`	*Type:* text (max 100 characters). Nullable The hardware serial number of the computer. The goal of this value is to be tied to the physical hardware, partition or virtual machine and to be as unique as possible across all computers in the organization. This is due to its use in tracking computers, particularly after an operating system rebuild. This value is also used to socialize computer inventory from different inventory sources, and is used to map virtual machine guest operating system inventory to the VM host on which the virtual machine is running. Example sources: • On Windows, the SMBios serial number. The WMI 'SerialNumber' property of the 'Win32_BIOS' class. Can fail over to the 'SerialNumber' property of the 'Win32_SystemEnclosure' class which is typically the same value. • On Linux, the SMBios serial number read using the command 'dmidecode -s system-serial-number'. Specifically, the 'System Information' section and the 'Serial Number' in that section is used. • On Solaris 10 8/07 or later, for a non-global zone, the UUID value from the /etc/zones/index file. For a global zone, the same as Solaris 10 releases earlier than 8/07. • For Solaris 10 releases earlier than 8/07, the hexadecimal version of 'SI_HW_SERIAL' with an appended hyphen character followed by the Zone's name. For example, '838bfc7b-global' or '838bfc7b-myzone'. • For Solaris 8 and 9, The hexadecimal version of 'SI_HW_SERIAL'. • For Mac OS X, the serial number of the machine as printed on the packaging and found in "About this Mac" from the desktop. • For HP-UX, the 'confstr _CS_PARTITION_IDENT' partition identifier if it is an nPar or vPar, or '_CS_MACHINE_IDENT' if not; with a failover to the machine serial number, and a final failover to the 'uname' machine identification number. • For AIX, the 'id_to_partition' system attribute, starting from the third character (strips a '0X' from the start). For example, if the 'id_to_partition' system attribute is '0X0473409002F7B201' then use '0473409002F7B201'.
`HostID`	*Type:* text (max 100 characters). Nullable An identifier for the host of the computer (when the computer is a virtual machine).
`LastLoggedOnUser`	*Type:* text (max 128 characters). Nullable The DOMAIN/SAMAccountName of the user last logged onto the computer.
`InventoryDate`	*Type:* datetime. Nullable The date the computer last had inventory reported.

Database Column	Details
`HardwareInventoryDate`	*Type:* datetime. Nullable The date (and optionally time) when the hardware was last inventoried. For automated/scheduled data uploads through an inventory beacon, make sure that inventory dates are kept current, as they are used to report out-of-date inventory sources. For a one-time upload to the central application server, leave inventory dates empty (null). At each import from the saved file, the import date is used as the inventory date, which prevents the inventory becoming stale.
`ServicesInventoryDate`	*Type:* datetime. Nullable The date when services (for example, Oracle) were last scanned on this computer. For automated/scheduled data uploads through an inventory beacon, make sure that inventory dates are kept current, as they are used to report out-of-date inventory sources. For a one-time upload to the central application server, leave inventory dates empty (null). At each import from the saved file, the import date is used as the inventory date, which prevents the inventory becoming stale.
`InventoryAgent`	*Type:* text (max 128 characters) The name of the person or tool that performed the last inventory.
`ComplianceComputerID`	*Type:* integer. Nullable Identifier of the computer in the `ComplianceComputer` table that this imported computer links to. This is populated by the import process and does not need to be provided by the source connections.
`ComplianceDomainID`	*Type:* integer. Nullable Identifier of the domain in the `ComplianceDomain` table that this computer belongs to. This is populated by the import process and does not need to be provided by the source connections.
`IncompleteRecord`	*Type:* boolean. Nullable Used to identify records which do not have all information specified. Primarily used for `ManageSoft` source connections where the domain name was not reliably reported.
`NumberOfSockets`	*Type:* integer. Nullable The number of sockets in the computer.
`PartialNumberOfProcessors`	*Type:* decimal. Nullable The fractional processor count available to this computer.
`UntrustedSerialNo`	*Type:* boolean Use when this computer is known to have a serial number from a data source that should not be trusted.

Database Column	Details
`FullDetailsFromExternalID`	*Type:* big integer. Nullable If this computer is marked as incomplete, and some of its properties are updated from another computer, record the external ID if the full computer.
`FullDetailsFrom ComplianceConnectionID`	*Type:* integer. Nullable If this computer is marked as incomplete, and some of its properties are updated from another computer, record the connection ID if the full computer.
`ComplianceComputerTypeID`	*Type:* integer. Nullable If you know that the computer is a virtual machine or VM host, record that data here. If you are unsure, leave this cell empty (NULL): this allows the system to infer the computer type (for example, a computer with VMs linked to it is inferred to be a VM host). If data comes from multiple inventory sources, leaving this value as null also allows the value to be inserted from another source. So, unless there is a very good reason, do not just specify 'Computer', but allow the inference rules to help.
`ILMTAgentID`	*Type:* big integer. Nullable Store the unique ID used by the ILMT agent on this device, if the inventory source is aware of this value.
`FNMPComputerUID`	*Type:* unique identifier. Nullable The unique identifier generated for the computer from the IM database. This property should only be populated by the ManageSoft inventory adapter.
`HostIdentifyingNumber`	*Type:* text (max 128 characters). Nullable Virtual hosts may have an identifier that is unique only across that hardware model. It is less unique than the true hardware serial number, for example.
`HostType`	*Type:* text (max 128 characters). Nullable The type of the physical host computer. This value is similar to the model number, but it is always for the physical server that an execution context may be running on. Therefore, this will generally be a known value for standalone machines and partitions, but it will not be known for virtual machines. This value is used for matching computers. Examples: • 'i86pc' • 'Sun-Fire-T1000' • 'rx7620' • '785' (for a 9000/785/C3700) • '8202' (for an IBM,8202-E4B).
`NumberOfLogicalProcessors`	*Type:* integer. Nullable The number of logical processors in the computer.

Database Column	Details
IsRemoteACLDevice	*Type:* boolean Used to determine if the current record is a remote ACL based device.
IsDuplicate	*Type:* boolean Used to identify that imported computer is a duplicate of another, whereby a new computer will not created.
LegacySerialNo	*Type:* text (max 100 characters). Nullable A previous serial number of this computer that can also be used for matching.
UUID	*Type:* unique identifier. Nullable The BIOS UUID of the computer.
IMEI	*Type:* text (max 256 characters). Nullable IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types.
PhoneNumber	*Type:* text (max 128 characters). Nullable The phone number of the device. Used for mobile devices.
EmailAddress	*Type:* text (max 256 characters). Nullable The email address associated with the device. Typically used for mobile devices.
CalculatedUser	*Type:* text (max 128 characters). Nullable The domain/SAMAccountName of the calculated user. Some inventory systems calculate the user who owns a computer. For example, it might be the user who, over the last ten logins, logged in most often.
LastSuccessful InventoryDate	*Type:* datetime. Nullable For incremental imports, this represents the inventory date of the computer in the source at the time this record was last successfully imported. If the import procedure has failed, this may be different to the inventory date. At the end of a successful incremental import, this value is updated to match the inventory date. If no value is present in this field, either there has not been a successful import of this computer or the reader for this record is not using an incremental update model.
MDScheduleGeneratedDate	*Type:* datetime. Nullable The last time the managed device schedule was regenerated.
MDScheduleContainsPVUScan	*Type:* boolean. Nullable Does this managed device include an event in its current schedule for running extra IBM PVU hardware scans.
FirmwareSerialNumber	*Type:* text (max 100 characters). Nullable Serial number in the system firmware such as BIOS, EEPROM etc.

Database Column	Details
MachineID	*Type:* text (max 100 characters). Nullable For AIX, it is the System ID. For HP-UX, it is the Machine/Software ID. It is unset for other platforms.
IgnoredDueToLicense	*Type:* boolean True if this machine is not imported into compliance computer table due to license limitation
CloudServiceProvider	*Type:* text (max 256 characters). Nullable The cloud service provider for the computer record.
CSPMetadataJsonBlob	*Type:* text (max 4000 characters). Nullable Contains the json blob for an Imported Computer record
AgentVersion	*Type:* text (max 32 characters). Nullable The version of the agent which generated the inventory package.

ImportedARSLicense Table

The ImportedARSLicense table stores Action Request System BMC licenses.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 684: Database columns for ImportedARSLicense table

Database Column	Details
ComplianceConnectionID	*Type:* integer. Key. Nullable The identifier of a data source connection in the ComplianceConnection table.
ComputerID	*Type:* big integer. Key The identifier used in the source connection to represent the computer.
SoftwareLicenseID	*Type:* integer. Nullable The identifier for the license in the SoftwareLicense table.
ARSLicenseID	*Type:* integer The identifier for the imported ARS license.
ComplianceComputerID	*Type:* integer. Nullable The identifier for the compliance computer in the ComplianceComputer table.

Database Column	Details
`LicenseType`	*Type:* text (max 128 characters). Key The ARS license name.
`ECMLicenseName`	*Type:* text (max 256 characters) The name of the license in the FlexNet Manager Suite.
`LicenseKey`	*Type:* text (max 32 characters). Key. Nullable The imported license key.
`LicenseSubType`	*Type:* text (max 16 characters). Key The license subtype (FlexNet Manager Suite license version).
`IssueDate`	*Type:* datetime. Key The identifier for the issue date.
`ExpiryDate`	*Type:* datetime. Key. Nullable The identifier for the expiry date.
`SiteName`	*Type:* text (max 64 characters) The identifier for the site name.
`HostID`	*Type:* text (max 64 characters) An identifier for the ARS host in the source connection (not used in FlexNet Manager Suite).
`LicenseNum`	*Type:* integer The purchase count for the ARS license.
`TokenList`	*Type:* text (max 128 characters). Nullable The ARS token list (not used in FlexNet Manager Suite).
`Comment`	*Type:* text. Nullable Extra information about the ARS license.
`Deleted`	*Type:* integer Set this flag if an ARS license is to be deleted.

ImportedAccessingDevice Table

The `ImportedAccessingDevice` table holds a record client access device information.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 685: Database columns for `ImportedAccessingDevice` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalAccessingDeviceID`	*Type:* big integer. Key The identifier used to identify the device in source connection
`AccessingDeviceID`	*Type:* integer. Key. Nullable Matching accessing device ID. Foreign key to the `AccessingDevice` table.
`IPAddress`	*Type:* text (max 256 characters). Key. Nullable IP Address of the client accessing device.
`ComputerName`	*Type:* text (max 256 characters). Key. Nullable Computer name of the client accessing device.
`SerialNo`	*Type:* text (max 100 characters). Nullable Serial no of the client accessing device.
`Domain`	*Type:* text (max 100 characters). Key. Nullable Domain name of the client accessing device.

ImportedAccessingUser Table

The `ImportedAccessingUser` table holds a record of the user access infomarion.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 686: Database columns for `ImportedAccessingUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalAccessingUserID`	*Type:* big integer. Key The accessing user id. This is part of the key.
`AccessingUserID`	*Type:* integer. Key. Nullable The matching AccessingUser ID. Foreign key to the `AccessingUser` table.

Database Column	Details
`UserName`	*Type:* text (max 256 characters). Key User name of the accessing user.
`DomainName`	*Type:* text (max 100 characters). Key. Nullable Domain name of the accessing user.
`SAMAccountName`	*Type:* text (max 64 characters). Nullable SAM account name of the accessing user.

ImportedActiveDirectoryComputer Table

The `ImportedActiveDirectoryComputer` table stores the incoming active directory data for computers.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 687: Database columns for `ImportedActiveDirectoryComputer` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`GUID`	*Type:* unique identifier. Key The GUID of the computer.
`ComputerName`	*Type:* text (max 64 characters) The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by `GetComputerName()`. For UNIX, it is the host name of the machine, as returned by `gethostname(2)`.
`DomainName`	*Type:* text (max 100 characters) The domain name for the computer.
`SID`	*Type:* text (max 256 characters). Nullable The SID of the computer.

ImportedActiveDirectoryDomain Table

The `ImportedActiveDirectoryDomain` table stores the incoming active directory domains for a connection source.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 688: Database columns for `ImportedActiveDirectoryDomain` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`DomainFQDN`	*Type:* text (max 100 characters). Key The fully qualified name domain name of the AD domain
`FlatName`	*Type:* text (max 32 characters) The AD domain flat name
`LastADImportTime`	*Type:* datetime The last time the AD data was imported

ImportedActiveDirectoryExternalMember Table

The `ImportedActiveDirectoryExternalMember` table stores the incoming active directory data for external AD member objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 689: Database columns for `ImportedActiveDirectoryExternalMember` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`ParentGroupGUID`	*Type:* unique identifier. Key The parent AD group GUID.
`SID`	*Type:* text (max 256 characters). Key The SID of the member object.

ImportedActiveDirectoryGroup Table

The `ImportedActiveDirectoryGroup` table stores the incoming active directory data for a connection source.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 690: Database columns for `ImportedActiveDirectoryGroup` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`GUID`	*Type:* unique identifier. Key The GUID of the AD group.
`SID`	*Type:* text (max 256 characters). Nullable The SID of the AD group.
`Name`	*Type:* text (max 128 characters). Nullable The AD group name
`DomainName`	*Type:* text (max 100 characters) The domain name for the user.

ImportedActiveDirectoryMember Table

The `ImportedActiveDirectoryMember` table stores the incoming active directory data for AD member objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 691: Database columns for `ImportedActiveDirectoryMember` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`GUID`	*Type:* unique identifier. Key The GUID of the member object.
`ParentGroupGUID`	*Type:* unique identifier. Key The parent AD group GUID.

ImportedActiveDirectoryUser Table

The `ImportedActiveDirectoryUser` table stores the incoming active directory data for users.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 692: Database columns for `ImportedActiveDirectoryUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`GUID`	*Type:* unique identifier. Key The GUID of the user.
`SAMAccountName`	*Type:* text (max 20 characters) The user name.
`DomainName`	*Type:* text (max 100 characters) The domain name for the user.
`Sid`	*Type:* text (max 256 characters). Nullable The Sid for the user.

ImportedActiveSyncDevice Table

The `ImportedActiveSyncDevice` table stores details of ActiveSync partnerships. A partnership is a user/device pair, so there may be multiple rows for one device.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 693: Database columns for `ImportedActiveSyncDevice` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* integer. Key. Generated ID The identifier used in the source connection for the computer.

Database Column	Details
`ActiveSyncID`	*Type:* text (max 512 characters). Key. Nullable The EASIdentity presented by the source, a combination of the AD user and the unique device ID.
`Domain`	*Type:* text (max 100 characters). Nullable The domain of the device. This may be a flat name or FQDN.
`DeviceID`	*Type:* text (max 100 characters). Nullable The unique device identifier.
`DeviceOS`	*Type:* text (max 100 characters). Nullable The device operating system.
`DeviceModel`	*Type:* text (max 100 characters). Nullable The device model.
`DeviceType`	*Type:* text (max 50 characters). Nullable The device type.
`DeviceUserAgent`	*Type:* text (max 100 characters). Nullable The device user agent; an ActiveSync client-specific value that may identify the device type.
`UserDisplayName`	*Type:* text (max 256 characters). Nullable The AD user display name.
`IMEI`	*Type:* text (max 256 characters). Nullable IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types.
`PhoneNumber`	*Type:* text (max 128 characters). Nullable The phone number of the device. Used for mobile devices.
`EmailAddress`	*Type:* text (max 256 characters). Nullable The user's primary email address.
`ExchangeServer`	*Type:* text (max 256 characters). Nullable The source exchange server for this information.
`WhenCreatedUTC`	*Type:* datetime. Nullable The date/time this partnership was created, in UTC.
`LastSyncAttemptTime`	*Type:* datetime. Nullable The last attempted sync time for this partnership, in UTC.
`LastSuccessSync`	*Type:* datetime. Nullable The last successful sync time for this partnership, in UTC.

ImportedAttributeMapping Table

The `ImportedAttributeMapping` table is used by the importer to link imported instance attributes with attributes in the `Attribute` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 694: Database columns for `ImportedAttributeMapping` table

Database Column	Details
`AttributeID`	*Type:* integer. Nullable The identifier for the instance attribute in the `Attribute` table.
`ExternalAttributeID`	*Type:* integer. Key. Nullable The identifier used in the source connection for the imported instance attribute.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier of a data source connection in the `ComplianceConnection` table.

ImportedClientAccessEvidence Table

The `ImportedClientAccessEvidence` table holds all of the client access evidence which has been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 695: Database columns for `ImportedClientAccessEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalAccessEvidenceID`	*Type:* big integer. Key The identifier of the client access evidence.
`ProductName`	*Type:* text (max 256 characters). Key. Nullable The name of the product being accessed by user or computer. This may include version and edition too.

Database Column	Details
`Version`	*Type:* text (max 72 characters). Key. Nullable The version of the installed product.
`Edition`	*Type:* text (max 50 characters). Nullable The edition of the installed product.
`UALRoleName`	*Type:* text (max 256 characters). Nullable The UAL role name of the product being accessed by user or computer. This is used when retrive data using UAL.
`UALRoleGUID`	*Type:* unique identifier. Nullable The UAL role GUID of the product being accessed by user or computer. This is used when retrive data using UAL
`Publisher`	*Type:* text (max 200 characters). Nullable The name of the publisher of the installed product. Defaulting to Microsoft as per recommendation.

ImportedClientAccessEvidenceMapping Table

The `ImportedClientAccessEvidenceMapping` is the mapping table for imported access evidence and access evidence

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 696: Database columns for `ImportedClientAccessEvidenceMapping` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalAccessEvidenceID`	*Type:* big integer. Key External Access evidend id. Foreign key to `ImportedClientAccessedAccessEvidence` table.
`AccessEvidenceID`	*Type:* integer. Key Access evidend id. Foreign key to `AccessEvidence` table.

ImportedClientAccessedAccessEvidence Table

The `ImportedClientAccessedAccessEvidence` table holds a record of the installer evidence that has been installed on a computer from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 697: Database columns for `ImportedClientAccessedAccessEvidence` table

Database Column	Details
`ImportedClientAccessed AccessEvidenceID`	*Type:* big integer. Key The identifier used in the source connection for the installer evidence.
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalAccessEvidenceID`	*Type:* big integer. Key Access evidence id .Foreign key to the `ImportedClientAccessEvidence` table.
`ExternalAccessingDeviceID`	*Type:* big integer. Key. Nullable Accessing computer id .Foreign key to the `ImportedAccessingDevice` table
`ExternalAccessingUserID`	*Type:* big integer. Key. Nullable Accessing userid. Foreign key to the `ImportedAccessingUser` table
`ExternalServerComputerID`	*Type:* big integer. Key. Nullable Server computer id .Foreign key to the `ImportedComputer` table.
`ClientAccessSource`	*Type:* text (max 100 characters). Key Referencing to the client access source type.

ImportedClientAccessedAccessOccurrence Table

The `ImportedClientAccessedAccessOccurrence` table holds the access information of device or user

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 698: Database columns for `ImportedClientAccessedAccessOccurrence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ImportedClientAccessed AccessEvidenceID`	*Type:* big integer. Key Access evidence id. Foreign key to the `ImportedClientAccessedAccessEvidence` table..
`AccessCount`	*Type:* integer Number of access frequency for given date
`InventoryDate`	*Type:* datetime. Key Date on which inventory occurance was recorded.
`LicenseDate`	*Type:* datetime. Key Date which will be used for licensing purpose.
`AccessDate`	*Type:* datetime. Nullable The access date.

ImportedCluster Table

The `ImportedCluster` table holds all of the clusters which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 699: Database columns for `ImportedCluster` table

Database Column	Details
`ExternalID`	*Type:* big integer. Key. Nullable The unique identifier for this imported cluster.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ClusterID`	*Type:* integer. Nullable The unique identifier for this imported cluster. Note that this maps to the 'ExternalID' column in the 'ImportedCluster' table, and not to the 'ClusterID' column.

Database Column	Details
`ExternalName`	*Type:* text (max 256 characters). Nullable The identifier of the cluster in the external cluster management system.
`Name`	*Type:* text (max 256 characters) The user-visible name of the cluster.
`Namespace`	*Type:* text (max 256 characters). Nullable The name of the domain/datacenter containing the cluster.
`ClusterTypeID`	*Type:* integer The type of cluster.
`InventoryDate`	*Type:* datetime. Nullable The date the cluster last had inventory reported.
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory.
`DRS`	*Type:* boolean. Nullable Whether Distributed Resource Scheduler (DRS) is enabled
`DPM`	*Type:* boolean. Nullable Whether Distributed Power Management (DPM) is enabled

ImportedClusterGroup Table

The `ImportedClusterGroup` table holds all of the group objects defined on clusters which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 700: Database columns for `ImportedClusterGroup` table

Database Column	Details
`ExternalID`	*Type:* big integer. Key. Nullable The unique identifier for this imported cluster group.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.

Database Column	Details
`ClusterID`	*Type:* integer. Nullable The assigned identifier for this cluster group.
`ClusterExternalID`	*Type:* big integer. Key The unique identifier for the imported cluster.
`Name`	*Type:* text (max 256 characters) The name of the cluster group.
`ClusterTypeID`	*Type:* integer Foreign key to the `ClusterType` table.

ImportedClusterGroupMember Table

The `ImportedClusterGroupMember` table holds all of the group memberships defined on clusters which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 701: Database columns for `ImportedClusterGroupMember` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ClusterGroupExternalID`	*Type:* big integer. Key The unique identifier for the imported cluster group.
`ComputerExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the external computer which is a member of the group.
`VCObjectID`	*Type:* text (max 256 characters). Key. Nullable The identifier of the virtual machine in Virtual Center.

ImportedClusterHostAffinityRule Table

The `ImportedClusterHostAffinityRule` table holds all of the host affinity rules for a cluster which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 702: Database columns for `ImportedClusterHostAffinityRule` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ClusterExternalID`	*Type:* big integer. Key The unique identifier for the imported cluster.
`Name`	*Type:* text (max 256 characters). Key The name of the cluster group.
`ClusterHostGroup ExternalID`	*Type:* big integer. Key The unique identifier for the imported cluster host group.
`ClusterVMGroupExternalID`	*Type:* big integer. Key The unique identifier for the imported cluster VM group.
`ClusterHostAffinity RuleTypeID`	*Type:* integer A unique identifier indicating a type of Cluster Host Affinity Rule.

ImportedClusterNode Table

The `ImportedClusterNode` table holds all of the cluster nodes which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 703: Database columns for `ImportedClusterNode` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ClusterExternalID`	*Type:* big integer. Key The unique identifier for the imported cluster.

Database Column	Details
`ComputerExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the external computer which is a member of the cluster.
`ClusterNodeTypeID`	*Type:* integer Foreign key to the `ClusterNodeType` table.

ImportedComputer Table

The `ImportedComputer` table holds all of the computers which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 704: Database columns for `ImportedComputer` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer.
`ComputerName`	*Type:* text (max 256 characters). Key. Nullable The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by `GetComputerName()`. For UNIX, it is the host name of the machine, as returned by `gethostname(2)`.
`Domain`	*Type:* text (max 100 characters). Key. Nullable The domain of the computer.
`OperatingSystem`	*Type:* text (max 128 characters). Nullable The operating system of the computer.
`ServicePack`	*Type:* text (max 128 characters). Nullable The service pack installed for the operating system.
`NumberOfProcessors`	*Type:* integer. Nullable The number of processors in the computer.

Database Column	Details
`ProcessorType`	*Type:* text (max 256 characters). Nullable The type of processor in the computer.
`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed of the fastest processor in the computer.
`NumberOfCores`	*Type:* integer. Nullable The number of cores in the computer.
`TotalMemory`	*Type:* big integer. Nullable The total RAM in the computer, in bytes.
`ChassisType`	*Type:* text (max 128 characters). Nullable The type of case of the computer. The value must be a (case insensitive) exact match for one of the values shown. Note that some license types use this information to optimize the licensing position, particularly with desktop and laptop computers.
`NumberOfHardDrives`	*Type:* integer. Nullable The number of hard drives in the computer.
`TotalDiskSpace`	*Type:* big integer. Nullable The total size of all hard drives in the computer.
`NumberOfNetworkCards`	*Type:* integer. Nullable The number of network cards in the computer.
`NumberOfDisplayAdapters`	*Type:* integer. Nullable The number of graphics cards in the computer.
`IPAddress`	*Type:* text (max 256 characters). Nullable The IP address of the computer.
`MACAddress`	*Type:* text (max 256 characters). Nullable The MAC address of the computer.

Database Column	Details
`Manufacturer`	*Type:* text (max 128 characters). Key. Nullable The manufacturer of the computer hardware. Some examples include: • On Windows, the SMBios manufacturer (the WMI Manufacturer property of the 'Win32_ComputerSystem' class). • On Linux, 'Manufacturer' in the 'System Information' section resulting from the 'dmidecode' command. Sample command: 'dmidecode -s system-manufacturer' • On Solaris x86, as for Linux, with failovers first to 'sysinfo SI_HW_PROVIDER' and then to 'ModelNo'. • On Solaris SPARC, the 'sysinfo SI_HW_PROVIDER'. Typically this value is 'Sun_Microsystems' or, more recently, 'Oracle Corporation'. Failover to the 'ModelNo'. • On HP-UX, the string literal 'HP'. • On AIX, the 'modelname' system attribute preceding the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use 'IBM'. This value is typically 'IBM'.
`ModelNo`	*Type:* text (max 128 characters). Nullable The model of the computer hardware or the virtual machine. This value is defined for the context of the current execution environment, rather than the physical server that may be hosting a virtual machine or partition. Examples: • On Windows, the SMBios product name. The WMI Model property of the Win32_ComputerSystem class. • On Linux, the SMBios product name read using the command 'dmidecode -s system-product-name'. Specifically, the 'System Information' section and the 'Product Name' in that section is used. • On Solaris x86, as for Linux, with failover to the 'sysinfo SI_PLATFORM', stripping 'SUNW', and replacing hyphen characters with space characters. • On Solaris SPARC, the 'openprom' "banner-name" value read from '/dev/openprom'. Failover to the 'sysinfo SI_PLATFORM', stripping 'SUNW', and replacing hyphen characters with space characters. • On HP-UX, the 'confstr _CS_MACHINE_MODEL'. • On AIX, the 'modelname' system attribute following the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use '8202-E4B'.

Database Column	Details
`SerialNo`	*Type:* text (max 100 characters). Nullable The hardware serial number of the computer. The goal of this value is to be tied to the physical hardware, partition or virtual machine and to be as unique as possible across all computers in the organization. This is due to its use in tracking computers, particularly after an operating system rebuild. This value is also used to socialize computer inventory from different inventory sources, and is used to map virtual machine guest operating system inventory to the VM host on which the virtual machine is running. Example sources: • On Windows, the SMBios serial number. The WMI 'SerialNumber' property of the 'Win32_BIOS' class. Can fail over to the 'SerialNumber' property of the 'Win32_SystemEnclosure' class which is typically the same value. • On Linux, the SMBios serial number read using the command 'dmidecode -s system-serial-number'. Specifically, the 'System Information' section and the 'Serial Number' in that section is used. • On Solaris 10 8/07 or later, for a non-global zone, the UUID value from the /etc/zones/index file. For a global zone, the same as Solaris 10 releases earlier than 8/07. • For Solaris 10 releases earlier than 8/07, the hexadecimal version of 'SI_HW_SERIAL' with an appended hyphen character followed by the Zone's name. For example, '838bfc7b-global' or '838bfc7b-myzone'. • For Solaris 8 and 9, The hexadecimal version of 'SI_HW_SERIAL'. • For Mac OS X, the serial number of the machine as printed on the packaging and found in "About this Mac" from the desktop. • For HP-UX, the 'confstr _CS_PARTITION_IDENT' partition identifier if it is an nPar or vPar, or '_CS_MACHINE_IDENT' if not; with a failover to the machine serial number, and a final failover to the 'uname' machine identification number. • For AIX, the 'id_to_partition' system attribute, starting from the third character (strips a '0X' from the start). For example, if the 'id_to_partition' system attribute is '0X0473409002F7B201' then use '0473409002F7B201'.

Database Column	Details
`HostID`	*Type:* text (max 100 characters). Nullable An identifier for the host of the computer (when inventorying a machine partition such as Solaris Zone, AIX lPar, HP-UX nPar/vPar). Examples: • For a Zone on Solaris, the hexadecimal version of SI_HW_SERIAL. • For nPar/vPar on HP-UX, the 'confstr _CS_MACHINE_IDENT' unique machine identifier. • For lPar on AIX, the 'modelname' system attribute following the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use '8202-E4B'.
`LastLoggedOnUser`	*Type:* text (max 128 characters). Nullable The DOMAIN/SAMAccountName of the user last logged onto the computer.
`InventoryDate`	*Type:* datetime. Nullable The date the computer last had inventory reported.
`HardwareInventoryDate`	*Type:* datetime. Nullable The date (and optionally time) when the hardware was last inventoried. For automated/scheduled data uploads through an inventory beacon, make sure that inventory dates are kept current, as they are used to report out-of-date inventory sources. For a one-time upload to the central application server, leave inventory dates empty (null). At each import from the saved file, the import date is used as the inventory date, which prevents the inventory becoming stale. Notice that this value is not available in the web interface.
`ServicesInventoryDate`	*Type:* datetime. Nullable The date when services (for example, Oracle) were last scanned on this computer. For automated/scheduled data uploads through an inventory beacon, make sure that inventory dates are kept current, as they are used to report out-of-date inventory sources. For a one-time upload to the central application server, leave inventory dates empty (null). At each import from the saved file, the import date is used as the inventory date, which prevents the inventory becoming stale.
`InventoryAgent`	*Type:* text (max 128 characters) The name of the person or tool that performed the last inventory. For imported spreadsheets, you may wish to include the name of the person preparing the data, in case there is subsequent follow-up required.
`ComplianceComputerID`	*Type:* integer. Key. Nullable Identifier of the computer in the `ComplianceComputer` table that this imported computer links to. This is populated by the import process and does not need to be provided by the source connections.

Database Column	Details
`ComplianceDomainID`	*Type:* integer. Key. Nullable Identifier of the domain in the `ComplianceDomain` table that this computer belongs to. This is populated by the import process and does not need to be provided by the source connections.
`IncompleteRecord`	*Type:* boolean. Nullable Used to identify records which do not have all information specified. Primarily used for `ManageSoft` source connections where the domain name was not reliably reported.
`NumberOfSockets`	*Type:* integer. Nullable The number of sockets in the computer.
`PartialNumberOfProcessors`	*Type:* decimal. Nullable The fractional processor count available to this computer.
`UntrustedSerialNo`	*Type:* boolean Is this computer known to have a serial number from a data source that should not be trusted.
`FullDetailsFromExternalID`	*Type:* big integer. Nullable If this computer is marked as incomplete, and some of its properties are updated from another computer, record the external ID if the full computer.
`FullDetailsFrom ComplianceConnectionID`	*Type:* integer. Nullable If this computer is marked as incomplete, and some of its properties are updated from another computer, record the connection ID if the full computer.
`ComplianceComputerTypeID`	*Type:* integer. Nullable If you know that the computer is a virtual machine or VM host, record that data here. If you are unsure, leave this cell empty (NULL): this allows the system to infer the computer type (for example, a computer with VMs linked to it is inferred to be a VM host). If data comes from multiple inventory sources, leaving this value as null also allows the value to be inserted from another source. So, unless there is a very good reason, do not just specify 'Computer', but allow the inference rules to help.

Database Column	Details
`ILMTAgentID`	*Type:* big integer. Key. Nullable The unique ID used by the IBM License Metric Tool (ILMT) inventory agent on this device, if the inventory source is aware of this value. This can be used to track a computer over time and can be used to socialize different inventory sources. Currently the ILMT and ManageSoft inventory adapters report this value. To find these values: • On Windows: The standalone and agent based ILMT configuration files are '$(WindowsFolder)/itlm/tlmstandalone.ini' and '$(WindowsFolder)/itlm/tlmagent.ini' respectively. Read the 'agentid' property from these files using a case-insensitive match against the property name. • On UNIX: The standalone and agent based ILMT configuration files are '/etc/tlmstandalone.ini' and '/etc/tlmagent.ini' respectively. Read the 'agentid' property from these files using a case-insensitive match against the property name.
`FNMPComputerUID`	*Type:* unique identifier. Key. Nullable The unique identifier generated for the computer from the IM database. This property should only be populated by the ManageSoft inventory adapter.
`HostIdentifyingNumber`	*Type:* text (max 128 characters). Key. Nullable Virtual hosts may have an identifier that is unique only across that hardware model. It is less unique than the true hardware serial number, for example.
`HostType`	*Type:* text (max 128 characters). Key. Nullable The type of the physical host computer. This value is similar to the model number, but it is always for the physical server that an execution context may be running on. Therefore, this will generally be a known value for standalone machines and partitions, but it will not be known for virtual machines. This value is used for matching computers. Examples: • 'i86pc' • 'Sun-Fire-T1000' • 'rx7620' • '785' (for a 9000/785/C3700) • '8202' (for an IBM,8202-E4B).
`NumberOfLogicalProcessors`	*Type:* integer. Nullable The number of logical processors in the computer.
`IsRemoteACLDevice`	*Type:* boolean. Key Used to determine if the current record is a remote ACL based device.

Database Column	Details
IsDuplicate	*Type:* boolean Used to identify that imported computer is a duplicate of another, whereby a new computer will not created.
LegacySerialNo	*Type:* text (max 100 characters). Nullable A previous serial number of this computer that can also be used for matching.
UUID	*Type:* unique identifier. Key. Nullable The BIOS UUID of the computer.
IMEI	*Type:* text (max 256 characters). Nullable IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types.
PhoneNumber	*Type:* text (max 128 characters). Nullable The phone number of the device. Used for mobile devices.
EmailAddress	*Type:* text (max 256 characters). Nullable The email address associated with the device. Typically used for mobile devices.
CalculatedUser	*Type:* text (max 128 characters). Nullable The domain/SAMAccountName of the calculated user. Some inventory systems calculate the user who owns a computer. For example, it might be the user who, over the last ten logins, logged in most often.
LastSuccessful InventoryDate	*Type:* datetime. Nullable For incremental imports, this represents the inventory date of the computer in the source at the time this record was last successfully imported. If the import procedure has failed, this may be different to the inventory date. At the end of a successful incremental import, this value is updated to match the inventory date. If no value is present in this field, either there has not been a successful import of this computer or the reader for this record is not using an incremental update model.
MDScheduleGeneratedDate	*Type:* datetime. Nullable The last time the managed device schedule was regenerated.
MDScheduleContainsPVUScan	*Type:* boolean. Nullable Does this managed device include an event in its current schedule for running extra IBM PVU hardware scans.
FirmwareSerialNumber	*Type:* text (max 100 characters). Nullable Serial number in the system firmware such as BIOS, EEPROM etc.
MachineID	*Type:* text (max 100 characters). Nullable For AIX, it is the System ID. For HP-UX, it is the Machine/Software ID. It is unset for other platforms.

Database Column	Details
IgnoredDueToLicense	*Type:* boolean True if this machine is not imported into compliance computer table due to license limitation
CloudServiceProvider	*Type:* text (max 256 characters). Nullable A unique identifier for a cloud service provider record.
CSPMetadataJsonBlob	*Type:* text (max 4000 characters). Nullable Contains the json blob for an Imported Computer record
AgentVersion	*Type:* text (max 32 characters). Nullable The version of the agent which generated the inventory package.

ImportedComputerCustomProperty Table

The ImportedComputerCustomProperty table is used by the importer to import custom properties for computers.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 705: Database columns for ImportedComputerCustomProperty table

Database Column	Details
ComplianceConnectionID	*Type:* integer. Key The identifier of a data source connection in the ComplianceConnection table.
ExternalID	*Type:* big integer. Key The identifier, in the source connection, of the computer that this property belongs to.
PropertyNameID	*Type:* integer. Key The identifier for custom property in the ImportedCustomPropertyName table.
PropertyValue	*Type:* text (max 256 characters) The value of the custom property.

ImportedComputerScriptResult Table

The ImportedComputerScriptResult table holds all of the script results which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 706: Database columns for `ImportedComputerScriptResult` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer.
`RecognitionRule`	*Type:* text (max 256 characters). Key. Nullable The recognition rule.
`Revision`	*Type:* integer. Nullable The revision number of the recognition rule.
`InventoryDate`	*Type:* datetime. Nullable The date the recognition rule ran.
`Result`	*Type:* text. Nullable The result of the recognition rule script.

ImportedCustomPropertyName Table

The `ImportedCustomPropertyName` table is used by the importer to store the names of custom properties.

Table 707: Database columns for `ImportedCustomPropertyName` table

Database Column	Details
`PropertyNameID`	*Type:* integer. Key. Generated ID A unique identifier for custom property.
`PropertyName`	*Type:* text (max 256 characters). Key The name of the custom property.

ImportedDomain Table

The `ImportedDomain` table holds all of the domains which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 708: Database columns for `ImportedDomain` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ComplianceDomainID`	*Type:* integer. Nullable Identifier of the domain in the `ComplianceDomain` table that this imported domain links to. This is populated as part of the import process and does not need to be provided by the source connections.
`QualifiedName`	*Type:* text (max 200 characters). Key. Nullable The fully qualified name of the domain.
`FlatName`	*Type:* text (max 200 characters). Key. Nullable The flat name of the domain.

ImportedEvidenceAttribute Table

The `ImportedEvidenceAttribute` table holds all of the instance attributes from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 709: Database columns for `ImportedEvidenceAttribute` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`AttributeID`	*Type:* integer. Key. Nullable The identifier used in the source connection for the instance attribute.
`AttributeName`	*Type:* text (max 256 characters). Key. Nullable The name of the instance attribute.

ImportedFNMEAFeature Table

The `ImportedFNMEAFeature` table is used by the importer to import FlexNet Manager for Engineering Applications features.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 710: Database columns for `ImportedFNMEAFeature` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer The identifier of a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* integer. Key. Generated ID The identifier of the feature from the external data source.
`Name`	*Type:* text (max 256 characters) The name for this feature.
`Version`	*Type:* text (max 32 characters). Nullable The version of this feature.
`Publisher`	*Type:* text (max 256 characters) The publisher of the feature.
`VendorDaemon`	*Type:* text (max 256 characters) The vendor daemon of the feature.
`ConsumedQuantity`	*Type:* integer The count of the feature installs.
`OutOfComplianceQuantity`	*Type:* integer The count of out-of-compliance feature installs, as calculated by FlexNet Manager for Engineering Applications.
`ComplianceStatus`	*Type:* text (max 32 characters) The compliance status of this feature, as calculated by FlexNet Manager for Engineering Applications.
`FNMEAFeatureID`	*Type:* integer. Nullable The identifier of the FlexNet Manager for Engineering Applications feature in the `FNMEAFeature` table that this imported FlexNet Manager for Engineering Applications feature links to. This is populated by the import process and does not need to be provided by the source connections.

ImportedFNMEAProduct Table

The `ImportedFNMEAProduct` table is used by the importer to import FlexNet Manager for Engineering Applications products.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 711: Database columns for `ImportedFNMEAProduct` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* text (max 256 characters). Key The identifier of the product from the external data source. This is the product number in FlexNet Manager for Engineering Applications.
`FeatureID`	*Type:* integer. Key The identifier (from the external data source) of the feature this product is associated with.
`Name`	*Type:* text (max 256 characters) The name for this product.
`Version`	*Type:* text (max 32 characters). Key The version of this product.
`VendorDaemon`	*Type:* text (max 256 characters). Key The vendor daemon of the products feature.
`Publisher`	*Type:* text (max 256 characters) The publisher of the product.
`PurchasedQuantity`	*Type:* integer The count of the products purchased.
`OutOfComplianceQuantity`	*Type:* integer The count of out-of-compliance product installs, as calculated by FlexNet Manager for Engineering Applications.
`ComplianceStatus`	*Type:* text (max 32 characters) The compliance status of this feature, as calculated by FlexNet Manager for Engineering Applications.

Database Column	Details
FeatureQuantity	*Type:* integer The count of the features available per product purchased.
SoftwareLicenseID	*Type:* integer. Nullable The identifier of the software license in the SoftwareLicense table that this imported FlexNet Manager for Engineering Applications product links to. This is populated by the import process and does not need to be provided by the source connections.

ImportedFNMEAUsageStatus Table

The ImportedFNMEAUsageStatus table is used by the importer to import FlexNet Manager for Engineering Applications status values.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 712: Database columns for ImportedFNMEAUsageStatus table

Database Column	Details
ComplianceConnectionID	*Type:* integer. Key The identifier of a data source connection in the ComplianceConnection table.
ProductNumber	*Type:* text (max 256 characters). Key The identifier of the product from the external data source. This is the product number in FlexNet Manager for Engineering Applications.
Version	*Type:* text (max 32 characters). Key The version of the product.
Publisher	*Type:* text (max 256 characters). Key The publisher of the product.
Month	*Type:* integer The month of the usage for this product.
Year	*Type:* integer The year of the usage of this product.
HWMUsage	*Type:* integer The high water mark usage of this product.

ImportedFileEvidence Table

The `ImportedFileEvidence` table holds all of the file evidence which has been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 713: Database columns for `ImportedFileEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalFileID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the file evidence.
`FileName`	*Type:* text (max 256 characters). Key. Nullable The name of the file used as evidence of software installation.
`FileVersion`	*Type:* text (max 100 characters). Nullable The version number of the file used as evidence of software installation.
`ProductVersion`	*Type:* text (max 200 characters). Nullable The product version number in the file header.
`ProductName`	*Type:* text (max 200 characters). Nullable The product name in the file header.
`FilePath`	*Type:* text (max 400 characters). Nullable The path of the file used as evidence of software installation.
`Company`	*Type:* text (max 100 characters). Key. Nullable The company in the file header.
`Description`	*Type:* text (max 200 characters) The description in the file header.
`FileSize`	*Type:* integer. Nullable The size of the file.
`Language`	*Type:* text (max 200 characters). Nullable The language in the file header.

Database Column	Details
AccessModeID	*Type:* integer. Key. Nullable The access mode ID of the file evidence.

ImportedFileEvidenceMapping Table

The ImportedFileEvidenceMapping table is used by the importer to link imported file evidence with evidence in the FileEvidence table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 714: Database columns for ImportedFileEvidenceMapping table

Database Column	Details
FileEvidenceID	*Type:* integer. Key. Nullable The identifier for the file evidence in the NewFileEvidence table.
ExternalFileID	*Type:* big integer. Key. Nullable The identifier used in the source connection for the imported file evidence.
ComplianceConnectionID	*Type:* integer. Key. Nullable The identifier of a data source connection in the ComplianceConnection table.

ImportedGuidMapping Table

The ImportedGuidMapping table is used by the importer to keep a history of entities that have been imported from a data source that uses GUID IDs rather than integer IDs.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 715: Database columns for ImportedGuidMapping table

Database Column	Details
ComplianceConnectionID	*Type:* integer. Key The identifier of a data source connection in the ComplianceConnection table.

Database Column	Details
`Category`	*Type:* text (max 100 characters). Key The importer category applicable for this ID space.
`OriginalID`	*Type:* unique identifier. Key The ID of this entity in the source database.
`MappedID`	*Type:* big integer. Generated ID A unique integer value we can use as an 'external ID' safely in the `ImportedComputer` table.

ImportedILMTPVUCounts Table

This table allows the summarised PVU sub capacity numbers to be imported from ILMT. These numbers are calculated by ILMT for a particular date range as PVU "reports".

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 716: Database columns for `ImportedILMTPVUCounts` table

Database Column	Details
`ExternalNodeID`	*Type:* big integer. Key The external ID of the server to which these points apply.
`ExternalVMID`	*Type:* big integer. Key. Nullable The external ID of the virtual machine associated with the node (server).
`ComplianceConnectionID`	*Type:* integer. Key The current connection ID for this data source.
`TitleName`	*Type:* text (max 512 characters). Key The name of the title these points apply to.
`Publisher`	*Type:* text (max 254 characters). Key The name of the publisher of the title these points apply to.
`SubCapacityCores`	*Type:* integer The number of sub-capacity licensable cores for the license on the computer.
`FullCapacityCores`	*Type:* integer The number of full-capacity licensable cores for the license on the computer.

Database Column	Details
SubCapacityPVU	*Type:* integer The number of sub-capacity PVU counts consumed for the license on the computer.
FullCapacityPVU	*Type:* integer The number of full-capacity PVU counts consumed for the license on the computer.
PeakSubCapacityPVU	*Type:* integer The peak number of sub-capacity PVU counts consumed for the license on the computer.
PeakFullCapacityPVU	*Type:* integer The peak number of full-capacity PVU counts consumed for the license on the computer.

ImportedILMTPVUCreatedLicenses Table

This table stores a history of IBM PVU licenses that have been created by the ILMT adapter.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 717: Database columns for `ImportedILMTPVUCreatedLicenses` table

Database Column	Details
SoftwareLicenseID	*Type:* integer The ID of the created license.
TitleName	*Type:* text (max 512 characters) The name of the title that triggered the creation of the license.
Publisher	*Type:* text (max 254 characters) The name of the publisher of the title that triggered the creation of the license.

ImportedILMTVMMapping Table

The `ImportedILMTVMMapping` table is used by the importer to keep a history of all Virtual Machine IDs (adm.VM records) that have been imported from ILMT data sources.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 718: Database columns for `ImportedILMTVMMapping` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`OriginalID`	*Type:* big integer. Key The agent ID of this agent in the ILMT database.
`MappedID`	*Type:* integer. Generated ID A unique integer value we can use as an 'external ID' safely in the `ImportedComputer` table.

ImportedInstalledFileEvidence Table

The `ImportedInstalledFileEvidence` table holds a record of the file evidence that has been installed on a computer from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 719: Database columns for `ImportedInstalledFileEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the file evidence is installed on.
`ExternalFileID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the file evidence.
`ExternalFilePathID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the path of the file evidence.

ImportedInstalledFileEvidencePath Table

The `ImportedInstalledFileEvidencePath` table holds a record of the path for file evidences that has been installed on a computer from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 720: Database columns for `ImportedInstalledFileEvidencePath` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalFilePathID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the path of the file evidence.
`ExternalFilePath`	*Type:* text (max 400 characters). Nullable The path to the installed file evidence.

ImportedInstalledFileEvidenceUsage Table

The `ImportedInstalledFileEvidenceUsage` table holds a record of end-users that are using file evidence from the source connection.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 721: Database columns for `ImportedInstalledFileEvidenceUsage` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`StartDate`	*Type:* text (max 10 characters). Nullable The start date of the file evidence usage tracking period.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the file evidence is installed on.

Database Column	Details
`ExternalUserID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the end-user that has used the file evidence.
`ExternalFileID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the file evidence.
`ActiveTimeInSeconds`	*Type:* big integer. Nullable The number of seconds that the file evidence was in use during the usage tracking period.
`NumberOfSessions`	*Type:* big integer. Nullable The number of sessions that the file evidence was in use during the usage tracking period.
`LastUsedDate`	*Type:* text (max 10 characters). Nullable The last used date of the file evidence.

ImportedInstalledInstallerEvidence Table

The `ImportedInstalledInstallerEvidence` table holds a record of the installer evidence that has been installed on a computer from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 722: Database columns for `ImportedInstalledInstallerEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalInstaller EvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the installer evidence is installed on.
`ExternalInstanceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the instance that the installer evidence is associated with.

Database Column	Details
`InstallDate`	*Type:* text (max 10 characters). Nullable The install date of the installer evidence.
`DiscoveryDate`	*Type:* text (max 10 characters). Nullable The date that the installer evidence was first seen.

ImportedInstalledInstallerEvidenceAttribute Table

The `ImportedInstalledInstallerEvidenceAttribute` table holds a record of the values of the instance attributes for each installer evidence which is reported to be installed on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 723: Database columns for `ImportedInstalledInstallerEvidenceAttribute` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalInstaller EvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the installer evidence is installed on.
`ExternalInstanceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the instance that the installer evidence is associated with.
`AttributeID`	*Type:* integer. Key The identifier used in the source connection for the instance attribute.
`Value`	*Type:* text The value of the instance attribute for the installed installer evidence.

ImportedInstalledInstallerEvidenceUsage Table

The `ImportedInstalledInstallerEvidenceUsage` table holds a record of installed evidence being used from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 724: Database columns for `ImportedInstalledInstallerEvidenceUsage` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`StartDate`	*Type:* text (max 10 characters). Nullable The start date of the installer evidence usage tracking period.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the installer evidence is installed on.
`ExternalInstallerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`ExternalInstanceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the instance that the installer evidence is associated with.
`NumberOfSessions`	*Type:* big integer. Nullable The number of sessions that the installer evidence was in use during the usage tracking period.
`LastUsedDate`	*Type:* text (max 10 characters). Nullable The last used date of the installed installer evidence.
`ExternalUserID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the user that the installer evidence was used on.

ImportedInstalledWMIEvidence Table

The `ImportedInstalledWMIEvidence` table holds a record of the WMI evidence that has been installed on a computer from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 725: Database columns for `ImportedInstalledWMIEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the WMI evidence is installed on.
`ExternalEvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the WMI evidence.
`InstanceName`	*Type:* text (max 256 characters). Key. Nullable The name of the WMI class instance used in the source connection for the WMI evidence

ImportedInstallerEvidence Table

The `ImportedInstallerEvidence` table holds all of the installer evidence which has been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 726: Database columns for `ImportedInstallerEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalInstallerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`DisplayName`	*Type:* text (max 256 characters). Key. Nullable The display name of the software as reported by the installer evidence.
`Version`	*Type:* text (max 72 characters). Key. Nullable The version of the software as reported by the installer evidence.
`Publisher`	*Type:* text (max 200 characters). Key. Nullable The publisher of the software as reported by the installer evidence.

Database Column	Details
`Evidence`	*Type:* text (max 32 characters). Nullable Identifier for the type of installer evidence.
`ProductCode`	*Type:* text (max 55 characters). Nullable The product code of the evidence. This is usually the MSI product code.
`AccessModeID`	*Type:* integer. Key. Nullable The access mode ID of the file evidence.

ImportedInstallerEvidenceMapping Table

The `ImportedInstallerEvidenceMapping` table is used by the importer to link imported installer evidence with evidence in the `InstallerEvidence` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 727: Database columns for `ImportedInstallerEvidenceMapping` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key. Nullable The identifier for the installer evidence in the `InstallerEvidence` table.
`ExternalInstallerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the imported installer evidence.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier of a data source connection in the `ComplianceConnection` table.

ImportedInstallerEvidenceRepackageMapping Table

The `ImportedInstallerEvidenceRepackageMapping` table is used by the importer to map the original and current installer evidence of repackaged softwares as reported by the ISO tag evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 728: Database columns for `ImportedInstallerEvidenceRepackageMapping` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier of a data source connection in the `ComplianceConnection` table.
`OrigDisplayName`	*Type:* text (max 256 characters). Key. Nullable The original display name of the repackaged software as reported by the ISO tag evidence.
`OrigVersion`	*Type:* text (max 72 characters). Key. Nullable The original version of the repackaged software as reported by the ISO tag evidence.
`OrigPublisher`	*Type:* text (max 200 characters). Key. Nullable The original publisher of the repackaged software as reported by the ISO tag evidence.
`CurrentDisplayName`	*Type:* text (max 256 characters). Key. Nullable The current display name of the repackaged software as reported by the ISO tag evidence.
`CurrentVersion`	*Type:* text (max 72 characters). Key. Nullable The current version of the repackaged software as reported by the ISO tag evidence.
`CurrentPublisher`	*Type:* text (max 200 characters). Key. Nullable The current publisher of the repackaged software as reported by the ISO tag evidence.

ImportedInstance Table

The `ImportedInstance` table holds all of the instances which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 729: Database columns for `ImportedInstance` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.

Database Column	Details
`InstanceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the instance.
`InstanceName`	*Type:* text (max 256 characters). Nullable The name of the instance.
`ParentInstanceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the parent instance.
`EnterpriseManager InstanceID`	*Type:* big integer. Nullable The identifier used in the source connection for the Oracle Enterprise Manager instance.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer.
`AuditEvidence`	*Type:* binary. Nullable Oracle LMS CVS files in zip archive.
`AuditEvidenceDate`	*Type:* datetime. Nullable Oracle LMS CSV files collection date.

ImportedInstanceUser Table

The `ImportedInstanceUser` table holds all of the end-users of an instance which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 730: Database columns for `ImportedInstanceUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key The identifier used in the source connection for the instance end-user.
`ComputerID`	*Type:* big integer. Key The identifier used in the source connection for the computer.

Database Column	Details
`InstanceID`	*Type:* big integer. Key The identifier used in the source connection for the instance.
`AccountStatus`	*Type:* text (max 256 characters). Nullable The current status of the end-user account.
`CreationDate`	*Type:* datetime. Nullable The date and time when the end-user was created.
`LastLogonDate`	*Type:* datetime. Nullable The date and time when the end-user last logged on to the computer.
`DefaultTablespace`	*Type:* text (max 256 characters). Nullable The default tablespace for an Oracle end-user.
`TempTablespace`	*Type:* text (max 256 characters). Nullable The temporary tablespace for an Oracle end-user.
`ApplicationID`	*Type:* text (max 400 characters). Key. Nullable The Oracle EBS application ID the user has access to.

ImportedMissingComputer Table

The `ImportedMissingComputer` table holds all of the computers which no longer have inventory records in the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 731: Database columns for `ImportedMissingComputer` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer.
`ComplianceComputerID`	*Type:* integer. Key. Nullable Identifier of the computer in the `ComplianceComputer` table that this imported computer links to.

Database Column	Details
`IsTerminated`	*Type:* boolean Flag to indicate that this imported computer links to a terminated `CloudServiceInstance`.

ImportedMissingLicenseUser Table

The `ImportedMissingLicenseUser` table holds all of the external end-users which no longer have inventory records in the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 732: Database columns for `ImportedMissingLicenseUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the external end-user.
`LicenseUserID`	*Type:* integer. Key. Nullable The identifier for the external end-user in the `LicenseUser` table.

ImportedMissingUser Table

The `ImportedMissingUser` table holds all of the end-users which no longer have inventory records in the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 733: Database columns for `ImportedMissingUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the end-user.
`ComplianceUserID`	*Type:* integer. Key. Nullable The identifier for the end-user in the `ComplianceUser` table.

ImportedPVUVirtualMachineLayer Table

The `ImportedPVUVirtualMachineLayer` table holds all of the computers which have been retrieved from the IM database.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 734: Database columns for `ImportedPVUVirtualMachineLayer` table

Database Column	Details
`ExternalID`	*Type:* integer. Key The identifier used in the source connection for the end-user.
`HostExternalID`	*Type:* integer. Key. Nullable The host item on which the layer resides, or the computer itself. Foreign key to the `ImportedPVUVirtualMachineLayer` table.
`ParentExternalID`	*Type:* integer. Key. Nullable The parent layer. Foreign key to the `ImportedPVUVirtualMachineLayer` table
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`FNMPComputerUID`	*Type:* unique identifier. Key. Nullable The unique identifier generated for the computer from the IM database. This property should only be populated by the ManageSoft inventory adapter.

Database Column	Details
VMPoolTypeID	*Type:* integer. Nullable The type of this VM pool. Foreign key to the VMPoolType table.
VMTypeID	*Type:* integer. Nullable The type of this virtual machine. Foreign key to the VMType table.
Name	*Type:* text (max 256 characters). Nullable The name of the layer (host/pool/VM).
Manufacturer	*Type:* text (max 128 characters). Nullable The manufacturer of this layer.
ModelNo	*Type:* text (max 128 characters). Nullable The model number of this layer.
SerialNo	*Type:* text (max 100 characters). Nullable The serial number of this layer.
IsFabricatedHost	*Type:* boolean Is the host generated from the virtual machine inventory.
PartialNumberOfProcessors	*Type:* decimal. Nullable The fractional processor count available to this layer.
ProcessorType	*Type:* text (max 256 characters). Nullable The type of processor in this layer.
MaxClockSpeed	*Type:* integer. Nullable The maximum clock speed (in megahertz) of the fastest processor in this layer.
NumberOfProcessors	*Type:* decimal. Nullable The processor count for this layer.
NumberOfCores	*Type:* decimal. Nullable The core count for this layer.
MaxNumberOfLogical Processors	*Type:* decimal. Nullable The maximum number of logical processors count for this layer.
NumberOfLogicalProcessors	*Type:* decimal. Nullable The thread count for this layer.
LicenseSimulationRow TypeID	*Type:* integer The type of hardware for this item. Foreign key to the LicenseSimulationRowType table.

ImportedProductCodeEvidenceMapping Table

The `ImportedProductCodeEvidenceMapping` table is used by the importer to link imported product code evidence with evidence in the `InstallerEvidence` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 735: Database columns for `ImportedProductCodeEvidenceMapping` table

Database Column	Details
`InstallerEvidenceID`	*Type:* integer. Key. Nullable The identifier for the installer evidence in the `InstallerEvidence` table.
`ExternalInstallerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the imported installer evidence.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier of a data source connection in the `ComplianceConnection` table.

ImportedRelatedInstalledInstallerEvidence Table

The `ImportedRelatedInstalledInstallerEvidence` table holds parent-child relationship between installer evidence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 736: Database columns for `ImportedRelatedInstalledInstallerEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ParentExternal InstallerEvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`ParentExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the installer evidence is installed on.

Database Column	Details
`ChildExternalInstallerEvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`ChildExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer that the installer evidence is installed on.
`IsCharged`	*Type:* boolean. Key. Nullable The identifier used in the source connection to determine the pricing relation between parent and child installer evidence (specifies if it is charged = 1 or free = 0).
`ConfidenceLevel`	*Type:* integer. Nullable Confidence level for each bundled installer evidence (as a percentage).

ImportedRemoteApplication Table

This `ImportedRemoteApplication` table stores all the published applications from Citrix XenApp/App-V Management Server.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 737: Database columns for `ImportedRemoteApplication` table

Database Column	Details
`FarmName`	*Type:* text (max 256 characters). Nullable The farm from which the application belongs to.
`AppID`	*Type:* text (max 256 characters). Key. Nullable The unique identifier for XenApp applications.
`AppName`	*Type:* text (max 256 characters). Nullable The application name available in XenApp.
`AppFileName`	*Type:* text (max 256 characters). Key. Nullable The application executable name.
`AppFileVersion`	*Type:* text (max 256 characters). Key. Nullable The application executable version.
`AppFilePublisher`	*Type:* text (max 256 characters). Key. Nullable The application publisher.

Database Column	Details
`AppFileDescription`	*Type:* text (max 256 characters). Key. Nullable The application description.
`IsStreamingProfile`	*Type:* boolean. Nullable Whether the application is a streaming profile.
`AccessModeID`	*Type:* integer. Key The access mode of the virtual application.

ImportedRemoteApplicationAccess Table

This `ImportedRemoteApplicationAccess` table stores all users/groups with sid who have access to what virtual applications.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 738: Database columns for `ImportedRemoteApplicationAccess` table

Database Column	Details
`FarmName`	*Type:* text (max 256 characters). Nullable The farm from which the virtual application belongs to.
`AppID`	*Type:* text (max 256 characters). Nullable The unique identifier for virtual applications.
`Sid`	*Type:* text (max 256 characters). Nullable The sid that has access to the application.
`AccessModeID`	*Type:* integer The access mode of the virtual application.
`DeliveryGroupID`	*Type:* text (max 256 characters). Nullable The unique identifier for delivery group for the virtual applications.
`ApplicationGroupID`	*Type:* text (max 256 characters). Nullable The unique identifier for application group for the virtual applications.

ImportedRemoteApplicationInstallerData Table

This `ImportedRemoteApplicationInstallerData` table stores all the MSI information in a streamed profile.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 739: Database columns for `ImportedRemoteApplicationInstallerData` table

Database Column	Details
`FarmName`	*Type:* text (max 256 characters). Nullable The farm from which the application belongs to.
`AppID`	*Type:* text (max 256 characters). Key. Nullable The unique identifier for virtual applications.
`DisplayName`	*Type:* text (max 256 characters). Key. Nullable The application name.
`Publisher`	*Type:* text (max 200 characters). Key. Nullable The application publisher name.
`Version`	*Type:* text (max 72 characters). Key. Nullable The application version.
`ProductCode`	*Type:* text (max 55 characters). Nullable The product code of the evidence. This is usually the MSI product code.
`AccessModeID`	*Type:* integer. Key The access mode of the virtual application.

ImportedRemoteApplicationServer Table

This `ImportedRemoteApplicationServer` table stores the servers from which applications are published from.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 740: Database columns for `ImportedRemoteApplicationServer` table

Database Column	Details
`FarmName`	*Type:* text (max 256 characters). Nullable The farm from which the server belongs to.
`AppID`	*Type:* text (max 256 characters). Key. Nullable The unique identifier for XenApp applications.

Database Column	Details
ServerName	*Type:* text (max 256 characters). Key. Nullable The XenApp server the application is available under.
ServerDomainName	*Type:* text (max 256 characters). Key. Nullable The XenApp server domain name.
VDIGroupUUID	*Type:* unique identifier. Nullable The desktop group UUID from which the application is published
AccessModeID	*Type:* integer. Key The access mode of the virtual application.

ImportedRemoteServerFileEvidenceMapping Table

The ImportedRemoteServerFileEvidenceMapping table stores the mapping between file evidence on servers to software titles

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 741: Database columns for ImportedRemoteServerFileEvidenceMapping table

Database Column	Details
ComplianceConnectionID	*Type:* integer. Key. Nullable The identifier for a data source connection in the ComplianceConnection table.
ExternalServerID	*Type:* big integer. Key. Nullable The External Server ID for the remote server.
ExternalFileID	*Type:* big integer. Key. Nullable The identifier used in the source connection for the file evidence.
SoftwareTitleID	*Type:* integer. Nullable The software title ID corresponding to the piece of file evidence.

ImportedRemoteUsage Table

This ImportedRemoteUsage table stores the remote usage for applications in remote hosting environments

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 742: Database columns for `ImportedRemoteUsage` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalServerID`	*Type:* big integer. Key. Nullable The External Server ID for the remote server.
`ExternalClientID`	*Type:* big integer. Nullable The External client ID for the remote client machine.
`ExternalFileID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the file evidence.
`ExternalInstaller EvidenceID`	*Type:* big integer. Nullable The identifier used in the source connection for the installer evidence.
`ExternalUserID`	*Type:* big integer. Nullable The identifier used in the source connection for the end-user that has used the file evidence.
`StartDate`	*Type:* text (max 10 characters). Nullable The start date of the remote usage tracking period.
`ActiveTimeInSeconds`	*Type:* big integer. Nullable The number of seconds that the file evidence was in use during the usage tracking period.
`NumberOfSessions`	*Type:* big integer. Nullable The number of sessions that the file evidence was in use during the usage tracking period.
`AccessModeID`	*Type:* integer. Nullable The access mode ID for the remote usage.

ImportedRemoteUserToApplicationAccess Table

The `ImportedRemoteUserToApplicationAccess` table stores the applications that remote users have access to

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying*

table to produce this view of data for the single, selected tenant.

Table 743: Database columns for `ImportedRemoteUserToApplicationAccess` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalServerID`	*Type:* big integer. Key. Nullable The External Server ID for the remote server.
`VDIGroupUUID`	*Type:* unique identifier. Nullable The desktop group UUID from which the application is published
`ExternalFileID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the file evidence.
`ExternalInstaller EvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the installer evidence.
`ExternalUserID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the end-user that has used the file evidence.
`AccessModeID`	*Type:* integer. Key. Nullable The access mode ID for the remote application access.
`LastUsedDate`	*Type:* datetime. Key. Nullable The last time the remote application was used by the user.

ImportedSite Table

The `ImportedSubnet` contains sites imported from Microsoft Active Directory

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 744: Database columns for `ImportedSite` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.

Database Column	Details
`Name`	*Type:* text (max 256 characters). Key The site's name.
`AutoPopulated`	*Type:* boolean Is the site auto populated at source?
`Enabled`	*Type:* boolean Is the site enabled?

ImportedSiteSubnet Table

The `ImportedSiteSubnet` contains sites and subnets imported from Microsoft Active Directory

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 745: Database columns for `ImportedSiteSubnet` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`SiteName`	*Type:* text (max 256 characters). Key The site's name.
`IPSubnet`	*Type:* text (max 64 characters). Key The IP subnet.
`IPSubnetBits`	*Type:* tiny integer. Key The IP subnet mask in CIDR notation.
`AutoPopulated`	*Type:* boolean Is the subnet auto populated at source?
`Enabled`	*Type:* boolean Is the subnet enabled?

ImportedSoftwareLicense Table

The `ImportedSoftwareLicense` table holds all of the licenses which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 746: Database columns for `ImportedSoftwareLicense` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalLicenseID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the license.
`LicenseName`	*Type:* text (max 256 characters). Nullable The name of the license.
`SoftwareLicenseTypeID`	*Type:* integer. Nullable The license type ID of the license.
`EntitlementCount`	*Type:* integer. Nullable The number of entitlements for the license.
`IsSubscription`	*Type:* boolean Indicates whether or not the license is a subscription license.
`ExpiryDate`	*Type:* datetime. Nullable The expiry date of a subscription license.
`PartNo`	*Type:* text (max 100 characters). Nullable The publisher's part number for this license.
`SoftwareLicenseID`	*Type:* integer. Nullable Identifier of the license in the `SoftwareLicense` table that this imported license links to. This is populated by the import process and does not need to be provided by the source connections.
`VirtualApplication AccessMaximumUsagePeriod`	*Type:* integer. Nullable This is a rule for virtual application access. This is used in conjunction with the AllowExternalRoamingUse. For Device licenses, a license will consume 1 entitlement per each user device when used in period specified here. For user licenses, if 1, this license will consume only when used in period specified here.
`PublisherName`	*Type:* text (max 64 characters). Nullable The name of the publisher.
`Version`	*Type:* text (max 60 characters). Nullable The name of the Version.

Database Column	Details
`Edition`	*Type:* text (max 60 characters). Nullable The name of the Edition.
`TrueUp`	*Type:* boolean Set this field to `True` if the license is a true-up license (and so never goes into at risk).
`UnlimitedConsumption`	*Type:* boolean Set this field to `True` if the license is a unlimited license (and so never goes into at risk).

ImportedSoftwareLicenseAllocation Table

The `ImportedSoftwareLicenseAllocation` table holds the links between licenses and end-users which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 747: Database columns for `ImportedSoftwareLicenseAllocation` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalLicenseID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the license. Foreign key to the `ImportedSoftwareLicense` table.
`ExternalUserID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the user. Foreign key to the `ImportedUser` table.
`ExternalAccessingUserID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the accessing user. Foreign key to the `ImportedAccessingUser` table.

ImportedSoftwareTitle Table

The `ImportedSoftwareTitle` table contains the application titles retrieved from the source connection.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 748: Database columns for `ImportedSoftwareTitle` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalSoftwareTitleID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for a Software title.
`FullName`	*Type:* text (max 512 characters). Nullable Software title.
`ProductName`	*Type:* text (max 200 characters). Nullable Software title product name.
`PublisherName`	*Type:* text (max 200 characters). Nullable Software title publisher.
`VersionName`	*Type:* text (max 50 characters). Nullable Software title version.
`EditionName`	*Type:* text (max 50 characters). Nullable Software title edition.
`SoftwareTitleID`	*Type:* integer. Nullable Identifier of the software title in the `SoftwareTitle` table that this software title links to. This is populated by the import process and does not need to be provided by the source connections.

ImportedSoftwareTitleAccessEvidence Table

The `ImportedSoftwareTitleAccessEvidence` table maps ClientAccessEvidence to SoftwareTitles retrieved from the source connection.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 749: Database columns for `ImportedSoftwareTitleAccessEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalSoftwareTitleID`	*Type:* big integer. Key A unique identifier for a software record. Foreign key to the `ImportedSoftwareTitle` table.
`ExternalClientAccess EvidenceID`	*Type:* big integer. Key A unique identifier for a client access evidence record. Foreign key to the `ImportedClientAccessEvidence` table.

ImportedSoftwareTitleLicense Table

The `ImportedSoftwareTitleLicense` table maps SoftwareLicenses to SoftwareTitles retrieved from the source connection.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 750: Database columns for `ImportedSoftwareTitleLicense` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalSoftwareTitleID`	*Type:* big integer. Key A unique identifier for a software title record. Foreign key to the `ImportedSoftwareTitle` table.
`ExternalSoftwareLicenseID`	*Type:* big integer. Key A unique identifier for a software license record. Foreign key to the `ImportedSoftwareLicense` table.

ImportedStringMapping Table

The `ImportedStringMapping` table is used by the importer to keep a history of entities that have been imported from a data source that uses string IDs rather than integer IDs.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 751: Database columns for `ImportedStringMapping` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`Category`	*Type:* text (max 100 characters). Key The importer category applicable for this ID space.
`OriginalID`	*Type:* text (max 400 characters). Key The ID of this entity in the source database.
`MappedID`	*Type:* big integer. Generated ID A unique integer value we can use as an 'external ID' safely in the `ImportedComputer` table.

ImportedStringMappingLatin1CS Table

The `ImportedStringMappingLatin1CS` table is used by the importer to keep a history of entities that have been imported from a data source that uses case sensitive string IDs rather than integer IDs.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 752: Database columns for `ImportedStringMappingLatin1CS` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`Category`	*Type:* text (max 100 characters). Key The importer category applicable for this ID space.
`OriginalID`	*Type:* text (max 400 characters). Key The ID of this entity in the source database.
`MappedID`	*Type:* big integer. Generated ID A unique integer value we can use as an 'external ID' safely in the `ImportedComputer` table.

ImportedTDSComputer Table

The `ImportedTDSComputer` table holds computer properties imported from TDS

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 753: Database columns for `ImportedTDSComputer` table

Database Column	Details
`ExternalID`	*Type:* big integer. Key. Generated ID The identifier used in the source connection for the computer.
`ResourceID`	*Type:* text (max 64 characters). Key Identifier of the computer in TDS.
`ProductTechnopediaID`	*Type:* text (max 64 characters). Nullable Computer product ID in Technopedia.
`ModelTechnopediaID`	*Type:* text (max 64 characters). Nullable Computer model ID in Technopedia.

ImportedUser Table

The `ImportedUser` table holds all of the end-users which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 754: Database columns for `ImportedUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the end-user.
`UserName`	*Type:* text (max 64 characters). Nullable The account name of the end-user.

Database Column	Details
`Domain`	*Type:* text (max 100 characters). Key. Nullable The domain of the end-user.
`SAMAccountName`	*Type:* text (max 64 characters). Key. Nullable The SAM account name of the end-user.
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory. For imported spreadsheets, you may wish to include the name of the person preparing the data, in case there is subsequent follow-up required.
`FirstName`	*Type:* text (max 128 characters). Nullable The first name of the end-user.
`LastName`	*Type:* text (max 128 characters). Nullable The last name or surname of the end-user.
`Email`	*Type:* text (max 200 characters). Nullable The email address of the end-user.
`EmployeeNumber`	*Type:* text (max 128 characters). Nullable The employee number of the end-user.
`CostCenter`	*Type:* text (max 128 characters). Nullable The cost center of the end-user, as reported in SAP. Does not necessarily map to a cost centre in the `GroupEx` table.
`ComplianceUserID`	*Type:* integer. Nullable Identifier of the end-user in the `ComplianceUser` table that this imported user links to. This is populated by the import process and does not need to be provided by the source connections.
`ComplianceDomainID`	*Type:* integer. Nullable Identifier of the domain in the `ComplianceDomain` table that this end-user belongs to. This is populated by the import process and does not need to be provided by the source connections.
`IsBlacklisted`	*Type:* boolean. Key This is populated by the import process and does not need to be provided by the source connections. The field is set to `True` if the end-user matches a record from the `UserNameBlacklist` table, meaning the account should not be included in compliance calculations.
`MapUsingEmailAddress`	*Type:* boolean Indicates whether or not the user's email address should be used to try and map it to an existing `ComplianceUser` record.

ImportedVDI Table

The `ImportedVDIUser` table stores the list of VDI devices, their master VM template and the VDI group the VDI device resides under.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 755: Database columns for `ImportedVDI` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer The identifier of a data source connection in the `ComplianceConnection` table.
`ExternalDeviceID`	*Type:* big integer. Nullable The identifier used in the source connection for the VDI device.
`ComputerName`	*Type:* text (max 64 characters). Nullable The computer name of the VDI.
`Domain`	*Type:* text (max 100 characters). Nullable The domain name of the VDI device.
`VDIGroupName`	*Type:* text (max 100 characters). Key. Nullable The VDI group the VDI device belongs to.
`TemplateName`	*Type:* text (max 100 characters). Key. Nullable The VDI template the VDI is cloned from.
`SiteName`	*Type:* text (max 256 characters). Key. Nullable The site name of the VDI.
`BrokerType`	*Type:* text (max 64 characters). Key. Nullable The broker type of the VDI device.
`IsPersistent`	*Type:* boolean. Key. Nullable Determine whether the VDI device is a persistent VDI device.
`VDIGroupUUID`	*Type:* unique identifier. Nullable The group UUID the VDI device belongs to.
`ApplicationDeliveryOnly`	*Type:* boolean. Nullable Determines whether the VDI device is used only to server applications.

ImportedVDIEndPointAccess Table

The `ImportedVDIEndPointAccess` table stores the list of users on end-points that have accessed VDI devices.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 756: Database columns for `ImportedVDIEndPointAccess` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Nullable The identifier of a data source connection in the `ComplianceConnection` table.
`ExternalDeviceID`	*Type:* big integer. Nullable The identifier used in the source connection for the device.
`ExternalUserID`	*Type:* big integer. Nullable The identifier used in the source connection for the user.
`VDIDeviceName`	*Type:* text (max 64 characters). Nullable The computer name of the VDI device.
`VDIDeviceDomain`	*Type:* text (max 100 characters). Nullable The domain name of the VDI device.
`VDITemplateName`	*Type:* text (max 256 characters). Nullable The VDI template the VDI device was cloned from.
`LogonTime`	*Type:* datetime. Key. Nullable The logon time of the VDI device by the user.
`BrokerType`	*Type:* text (max 64 characters). Nullable The broker type of the VDI device.

ImportedVDITemplate Table

The `ImportedVDITemplate` table stores the list of VDI templates.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 757: Database columns for `ImportedVDITemplate` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer The identifier of a data source connection in the `ComplianceConnection` table.
`TemplateName`	*Type:* text (max 64 characters). Key. Nullable The template name of the VDI template.
`SiteName`	*Type:* text (max 256 characters). Key. Nullable The site name of the VDI.
`BrokerType`	*Type:* text (max 64 characters). Key. Nullable The broker type of the VDI template.
`VDITemplateExternalID`	*Type:* big integer. Nullable The ExternalID of the VDI template in the `ImportedComputer` table.

ImportedVDIUser Table

The `ImportedVDIUser` table stores the list of users that have been granted access to VDI groups.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 758: Database columns for `ImportedVDIUser` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier of a data source connection in the `ComplianceConnection` table.
`ExternalUserID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the end-user that has access to the VDI.
`VDIGroupName`	*Type:* text (max 100 characters). Nullable The VDI group the end-user has access to.
`SiteName`	*Type:* text (max 256 characters). Nullable The site name of the VDI.
`BrokerType`	*Type:* text (max 64 characters). Nullable The broker type of the VDI for the end user.

ImportedVMHostDatastore Table

The `ImportedVMHostDatastore` table holds all of the datastore objects available to virtual machines hosts.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 759: Database columns for `ImportedVMHostDatastore` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the virtual machine's host computer ID.
`Datastore`	*Type:* text (max 64 characters). Nullable The datastore availaboe on the VM host.

ImportedVMHostManagedBySoftware Table

The `ImportedVMHostManagedBySoftware` table contains relationships between installer evidence of management software and VM hosts it manages.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 760: Database columns for `ImportedVMHostManagedBySoftware` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalInstallerID`	*Type:* big integer. Key The identifier used in the source connection for an installer evidence of management software.

Database Column	Details
`ExternalComputerID`	*Type:* big integer. Key The identifier used in the source connection for the computer that the management software installer evidence is installed on.
`RelationType`	*Type:* text (max 100 characters). Key Identifier for the type of relation, to be matched against ImporterString column of `RelationType` table.
`ExternalVMHostID`	*Type:* big integer. Key The identifier used in the source connection for the VM host computer that is managed by a management software.

ImportedVMHostProperty Table

The `ImportedVMHostProperty` table holds additional properties for virtual machines hosts which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 761: Database columns for `ImportedVMHostProperty` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the virtual machine's host computer ID.
`VMTypeID`	*Type:* big integer. Nullable The VMHost technology type. Foreign key to the `VMType` table.
`HypervisorVersion`	*Type:* text (max 32 characters). Nullable The hypervisor version of the VM host.
`HyperThreadingEnabled`	*Type:* boolean. Nullable Set this to `True` if this VM host has hyper threading enabled.
`PowerState`	*Type:* text (max 32 characters). Nullable The power state of the VM host.

Database Column	Details
`ManagingSoftwareVersion`	*Type:* text (max 32 characters). Nullable The version of the managing software for the VM host.
`ConnectionState`	*Type:* text (max 32 characters). Nullable The connection state of the VM host to the managing software envrionment.
`InstanceType`	*Type:* text (max 256 characters). Nullable Cloud provider instance type.
`Region`	*Type:* text (max 256 characters). Nullable Region of the host.
`AvailabilityZone`	*Type:* text (max 256 characters). Nullable Location of the host.
`AllocationTime`	*Type:* datetime. Nullable The time that the Dedicated Host was allocated.
`ReleaseTime`	*Type:* datetime. Nullable The time that the Dedicated Host was released.
`Autoplacement`	*Type:* boolean. Nullable Whether auto-placement is on or off.

ImportedVMPool Table

The `ImportedVMPool` table holds all of the virtual machine pools which have been retrieved from the source connections and the number of processors and cores that are assigned to each pool.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 762: Database columns for `ImportedVMPool` table

Database Column	Details
`PoolName`	*Type:* text (max 100 characters). Key. Nullable The name of the pool.
`VCObjectID`	*Type:* text (max 256 characters). Nullable The identifier of the virtual machine folder in Virtal Center.
`ParentName`	*Type:* text (max 100 characters). Nullable The name of the parent pool. This is the PoolName property for the parent pool.

Database Column	Details
`PoolFriendlyName`	*Type:* text (max 256 characters). Nullable The friendly name of the pool.
`HostComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer which is hosting the pool.
`ObjectType`	*Type:* text (max 256 characters). Key. Nullable The type of pool.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ParentObjectType`	*Type:* text (max 256 characters). Nullable The type of pool of the parent.
`NumberOfProcessors`	*Type:* decimal. Nullable The number of processors available to this pool.
`NumberOfCores`	*Type:* decimal. Nullable The number of cores available to this pool.
`NumberOfLogicalProcessors`	*Type:* integer. Nullable The active number of threads used by this pool.
`MaxNumberOfLogical Processors`	*Type:* integer. Nullable Maximum number of threads allocated to this pool of type processor set.

ImportedVirtualMachine Table

The `ImportedVirtualMachine` table holds all of the virtual machines which have been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 763: Database columns for `ImportedVirtualMachine` table

Database Column	Details
`HostComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the virtual machine's host computer.

Database Column	Details
VirtualMachineType	*Type:* text (max 100 characters). Nullable The type of virtual machine.
VMName	*Type:* text (max 256 characters). Nullable The name of the virtual machine.
VCObjectID	*Type:* text (max 256 characters). Nullable The identifier of the virtual machine in Virtual Center.
FriendlyName	*Type:* text (max 256 characters). Nullable The friendly name of the virtual machine.
ComputerName	*Type:* text (max 256 characters). Nullable The computer name of the virtual machine.
UUID	*Type:* text (max 256 characters). Key. Nullable The UUID of the virtual machine.
TotalMemory	*Type:* big integer. Nullable The total RAM in the computer, in bytes.
PoolName	*Type:* text (max 100 characters). Nullable The name of the pool that the virtual machine belongs to.
CPUUsage	*Type:* integer. Nullable The maximum CPU usage of the virtual machine (MHz).
MemoryUsage	*Type:* big integer. Nullable The maximum memory usage of the virtual machine (bytes).
MaxNumberOfLogical Processors	*Type:* decimal. Nullable The maximum number of threads this VM is allowed to access.
VMEnabledStateID	*Type:* integer. Nullable The state of the machine (powered on, off, etc).
ModelNo	*Type:* text (max 128 characters). Nullable The model number of the virtual machine.

Database Column	Details
`Manufacturer`	*Type:* text (max 128 characters). Nullable The manufacturer of the computer hardware. Some examples include: • On Windows, the SMBios manufacturer (the WMI Manufacturer property of the 'Win32_ComputerSystem' class). • On Linux, 'Manufacturer' in the 'System Information' section resulting from the 'dmidecode' command. Sample command: 'dmidecode -s system-manufacturer' • On Solaris x86, as for Linux, with failovers first to 'sysinfo SI_HW_PROVIDER' and then to 'ModelNo'. • On Solaris SPARC, the 'sysinfo SI_HW_PROVIDER'. Typically this value is 'Sun_Microsystems' or, more recently, 'Oracle Corporation'. Failover to the 'ModelNo'. • On HP-UX, the string literal 'HP'. • On AIX, the 'modelname' system attribute preceding the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use 'IBM'. This value is typically 'IBM'.
`NumberOfProcessors`	*Type:* integer. Nullable The number of processors in the virtual machine.
`ProcessorType`	*Type:* text (max 256 characters). Nullable The type of processor in the virtual machine.
`NumberOfHardDrives`	*Type:* integer. Nullable The number of hard drives in the virtual machine.
`NumberOfNetworkCards`	*Type:* integer. Nullable The number of network cards in the virtual machine.
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`VMLocation`	*Type:* text (max 256 characters). Nullable Location of the virtual machine on the file system.
`GuestFullName`	*Type:* text (max 256 characters). Nullable Configured operating system for the guest.

Database Column	Details
`VMComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the virtual machine's computer.
`PoolType`	*Type:* text (max 100 characters). Nullable The type of the pool that the virtual machine belongs to.
`ZoneResourceManagement MethodType`	*Type:* text (max 100 characters). Nullable Resource management method used for this virtual machine in Solaris Zone.
`AffinityEnabled`	*Type:* boolean Set this to `True` if this VM is unable to move to different host computers.
`CPUAffinity`	*Type:* text (max 256 characters). Nullable Contains the CPU Affinity value for virtual machine(Host Logical Processors)
`CoreAffinity`	*Type:* text (max 256 characters). Nullable Contains the Core Affinity value for virtual machine
`PartitionID`	*Type:* text (max 100 characters). Nullable Partition ID generated and used by the managing virtualization platform
`PartitionNumber`	*Type:* integer. Nullable Number of this partition
`FullComputerName`	*Type:* text (max 256 characters). Nullable The virtual machine full computer name as determined by the VM guest managing tool.
`IPAddress`	*Type:* text (max 256 characters). Nullable IP Address of the virtual machine as determined by the VM guest managing tool.

ImportedWMIEvidence Table

The `ImportedWMIEvidence` table holds all of the WMI evidence which has been retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 764: Database columns for `ImportedWMIEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ClassName`	*Type:* text (max 50 characters). Key. Nullable The WMI class name of the WMI evidence.
`PropertyName`	*Type:* text (max 50 characters). Key. Nullable The WMI property name of the WMI evidence.
`PropertyValue`	*Type:* text (max 256 characters). Key. Nullable The value of the property of the WMI evidence.
`ExternalEvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the WMI evidence.

ImportedWMIEvidenceRuleMapping Table

The `ImportedWMIEvidenceRuleMapping` table is used by the importer to link imported WMI evidence with evidence in the `WMIEvidence` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 765: Database columns for `ImportedWMIEvidenceRuleMapping` table

Database Column	Details
`EvidenceRuleID`	*Type:* integer. Nullable The identifier for the WMI evidence in the `WMIEvidence` table.
`ExternalEvidenceID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the imported WMI evidence.
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier of a data source connection in the `ComplianceConnection` table.

ImporterValueMapping Table

The `ImporterValueMapping` table stores mapping pairs for use by importer tasks. It serves as a basic lookup

translation table that is not connection-specific.

Table 766: Database columns for `ImporterValueMapping` table

Database Column	Details
`ImporterValueMappingID`	*Type:* integer. Key. Generated ID Unique auto-incrementing identifier.
`Category`	*Type:* text (max 100 characters). Key The importer section applicable for this key, uses dotted notation: e.g. “MobileDevice.Apple.Model”.
`FromValue`	*Type:* text (max 256 characters). Key The value to translate.
`ToValue`	*Type:* text (max 256 characters). Nullable The required destination value for the Category/FromValue pair.

InstalledApplications Table

The `InstalledApplications` table is populated by the import process to track which software has been installed.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 767: Database columns for `InstalledApplications` table

Database Column	Details
`ComplianceComputerID`	*Type:* integer. Key The identifier for the computer in the `ComplianceComputer` table that the software is installed on.
`SoftwareTitleID`	*Type:* integer. Key The identifier for the software in the `SoftwareTitle` table that is installed.
`InstanceName`	*Type:* text (max 256 characters). Nullable The name of the instance that the software installation is associated with.
`InstallerEvidence`	*Type:* boolean This field is `True` if the installation is reported due to installer evidence.
`FileEvidence`	*Type:* boolean This field is `True` if the installation is reported due to file evidence.

Database Column	Details
WMIEvidence	*Type:* boolean This field is True if the installation is reported due to WMI evidence.
AccessModeID	*Type:* integer The access mode for which the installed application has been accessed. Foreign key to the AccessMode table.
IsACL	*Type:* boolean Determines whether the access mode record came from ACL data.

RelatedInstalledApplications Table

The RelatedInstalledApplications table is populated by the import process to track which relationship between applications.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 768: Database columns for RelatedInstalledApplications table

Database Column	Details
ParentComplianceComputerID	*Type:* integer. Key The parent identifier for the computer in the ComplianceComputer table that the software is installed on.
ParentSoftwareTitleID	*Type:* integer. Key The parent identifier for the software in the SoftwareTitle table that is installed.
ParentAccessModeID	*Type:* integer. Key The access mode for which the installed application has been accessed. Foreign key to the AccessMode table.
ChildComplianceComputerID	*Type:* integer. Key The child identifier for the computer in the ComplianceComputer table that the software is installed on.
ChildSoftwareTitleID	*Type:* integer. Key The child identifier for the software in the SoftwareTitle table that is installed.

Database Column	Details
`ChildAccessModeID`	*Type:* integer. Key The access mode for which the installed application has been accessed. Foreign key to the `AccessMode` table.
`IsCharged`	*Type:* boolean. Key The identifier used in the source connection to determine the pricing relation between parent and child installer evidence (specifies if it is charged or free).
`ConfidenceLevel`	*Type:* integer. Nullable Confidence level for each bundled installer evidence (as a percentage).

Compliance.InventoryWriter.Matching Tables

The complete set of database tables documented here includes:

- ClusterMatchResult table (see ClusterMatchResult Table)
- ComplianceComputerMatchResult table (see ComplianceComputerMatchResult Table)
- ImportedClusterMatchResult table (see ImportedClusterMatchResult Table)
- ImportedComputerMatchResult table (see ImportedComputerMatchResult Table)
- ImportedSoftwareBundleInstallerEvidence table (see ImportedSoftwareBundleInstallerEvidence Table)
- ImportedVirtualMachineMatchResult table (see ImportedVirtualMachineMatchResult Table)
- VirtualMachineMatchResult table (see VirtualMachineMatchResult Table)

ClusterMatchResult Table

The `ClusterMatchResult` table stores the results of performing matching between `ImportedClusters` and `Clusters`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 769: Database columns for `ClusterMatchResult` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the `ImportedCluster`.
`ExternalID`	*Type:* big integer. Key The identifier used in the source connection for the `ImportedCluster`.
`ClusterID`	*Type:* integer. Key Identifier of the computer in the `Cluster` table that this `ImportedCluster` links to.
`MatchingRule`	*Type:* text (max 128 characters) The matching rule which determined the match between this `ImportedCluster` and `Cluster`.

ComplianceComputerMatchResult Table

The `ComplianceComputerMatchResult` table stores the results of performing matching between `ImportedComputers` and `ComplianceComputers`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 770: Database columns for `ComplianceComputerMatchResult` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the `ImportedComputer`.
`ExternalID`	*Type:* big integer. Key The identifier used in the source connection for the `ImportedComputer`.
`ComplianceComputerID`	*Type:* integer. Key Identifier of the computer in the `ComplianceComputer` table that this `ImportedComputer` links to.
`MatchingRule`	*Type:* text (max 128 characters) The matching rule which determined the match between this `ImportedComputer` and `ComplianceComputer`.

ImportedClusterMatchResult Table

The `ImportedClusterMatchResult` table stores the results of performing matching between `ImportedClusters`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 771: Database columns for `ImportedClusterMatchResult` table

Database Column	Details
`PrimaryComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the primary `ImportedCluster`.
`PrimaryExternalID`	*Type:* big integer. Key The identifier used in the source connection for the primary `ImportedCluster`.
`MatchedComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table that supplied the matched `ImportedCluster`.
`MatchedExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the matched `ImportedCluster`.
`MatchingRule`	*Type:* text (max 128 characters) The matching rule which determined the match between these `ImportedClusters`.

ImportedComputerMatchResult Table

The `ImportedComputerMatchResult` table stores the results of performing matching between `ImportedComputers`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 772: Database columns for `ImportedComputerMatchResult` table

Database Column	Details
`PrimaryComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the primary `ImportedComputer`.
`PrimaryExternalID`	*Type:* big integer. Key The identifier used in the source connection for the primary `ImportedComputer`.
`MatchedComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table that supplied the matched `ImportedComputer`.
`MatchedExternalID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the matched `ImportedComputer`.
`MatchingRule`	*Type:* text (max 128 characters) The matching rule which determined the match between these `ImportedComputers`.

ImportedSoftwareBundleInstallerEvidence Table

The `ImportedSoftwareBundleInstallerEvidence` table holds software bundle to installer evidence information retrieved from the source connections.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 773: Database columns for `ImportedSoftwareBundleInstallerEvidence` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`BundleName`	*Type:* text (max 256 characters). Key. Nullable Bundle name
`BundlePublisher`	*Type:* text (max 64 characters). Key. Nullable Bundle's publisher

Database Column	Details
`EvidenceDisplayName`	*Type:* text (max 256 characters). Key. Nullable The version of the software as reported by the installer evidence.
`EvidenceVersion`	*Type:* text (max 72 characters). Key. Nullable Identifier for the type of installer evidence.
`EvidencePublisher`	*Type:* text (max 200 characters). Key. Nullable The publisher of the software as reported by the installer evidence.
`Supplementary`	*Type:* boolean Whether this installer evidence on this bundle is supplementary (counted for consumption) or not.
`MeasuredForCompliance`	*Type:* boolean Whether this installer evidence on this bundle is measured for compliance risks.
`ProductRatio`	*Type:* integer If this installer evidence is supplementary on the bundle, the number of entitlements consumed related to the entitlements consumed for the parent product.
`ParentProductRatio`	*Type:* integer If this installer evidence is supplementary on the bundle, the number of entitlements consumed related to the entitlements consumed for the supplementary product.
`DowngradeEnabled`	*Type:* boolean If this field is `True`, this bundle can cover previous releases, or lower editions, of applications linked to this license. If this field is `False` (the default), there is no downgrade right conferred by this license.
`DowngradeToVersion`	*Type:* boolean If this field is `True`, the bundle covers previous releases (with the same edition) of the primary application. If this field is `False` (the default), earlier versions of the primary application are not covered by downgrade rights.
`DowngradeToVersionID`	*Type:* integer. Nullable If the previous field is `True` and the value of this field is NULL, downgrade rights cover all earlier releases (with the same edition) of the primary application. If not NULL, downgrade rights cover all versions of the primary application down to and including this version. Foreign key to the `SoftwareTitleVersion` table.
`DowngradeToEdition`	*Type:* boolean If this field is `True`, the license covers lower editions (with the same version) of the primary application. If this field is `False` (the default), lower editions of the primary application are not covered by downgrade rights.

Database Column	Details
DowngradeToEditionID	*Type:* integer. Nullable If the previous field is True and the value of this field is NULL, downgrade rights cover all lower editions (with the same version) of the primary application. If not NULL, downgrade rights cover all editions of the primary application down to and including this edition. Foreign key to the SoftwareTitleEdition table.
UpgradeEnabled	*Type:* boolean If this field is True, the license can cover future releases (with the same edition) of the primary application. If this bit is False (the default), there is no upgrade right conferred by this license.
UpgradeToVersion	*Type:* boolean If this field is True, the license covers later releases (with the same edition) of the primary application. If this field is False (the default), later versions of the primary application are not covered by upgrade rights.
UpgradeToVersionID	*Type:* integer. Nullable If the previous field is True and the value of this field is NULL, upgrade rights cover all later version (with the same edition) of the primary application. If not NULL, upgrade rights cover all versions of the primary application up to and including this version. Foreign key to the SoftwareTitleEdition table.
UpgradeUntil	*Type:* boolean If this bit is 1, the upgrade right covers future releases of applications that get linked to this license, provided that the release date of each version is before (or on) a specified date. If this bit is zero (the default), the upgrade right is not date limited.
UpgradeUntilDate	*Type:* datetime. Nullable If this field is set, only applications released before this date are covered by upgrade rights.

ImportedVirtualMachineMatchResult Table

The ImportedVirtualMachineMatchResult table stores the results of performing matching between ImportedVirtualMachines.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 774: Database columns for `ImportedVirtualMachineMatchResult` table

Database Column	Details
`PrimaryComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the primary `ImportedVirtualMachine`.
`PrimaryVMComputerID`	*Type:* big integer. Key The identifier used in the source connection for the primary `ImportedVirtualMachine`.
`PrimaryHostComputerID`	*Type:* big integer. Key The identifier used in the source connection for the primary host `ImportedVirtualMachine`.
`MatchedComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the matched `ImportedVirtualMachine`.
`MatchedVMComputerID`	*Type:* big integer. Key The identifier used in the source connection for the matched `ImportedVirtualMachine`.
`MatchedHostComputerID`	*Type:* big integer. Key The identifier used in the source connection for the matched host `ImportedVirtualMachine`.
`MatchingRule`	*Type:* text (max 128 characters) The matching rule which determined the match between these `ImportedVirtualMachines`.
`NeedsCreation`	*Type:* boolean Whether this `ImportedVirtualMachine` is awaiting creation as a `VirtualMachine` or not.

VirtualMachineMatchResult Table

The `VirtualMachineMatchResult` table stores the results of performing matching between `ImportedVirtualMachines` and `VirtualMachines`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 775: Database columns for `VirtualMachineMatchResult` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key The identifier for a data source connection in the `ComplianceConnection` table that supplied the `ImportedVirtualMachine`.
`VMComputerID`	*Type:* big integer. Key The identifier used in the source connection for the `ImportedVirtualMachine`.
`HostComputerID`	*Type:* big integer. Key The identifier used in the source connection for the host of the `ImportedVirtualMachine`.
`VirtualMachineID`	*Type:* integer. Key Identifier of the virtual machine in the `VirtualMachine` table that this `ImportedVirtualMachine` links to.
`MatchingRule`	*Type:* text (max 128 characters) The matching rule which determined the match between these `VirtualMachines`.

Compliance.InventoryReader.CloudServiceData Tables

The complete set of database tables documented here includes:

- ImportedCloudServiceInstance table (see ImportedCloudServiceInstance Table)

ImportedCloudServiceInstance Table

The `ImportedCloudServiceInstance` table holds all of the data which have been retrieved from the cloud instance.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 776: Database columns for `ImportedCloudServiceInstance` table

Database Column	Details
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table.
`ExternalComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the computer.
`HostID`	*Type:* text (max 256 characters). Nullable The ID of the dedicated host instance.
`HostComputerID`	*Type:* big integer. Key. Nullable The identifier used in the source connection for the CloudServiceInstance's host computer.
`CloudServiceProvider`	*Type:* text (max 256 characters). Key. Nullable The cloud service provider for this instance. Foreign key to the `CloudServiceProvider` table.
`InstanceCloudID`	*Type:* text (max 256 characters). Key The ID of the cloud instance.
`InstanceAffinity`	*Type:* text (max 256 characters). Nullable The affinity setting for the instance on the Dedicated Host.
`ImageID`	*Type:* text (max 256 characters). Nullable The ID of the image used to launch the instance.
`LaunchTime`	*Type:* datetime. Nullable The time the cloud instance was launched or the Reserved Instance started.
`NetworkID`	*Type:* text (max 256 characters). Nullable The ID of the Virtual Private Cloud.
`MACAddress`	*Type:* text (max 256 characters). Key. Nullable The MAC address of the computer. This may be a comma-separated list if there is more than one active network adapter in the system. Do not include inactive network adapters and network adapters with invalid MAC addresses.
`LifecycleMode`	*Type:* text (max 256 characters). Nullable The time the instance was launched.
`ExpiryTime`	*Type:* datetime. Nullable The time when the Reserved Instance expires.

Database Column	Details
`InstanceCount`	*Type:* integer. Nullable The number of reservations purchased.
`OfferingClass`	*Type:* text (max 256 characters). Nullable The offering class of the Reserved Instance.
`OfferingType`	*Type:* text (max 256 characters). Nullable The Reserved Instance offering type.
`Scope`	*Type:* text (max 256 characters). Nullable The scope of the Reserved Instance.
`Account`	*Type:* text (max 256 characters). Nullable The Account that is used to create the instance.
`CoreCount`	*Type:* integer. Nullable The number of core of the instance.
`ThreadsPerCore`	*Type:* integer. Nullable The number of thread per core of the instance.
`InstanceType`	*Type:* text (max 256 characters). Nullable Cloud provider instance type.
`Region`	*Type:* text (max 256 characters). Nullable Region of the instance.
`AvailabilityZone`	*Type:* text (max 256 characters). Nullable Location of the instance.
`InstanceTenancy`	*Type:* text (max 256 characters). Nullable Instance tenancy of the instance
`VMEnabledStateID`	*Type:* integer. Nullable The state of the machine (powered on, off, etc).
`InventoryDate`	*Type:* datetime. Nullable The time of inventory date.

6

Inventory Database Schema

This chapter describes the schema for the FlexNet Manager Suite database that collects inventory uploaded by the FlexNet inventory agent, either when installed on 'adopted' devices, or when executing a remote, zero-touch inventory.

This inventory data undergoes some rationalization within this schema. The import of the resulting clean inventory data from this database to the compliance database is the work of the Compliance Reader, making use of another intermediate schema (see Compliance Reader Database Schema).

Information Structure

The following information is provided about database tables. Items appear only when relevant to the database column, and are suppressed where they do not apply. Two of these items (shown bold) are columns in the following pages, and the remainder are displayed within the **Details**.

Item	Comment
Database Column	The name of the column in the SQL table.
Type	The data type of the contents of the database column.
Size	For types that have a maximum capacity, the upper limit is provided in parentheses.
Key	The word “Key” appears when a column is a unique key field within the table. It is possible for several database columns to be part of the key, so that this indicator may appear for several columns in a table.
Generated ID	This indicates that a numeric ID is assigned by the database.
Nullable	If this indicator is present, the database column permits nulls.
Computed	This indicator appears for columns that are automatically computed by the database.
Default	If a column has a default value declared in the schema, this is specified at the end of the first set of details for the column.

Item	Comment
Details	Describes the data stored in the database column, including many of the indicators described above.

AD Tables

The complete set of database tables documented here includes:

- ADComputer table (see ADComputer Table)
- ADExternalMember table (see ADExternalMember Table)
- ADSDOU table (see ADSDOU Table)
- ADUser table (see ADUser Table)

ADComputer Table

The `ADComputer` table is populated with data from Active Directory in preparation for an Active Directory reconciliation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 777: Database columns for `ADComputer` table

Database Column	Details
`DomainID`	*Type:* integer. Key `OrganizationID` of the domain in which the computer resides.
`ComputerCN`	*Type:* text (max 64 characters). Key The computer's common name.
`ComputerOURDN`	*Type:* text (max 384 characters). Key The relative distinguished name of the organizational unit or container holding this computer.
`GUID`	*Type:* binary (max 16 bytes). Key The `objectGUID` of the Active Directory object that represents this computer, if known.
`SID`	*Type:* text (max 256 characters). Nullable The computer's SID.

ADExternalMember Table

The `ADExternalMember` table stores cross domain Active Directory objects.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 778: Database columns for `ADExternalMember` table

Database Column	Details
`GroupID`	*Type:* integer. Key The GroupID the external member belongs to.
`ExternalMemberSID`	*Type:* text (max 256 characters). Key The external member SID.

ADSDOU Table

The `ADSDOU` table is populated with domain, and organizational unit data from Active Directory in preparation for an Active Directory reconciliation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 779: Database columns for `ADSDOU` table

Database Column	Details
`DomainID`	*Type:* integer. Key The domain in which this object resides.
`RDN`	*Type:* text (max 400 characters). Key The relative distinguished name of this object.
`GUID`	*Type:* binary (max 16 bytes). Key The ObjectGUID of this Active Directory object.
`BlockInheritance`	*Type:* boolean True (1) if package allocations should not be inherited from parent OUs or `Domain`, unless no-override is set for the Allocation (in the policy group membership mode).

ADUser Table

The `ADUser` table contains is populated with data from Active Directory in preparation for an Active Directory reconciliation. It is a temporary table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 780: Database columns for `ADUser` table

Database Column	Details
`DomainID`	*Type:* integer. Key The domain in which this user resides.
`UserCN`	*Type:* text (max 64 characters). Key The user's common name.
`UserOURDN`	*Type:* text (max 384 characters). Key The relative distinguished name of the organizational unit or container holding this user.
`GUID`	*Type:* binary (max 16 bytes). Key. Nullable The Active Directory GUID of this user.
`SAMAccountName`	*Type:* text (max 20 characters). Nullable The user's logon name used to support clients and servers from versions of Windows prior to Windows 2000.
`Sid`	*Type:* text (max 512 characters). Nullable User's Sid

ClientAccess Tables

The complete set of database tables documented here includes:

- ClientAccessDetail table (see ClientAccessDetail Table)
- ClientAccessOccurrence table (see ClientAccessOccurrence Table)
- ClientAccessingDevice table (see ClientAccessingDevice Table)
- ClientAccessingUser table (see ClientAccessingUser Table)
- UALSoftwareDetail table (see UALSoftwareDetail Table)

ClientAccessDetail Table

Records the entries for the client accesses obtained from the User access logging.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 781: Database columns for `ClientAccessDetail` table

Database Column	Details
`ClientAccessDetailID`	*Type:* big integer. Key. Generated ID Auto-generated identity number.
`ClientAccessingDeviceID`	*Type:* integer. Key. Nullable The client access device related record. This is a foreign key into the `ClientAccessingDevice` table.
`ClientAccessingUserID`	*Type:* integer. Key. Nullable The client access user related record. This is a foreign key into the `ClientAccessingUser` table.
`ServerComputerID`	*Type:* integer. Key The record of the server from which the inventory is obtained. This is a foreign key into the `Computer` table.
`UALSoftwareDetailID`	*Type:* integer. Key The client access software related record. This is a foreign key into the `UALSoftwareDetail` table.

ClientAccessOccurrence Table

Records the entries for the software access occurrence in the User access logging.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 782: Database columns for `ClientAccessOccurrence` table

Database Column	Details
`ClientAccessDetailID`	*Type:* big integer. Key Client access related record. This is a foreign key into the `ClientAccessingDetails` table.

Database Column	Details
`AccessDate`	*Type:* datetime. Nullable Date and time at which access was made to server.
`InventoryDate`	*Type:* datetime. Key Date and time at which this inventory occurrence was recorded.
`LicenseDate`	*Type:* datetime. Key Date used for licensing purposes.
`AccessCount`	*Type:* integer Number of times access was made to server.

ClientAccessingDevice Table

Records the entries for the client accessing devices obtained from the User access logging.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 783: Database columns for `ClientAccessingDevice` table

Database Column	Details
`ClientAccessingDeviceID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`IPAddress`	*Type:* text (max 256 characters). Key. Nullable IP Address of the client accessing device.
`ComputerName`	*Type:* text (max 256 characters). Key. Nullable Computer name of the client accessing device.

ClientAccessingUser Table

Records the entries for the client accessing users obtained from the User access logging.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 784: Database columns for `ClientAccessingUser` table

Database Column	Details
`ClientAccessingUserID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`UserName`	*Type:* text (max 256 characters). Key User name of the accessing user.
`DomainName`	*Type:* text (max 100 characters). Key. Nullable Domain name of the accessing user.

UALSoftwareDetail Table

Records the entries for the softwares registered in the User access logging.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 785: Database columns for `UALSoftwareDetail` table

Database Column	Details
`UALSoftwareDetailID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`ProductName`	*Type:* text (max 256 characters). Key The name of the instalaltion product. This may include version and edition too.
`RoleName`	*Type:* text (max 256 characters). Key. Nullable The URL role name. This is used when retrive data using UAL.
`RoleGUID`	*Type:* unique identifier. Key. Nullable The URL role GUID. This is used when retrive data using UAL
`ClientAccessSource`	*Type:* text (max 100 characters). Key Referencing to the client access source type.

DirectoryObjects Tables

The complete set of database tables documented here includes:

- Computer table (see Computer Table)

- OperatingSystem table (see OperatingSystem Table)
- User table (see User Table)

Computer Table

The `Computer` table contains all computers that have ever reported information or have been targeted by policy in a FlexNet Manager Suite environment.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 786: Database columns for `Computer` table

Database Column	Details
`ComputerID`	*Type:* integer. Key. Generated ID The ID for the computer. This is automatically generated by SQL Server.
`AgentID`	*Type:* text (max 256 characters). Key. Nullable The unique ID for the agent on the computer.
`ComputerOUID`	*Type:* integer. Key The organizational unit of the computer in Active Directory. In an SMS organization, this is set to the OUID of the unknown OU.
`ComputerCN`	*Type:* text (max 256 characters). Key The computer's common name. In an Active Directory environment this is the common name attribute of the computer's distinguished name. This is the same as the SAM account name.
`ComputerUID`	*Type:* unique identifier. Key A unique external identifier for the computer.
`OperatingSystemID`	*Type:* integer. Nullable The operating system of the computer, if known. This allows efficient determination of the operating system breakdown of computers in an organization.
`GUID`	*Type:* binary (max 16 bytes). Key. Nullable The `objectGUID` of the Active Directory object that represents this computer, if known.

OperatingSystem Table

This table stores the information about different types of OS available on the network devices

Table 787: Database columns for `OperatingSystem` table

Database Column	Details
`OperatingSystemID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`OperatingSystemName`	*Type:* text (max 128 characters). Key Name of operating system
`Category`	*Type:* integer. Nullable Reference to operating system category

User Table

The `User` table contains all of the users that have ever reported information in a FlexNet Manager Suite environment. A row has been added to cater for installations that occur from computer policy. This row has a `UserID` set to 1 and a `UserCN` of `SYSTEM`, and belongs to `Organization` 1, which is the universal `Domain`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 788: Database columns for `User` table

Database Column	Details
`UserID`	*Type:* integer. Key. Generated ID The ID for the user. This is automatically generated by SQL Server.
`UserOUID`	*Type:* integer. Key The organizational unit of the user in Active Directory. This is a foreign key into the `Organization` table. In an SMS environment, this is always set to the unknown OU.
`UserCN`	*Type:* text (max 64 characters). Key The user's common name. In an Active Directory environment this is the common name attribute of the user's distinguished name.
`GUID`	*Type:* binary (max 16 bytes). Key. Nullable The `objectGUID` of the Active Directory object that represents this user, if known.
`SAMAccountName`	*Type:* text (max 20 characters). Key. Nullable The SAM account name used to uniquely identify this user in event logs and user inventories.

Directory Tables

The complete set of database tables documented here includes:

- Domain table (see Domain Table)
- DomainConfiguration table (see DomainConfiguration Table)
- Group table (see Group Table)
- Member table (see Member Table)
- Organization table (see Organization Table)

Domain Table

The `Domain` table, in combination with the `Organization` table, contains data about all of the domains, and organizational units that have ever had users or computers report information in a FlexNet Manager Suite environment.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 789: Database columns for `Domain` table

Database Column	Details
`OrganizationID`	*Type:* integer. Key Organizational ID. This is a foreign key into the `Organization` table.
`DN`	*Type:* text (max 100 characters). Key. Nullable Fully qualified distinguished name.
`DomainType`	*Type:* text (max 4 characters). Key. Nullable The type of directory service running, for example AD, NT 4.
`FlatName`	*Type:* text (max 32 characters). Nullable The NT 4 domain name.
`PreferredDomainController`	*Type:* text (max 32 characters). Nullable Preferred domain controller to query.
`PreferredDomain ControllerOnly`	*Type:* boolean Whether (0) or not to fail over to alternate server if the preferred domain controller is not contactable.

Database Column	Details
`ADReconcile`	*Type:* boolean Whether (1) or not (0) to reconcile the FlexNet Manager Suite database with Active Directory.
`ADLoadLatency`	*Type:* integer If reconciling Active Directory with the FlexNet Manager Suite database, the length of time in minutes before the Active Directory data is refreshed in the FlexNet Manager Suite database. The default value is 60 minutes. A value of 0 means load the Active Directory data into the FlexNet Manager Suite database at each reconciliation. Set this to a high value to minimize network traffic for domains for delayed reconciliation is acceptable.
`MergePolicies`	*Type:* boolean Whether (1) or not (0) to generate merged policies.
`LastADReconcile`	*Type:* datetime. Nullable The date and time of the last reconciliation of the FlexNet Manager Suite database with Active Directory.
`LastADReconcileStatus`	*Type:* boolean This field is currently unused.
`LastADLoad`	*Type:* datetime. Nullable The date and time of the last Active Directory load. A value of NULL indicates that Active Directory data should be loaded at the next reconcile operation.
`LastPolicyMerge`	*Type:* datetime. Nullable The date and time of the last generation of merged policy.
`LastPolicyMergeStatus`	*Type:* boolean This field is currently unused.
`DNReverse`	*Type:* text (max 100 characters). Key. Nullable Fully qualified distinguished name, in reverse order (to improve sub-domain search performance).

DomainConfiguration Table

The `DomainConfiguration` table contains configuration properties for the `Domain` table

Table 790: Database columns for `DomainConfiguration` table

Database Column	Details
`DomainID`	*Type:* integer. Key `OrganizationID` of the domain in which the entry resides.
`Property`	*Type:* text (max 32 characters). Key The name of the property.
`Value`	*Type:* text (max 256 characters). Nullable The value of the property.
`DateValue`	*Type:* datetime. Nullable The date and time value of the property.
`LastUpdate`	*Type:* datetime The date and time the property was last updated.

Group Table

Each `Group` identifies either a named group or an unnamed Access Control List (ACL). Each `Group` is associated with rows in the `Member` table.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 791: Database columns for `Group` table

Database Column	Details
`GroupID`	*Type:* integer. Key. Generated ID The ID for the group, automatically generated by SQL Server.
`GUID`	*Type:* binary (max 16 bytes). Key The Globally Unique IDentifier for the group. In the case where this `Group` represents an Access Control List for a `Policy` or a `PackageAllocation`, the GUID is that of this object.
`GroupCN`	*Type:* text (max 128 characters). Key. Nullable The Common Name for the group. In the case where this `Group` represents an Access Control List for a `Policy` or a `PackageAllocation`, the GroupCN is NULL.

Database Column	Details
GroupOUID	*Type:* integer. Key A reference to the Organization to which the group belongs.
GroupType	*Type:* integer. Nullable The bitmask of flags defining the type of this Group.
SID	*Type:* text (max 256 characters). Nullable The security identifier of this Group.

Member Table

The Member table stores the membership lists for every group. Each Member details a User, Computer, Group, or Organization (only Policy ACL groups), and whether the specified item is excluded (only ACL groups), included (the default) or included mandatorily (cannot be excluded - used only for Organizations in Policy ACLs).

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 792: Database columns for Member table

Database Column	Details
GroupID	*Type:* integer. Key The Group of which this is a Member.
TargetTypeID	*Type:* integer. Key The TargetType. Possible values are: • 1 = Computer • 2 = User • 3 = Group • 8 = OrgUnit • 16 = Operator
TargetID	*Type:* integer. Key The ComputerID, UserID, GroupID or OrganizationID.
MemberMode	*Type:* integer The MemberMode is 0 for Exclude (regardless of any other memberships, the principals of this Target are excluded from this group), 1 for Include, and 2 for Always - NoOverride.

Organization Table

The `Organization` table contains data about organizational units used in a FlexNet Manager Suite environment.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 793: Database columns for `Organization` table

Database Column	Details
`OrganizationID`	*Type:* integer. Key. Generated ID The ID for the organizational unit. (1 is used for “unknown OU” in the universal domain).This is automatically generated by SQL Server.
`RDN`	*Type:* text (max 400 characters). Key. Nullable The relative distinguished name of this organizational unit.
`GUID`	*Type:* binary (max 16 bytes). Key. Nullable The `objectGUID` of the Active Directory object that represents this organizational unit, if known.
`DomainID`	*Type:* integer. Key. Nullable `OrganizationID` of the domain in which the entry resides. For a domain, must be set to reference self.
`RDNReverse`	*Type:* text (max 400 characters). Key. Nullable The relative distinguished name of the computer, reversed for superior performance on sub-organization searching.
`IsUnknown`	*Type:* integer True (1) if the organizational unit cannot be resolved through Active Directory (for example, the unknown OU, which has a NULL RDN), false (0) otherwise (if the OU has a non-empty RDN).
`IsDomain`	*Type:* integer True (1) if the organizational unit is a domain (has an empty - not NULL - RDN), false (0) otherwise.
`BlockInheritance`	*Type:* boolean True (1) if package allocations should not be inherited from the parent `Organization`, unless `NoOverride` is set for the `Policy`. `NoOverride` is set using `MemberMode=2 (Always)` on the `Organization` member in the `Policy` ACL group.

Distribution Tables

The complete set of database tables documented here includes:

- DistributedPackage table (see DistributedPackage Table)
- DistributionGroup table (see DistributionGroup Table)
- DistributionGroupMember table (see DistributionGroupMember Table)
- DistributionServer table (see DistributionServer Table)
- DistributionServerStatus table (see DistributionServerStatus Table)
- DistributionServerType table (see DistributionServerType Table)

DistributedPackage Table

The `DistributedPackage` table stores the status (both of current and pending distributions) of package distributions to distribution servers and distribution locations.

Table 794: Database columns for `DistributedPackage` table

Database Column	Details
`DistributedPackageID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`ServerUID`	*Type:* binary (max 16 bytes). Key The distribution server or distribution location related to the status record. This is a foreign key into the `DistributionServer` table.
`RequestedVersionID`	*Type:* integer. Key. Nullable The id for the Requested `PackageVersion`.
`RequestState`	*Type:* text (max 16 characters). Nullable The state of a package that is pending distribution. The possible values are: • empty (literal string) • pending • removing If the `RequestState` field contains a value other than the literal string `empty`, the `RequestState` overrides the `ConfirmedState` of the package.
`RequestDate`	*Type:* datetime. Nullable The date and time at which the package distribution began. Only used for distributions currently in progress.

Database Column	Details
`ConfirmedVersionID`	*Type:* integer. Key. Nullable The id for the Existing `PackageVersion`
`ConfirmedState`	*Type:* text (max 16 characters). Nullable The state of the package currently on the distribution server or distribution location. The possible values are: • available • unavailable If the `RequestState` field contains a value other than the literal string `empty`, then the `RequestState` overrides the `ConfirmedState` of the package.
`ConfirmedDate`	*Type:* datetime. Nullable The date and time that the current distribution status of a package was recorded.
`ConfirmedReason`	*Type:* text. Nullable The reason that package distribution failed. This is only specified in the case of a failure.

DistributionGroup Table

All defined distribution groups are stored in the `DistributionGroup` table.

Table 795: Database columns for `DistributionGroup` table

Database Column	Details
`GroupUID`	*Type:* binary (max 16 bytes). Key A unique identifier for this distribution group.
`GroupName`	*Type:* text (max 128 characters). Key The descriptive name assigned to this distribution group.

DistributionGroupMember Table

Any distribution servers and distribution locations assigned to distribution groups are stored in the `DistributionGroupMember` table.

Table 796: Database columns for `DistributionGroupMember` table

Database Column	Details
`GroupUID`	*Type:* binary (max 16 bytes). Key A unique identifier for this distribution group. This UID is a foreign key to the `GroupUID` in the `DistributionGroup` table.
`MemberID`	*Type:* binary (max 16 bytes). Key A unique identifier for the distribution server or distribution location that is a member of this group. This UID is a foreign key to the `ServerUID` in the `DistributionServer` table.
`MemberType`	*Type:* integer An identifier for the type of this distribution group member. This identifier is a foreign key to the `TargetTypeID` in the `DistributionServerType` table.

DistributionServer Table

The `DistributionServer` table stores all of the distribution servers and distribution locations in the FlexNet Manager Suite distribution hierarchy.

Table 797: Database columns for `DistributionServer` table

Database Column	Details
`ServerUID`	*Type:* binary (max 16 bytes). Key A unique identifier for the distribution server or distribution location. The core distribution server has a value of all zeroes.
`DNSName`	*Type:* text (max 128 characters). Nullable DNS name of the server
`ServerType`	*Type:* small integer The server type. The possible values are: • 0 for distribution location • 1 for distribution server
`ServerName`	*Type:* text (max 64 characters) The name of the distribution server or distribution location.
`PrimaryParentUID`	*Type:* binary (max 16 bytes). Key. Nullable The parent of the distribution server or distribution location. For the core distribution server, the `PrimaryParentUID` is NULL.

Database Column	Details
`ConfigState`	*Type:* text (max 20 characters). Nullable The state of configuration of the distribution server. This is only set for distribution servers (`ServerType` is 1).This can be one of the following values: • configure • failed • pending • NULL
`LastConfigStart`	*Type:* datetime. Nullable The date and time of the last configuration message sent to the distribution server. This is only set for distribution servers (`ServerType` is 1).
`LastConfigJobId`	*Type:* text (max 40 characters). Nullable The job identifier for the last configuration message sent to the distribution server. This is only set for distribution servers (`ServerType` is 1).
`ConfigFailReason`	*Type:* text. Nullable The reason for a configuration failure for the distribution server.
`PolicyQuarantined`	*Type:* boolean Boolean value indicating whether this distribution location is quarantined from receiving policy distributions.
`TenantID`	*Type:* small integer The Tenant ID this Distribution Server has been assigned to.

DistributionServerStatus Table

The `DistributionServerStatus` table stores status information for the distribution servers in the FlexNet Manager Suite distribution hierarchy.

Table 798: Database columns for `DistributionServerStatus` table

Database Column	Details
`ServerUID`	*Type:* binary (max 16 bytes). Key The distribution server related to the status record. This is a foreign key into the `DistributionServer` table.
`ReportedDate`	*Type:* datetime The date and time at which the distribution server last reported status information for this parameter.

Database Column	Details
`Type`	*Type:* text (max 32 characters). Key The type of the status parameter reported. Currently supported types are 'job' for jobs on the distribution server and 'logs' for log files awaiting upload from the distribution server.
`Name`	*Type:* text (max 64 characters). Key The name of the status parameter reported. This is an internal name for the parameter and is not intended for display.
`Count`	*Type:* integer The count of items for this status parameter currently awaiting processing by this distribution server.
`DelayedCount`	*Type:* integer The count of items for this status parameter that are older than a configurable time period that are currently awaiting processing by this distribution server. This will not necessarily have meaning for each status parameter.

DistributionServerType Table

The available distribution server types are defined in the `DistributionServerType` table.

Table 799: Database columns for `DistributionServerType` table

Database Column	Details
`DistributionServerTypeID`	*Type:* integer. Key An identifier for this distribution server type.
`DistributionServerType Name`	*Type:* text (max 256 characters). Key The descriptive name assigned to this distribution server type.

Installation Tables

The complete set of database tables documented here includes:

- Installation table (see Installation Table)
- InstallationHistory table (see InstallationHistory Table)
- Reason table (see Reason Table)

Installation Table

The `Installation` table contains the latest installation status of each package for each user and computer. Success or failure of installations is recorded. When a package is uninstalled, its installation status record is removed from the table. When an installation is successful, the successful installation status record replaces any earlier failure status records. Once an installation is successful, the `Installation` table retains the successful installation status record even if there are subsequent failed installation attempts. If there have been any subsequent failed installations, the latest of these failure records is also be retained. For example, if an installation fails, and then succeeds on a subsequent attempt, only the successful status is recorded. If an installation succeeds, but a later installation attempt fails, then both the success and failure status records are stored. All other installations are added to the `InstallationHistory` table as new status information is generated.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 800: Database columns for `Installation` table

Database Column	Details
`ComputerID`	*Type:* integer. Key The computer that the installation event occurred on. This is a foreign key into the `Computer` table.
`UserID`	*Type:* integer. Key The user associated with the installation event. This is a foreign key into the `User` table. If the `UserID` is 1 (system user), then the installation event occurred as part of computer policy. Otherwise, the installation event occurred as part of user policy.
`PackageVersionID`	*Type:* integer. Key Package version that was installed
`OrganizationID`	*Type:* integer. Key The organizational unit of the user or computer associated with the installation event. This is a foreign key into the `Organization` table. This column is included in the table for clustering purposes. The value of OUID could be determined by looking up the `User` or `Computer` table. If the `UserID` is 1 (system user), OUID represents the organizational unit of the computer that the installation event occurred on. Otherwise, it represents the organizational unit of the user associated with the installation event.
`Action`	*Type:* text (max 10 characters) The action performed on the package. This is currently set to “install”. In future, “upgrade”, “update” and “selfheal” may be added.

Database Column	Details
`Reported`	*Type:* datetime. Nullable The date and time that the installation event occurred.
`Received`	*Type:* datetime. Nullable The date and time that the installation status event was received into the database.
`FailReasonID`	*Type:* integer. Nullable A reference to the reason for the installation failure. If the installation succeeded then this value is NULL.
`Result`	*Type:* text (max 16 characters). Nullable The result of the package installation. Possible values are success or failure.

InstallationHistory Table

The action performed on the package, normally “install” or “uninstall”. In the event that installation event logs were lost, entries may be reconstructed here from data in inventory (cache tracking). Such entries may be less reliable (in particular the recorded date will be the date of the inventory) and will have one of the following `Action` values:

- “inv insert”
- “inv delete”
- “inv update”

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 801: Database columns for `InstallationHistory` table

Database Column	Details
`ComputerID`	*Type:* integer. Key The computer that the installation event occurred on. This is a foreign key into the `Computer` table.
`UserID`	*Type:* integer. Key The user associated with the installation event. This is a foreign key into the `User` table. If the `UserID` is 1 then the installation event occurred as part of computer policy. Otherwise, the installation event occurred as part of user policy.
`PackageVersionID`	*Type:* integer. Key The id for the `PackageVersion` installed

Database Column	Details
`Reported`	*Type:* datetime. Key The date and time that the installation event occurred.
`Action`	*Type:* text (max 10 characters) The action performed on the package. This value can be either, install or uninstall.
`Received`	*Type:* datetime. Nullable The date and time that the installation status event was received into the database.
`FailReasonID`	*Type:* integer. Nullable A reference to the reason for the installation failure. If the installation succeeded then this value is NULL.
`Result`	*Type:* text (max 16 characters). Nullable The result of the package installation. Possible values are either success or failure.

Reason Table

Stores extended text uploaded from logs to describe operational failures.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 802: Database columns for `Reason` table

Database Column	Details
`ReasonID`	*Type:* integer. Key. Generated ID The ID for the Reason. This is automatically generated by SQL Server.
`ReasonHash`	*Type:* integer. Key The checksum of the ReasonText, calculated by SQL Server.
`ReasonText`	*Type:* text The Reason text.

Inventory Tables

The complete set of database tables documented here includes:

- ComputerResourceData table (see ComputerResourceData Table)

- ComputerResourceHierarchySCD table (see ComputerResourceHierarchySCD Table)
- ComputerResourceProcessorSCD table (see ComputerResourceProcessorSCD Table)
- ComputerResourceType table (see ComputerResourceType Table)
- ComputerResourceVMPoolType table (see ComputerResourceVMPoolType Table)
- ComputerResourceVMType table (see ComputerResourceVMType Table)
- ComputerResourceVirtualMachine table (see ComputerResourceVirtualMachine Table)
- HardwareClass table (see HardwareClass Table)
- HardwareObject table (see HardwareObject Table)
- HardwareProperty table (see HardwareProperty Table)
- HardwareValue table (see HardwareValue Table)
- InventoryReport table (see InventoryReport Table)
- ServiceComponent table (see ServiceComponent Table)
- ServiceProvider table (see ServiceProvider Table)
- ServiceProviderApplicationOracle table (see ServiceProviderApplicationOracle Table)
- ServiceProviderApplicationUsagePerMonth table (see ServiceProviderApplicationUsagePerMonth Table)
- ServiceProviderApplicationUsageType table (see ServiceProviderApplicationUsageType Table)
- ServiceProviderApplicationUserOracle table (see ServiceProviderApplicationUserOracle Table)
- ServiceProviderComponent table (see ServiceProviderComponent Table)
- ServiceProviderComponentProperty table (see ServiceProviderComponentProperty Table)
- ServiceProviderComponentValue table (see ServiceProviderComponentValue Table)
- ServiceProviderName table (see ServiceProviderName Table)
- ServiceProviderProperty table (see ServiceProviderProperty Table)
- ServiceProviderType table (see ServiceProviderType Table)
- ServiceProviderValue table (see ServiceProviderValue Table)
- ServiceUser table (see ServiceUser Table)
- ServiceUserOracle table (see ServiceUserOracle Table)
- SoftwareDetails table (see SoftwareDetails Table)
- SoftwareFile table (see SoftwareFile Table)
- SoftwareFileName table (see SoftwareFileName Table)
- SoftwareFilePath table (see SoftwareFilePath Table)

- SoftwareIsoTagEntity table (see SoftwareIsoTagEntity Table)
- SoftwareIsoTagFile table (see SoftwareIsoTagFile Table)
- SoftwareIsoTagSoftwareVersion table (see SoftwareIsoTagSoftwareVersion Table)
- SoftwareIsoTagUnique table (see SoftwareIsoTagUnique Table)
- SoftwareOccurrence table (see SoftwareOccurrence Table)
- SoftwareOccurrenceSoftwareIsoTagFile table (see SoftwareOccurrenceSoftwareIsoTagFile Table)
- SoftwareProperty table (see SoftwareProperty Table)
- SoftwareValue table (see SoftwareValue Table)
- SoftwareVersion table (see SoftwareVersion Table)
- VirtualDesktopAccess table (see VirtualDesktopAccess Table)
- VirtualDesktopApplicationUsage table (see VirtualDesktopApplicationUsage Table)
- VirtualDesktopGroupAccess table (see VirtualDesktopGroupAccess Table)
- VirtualDesktopGroupAccessScan table (see VirtualDesktopGroupAccessScan Table)

ComputerResourceData Table

`ComputerResourceData` stores information about computer resources used in the enterprise.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 803: Database columns for `ComputerResourceData` table

Database Column	Details
`ComputerResourceID`	*Type:* integer. Key. Generated ID A unique identifier for a `ComputerResourceData`.
`ComputerResourceTypeID`	*Type:* integer. Key The type of resource. Foreign key to the `ComputerResourceType` table.
`ComputerUID`	*Type:* unique identifier. Key. Nullable The computer resource's UUID, in the byte order reported in inventory. Foreign key to the `Computer` table.
`ComputerResourceVMPool TypeID`	*Type:* integer. Nullable If this resource is a resource pool, this specifies the type of pool. Foreign key to the `ComputerResourceVMPoolType` table.

Database Column	Details
`ComputerResourceVMTypeID`	*Type:* integer. Key. Nullable If this resource is a virtual machine, this specifies the type of virtual machine or partition. Foreign key to the `ComputerResourceVMType` table.
`NormalizedSerialNo`	*Type:* text (max 100 characters). Key. Nullable The serial number of the resource in a normalized format.
`Name`	*Type:* text (max 256 characters). Nullable The name of the resource.
`Manufacturer`	*Type:* text (max 128 characters). Key. Nullable The manufacturer of the resource.
`ModelNo`	*Type:* text (max 128 characters). Key. Nullable The model number of the resource.
`SerialNo`	*Type:* text (max 100 characters). Key. Nullable The serial number of the resource.
`IsFabricatedHost`	*Type:* boolean Is the host generated from the virtual machine inventory.
`LastUpdated`	*Type:* datetime. Nullable The last time this computer resource was updated.

ComputerResourceHierarchySCD Table

`ComputerResourceHierarchySCD` is a table defining relationships between computer resources

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 804: Database columns for `ComputerResourceHierarchySCD` table

Database Column	Details
`ComputerResourceID`	*Type:* integer. Key The guest resource in the relationship. Foreign key to the `ComputerResourceData` table.

Database Column	Details
HostComputerResourceID	*Type:* integer. Key The host resource in the relationship. For an unhosted computer and for a host computer this value will be identical to the ComputerResourceID value. Foreign key to the ComputerResourceData table.
ParentComputerResourceID	*Type:* integer. Key. Nullable The direct parent of the guest resource (which might be, for example, a resource pool). Is NULL when the guest resource has no parents within the hierarchy. Foreign key to the ComputerResourceData table.
ValidFrom	*Type:* datetime. Key Date from which this relationship record became valid.
ValidTo	*Type:* datetime. Key Date this hierarchy was valid to, or 9999-12-31T23:59:59.997 if it is currently valid. (This string is used to represent an indefinite future.)

ComputerResourceProcessorSCD Table

ComputerResourceProcessorSCD is a table listing processor specifications for a particular computer resource.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 805: Database columns for ComputerResourceProcessorSCD table

Database Column	Details
ComputerResourceID	*Type:* integer. Key The resource to which the processor specification applies. Foreign key to the ComputerResourceData table.
NumberOfProcessors	*Type:* decimal. Nullable The number of processors in the resource.
NumberOfCores	*Type:* decimal. Nullable The number of cores in the resource.
NumberOfLogicalProcessors	*Type:* integer. Nullable The number of logical processors in the resource. A logical processor is a processor thread.

Database Column	Details
`PartialNumberOfProcessors`	*Type:* decimal. Nullable The fractional processor count available to this computer.
`ProcessorType`	*Type:* text (max 256 characters). Nullable The type of processor in the resource.
`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed (in megahertz) of the fastest processor in the resource.
`ValidFrom`	*Type:* datetime. Key Date from which these properties became valid.
`ValidTo`	*Type:* datetime. Key Date these properties were valid to, or 9999-12-31T23:59:59.997 if they are currently valid.

ComputerResourceType Table

`ComputerResourceType` is a static table listing all possible computer resource types.

Table 806: Database columns for `ComputerResourceType` table

Database Column	Details
`ComputerResourceTypeID`	*Type:* integer. Key. Generated ID A unique identifier for each `ComputerResourceType`. Possible values and the corresponding default strings are: • 1 = Host • 2 = Resource pool • 3 = Virtual machine • 4 = Physical machine that is not a virtual host of any kind.
`Name`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a resource type.
`DefaultValue`	*Type:* text (max 128 characters) The text to display if the resource type resource string has no translation.

ComputerResourceVMPoolType Table

`VMPoolType` is a static table listing the possible types of a virtual machine pool.

Table 807: Database columns for `ComputerResourceVMPoolType` table

Database Column	Details
`ComputerResourceVMPoolTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a `VMPoolType`. Possible values and the corresponding default names are: • 1 = Folder • 2 = Data Center • 3 = Compute Resource • 4 = Host System • 5 = Resource Pool • 6 = Virtual Machine • 7 = Physical Shared Pool • 8 = Virtual Shared Pool • 9 = LPAR • 10 = RSET • 11 = Cluster Compute Resource. • 12 = PSET
`VCTypeID`	*Type:* text (max 32 characters) The type of the virtual machine folder in VMware Virtual Center.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a pool type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the pool type resource string has no translation.

ComputerResourceVMType Table

`VMType` is a static table listing the possible types of virtual machine or partition.

Table 808: Database columns for `ComputerResourceVMType` table

Database Column	Details
`ComputerResourceVMTypeID`	*Type:* integer. Key. Generated ID A unique identifier for a `VMType`. Possible values and the corresponding default names are: • 1 = VMware • 2 = Hyper-V • 3 = LPAR • 4 = WPAR • 5 = nPar • 6 = vPar • 7 = SRP • 8 = Zone • 9 = Unknown • 10 = Oracle VM • 11 = AWS EC2 • 12 = Linux KVM.
`ResourceName`	*Type:* text (max 256 characters). Key The unique name of the localizable resource string representing a virtual machine or partition type. Foreign key to the `ComplianceResourceString` table.
`DefaultValue`	*Type:* text (max 100 characters) The text to display if the type resource string has no translation.

ComputerResourceVirtualMachine Table

`ComputerResourceVirtualMachine` is a table containing the type and normalized UUID of virtual machines and the host they are currently known to be on. The normalized UUID is the virtual machine UUID with hyphen and white space characters removed.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 809: Database columns for `ComputerResourceVirtualMachine` table

Database Column	Details
`ComputerResourceID`	*Type:* integer. Key The host the virtual machine is currently known to be on. Foreign key to the `ComputerResourceData` table.
`ComputerResourceVMTypeID`	*Type:* integer. Key Type of virtual machine or partition. Foreign key to the `ComputerResourceVMType` table.
`NormalizedUUID`	*Type:* text (max 100 characters). Key The normalized UUID of the virtual machine.

HardwareClass Table

`HardwareClass` contains a record for every class of hardware object found during hardware inventories, including mainly the WMI classes

Table 810: Database columns for `HardwareClass` table

Database Column	Details
`HardwareClassID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`Class`	*Type:* text (max 256 characters). Key Hardware Class name
`SuperClassID`	*Type:* integer. Key. Nullable Reference to superclass, if any (and known)

HardwareObject Table

The `HardwareObject` table entries describe a specific configuration item (usually a piece of physical hardware) associated with a computer. The information is represented in the database as Windows Management Instrumentation (WMI) classes.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 811: Database columns for `HardwareObject` table

Database Column	Details
`HardwareObjectID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`ComputerID`	*Type:* integer. Key The computer on which the hardware was found. It is a foreign key into the `Computer` table.
`HardwareName`	*Type:* text (max 256 characters). Key The hardware name as reported by the system.
`Occurrence`	*Type:* integer. Key The distinguishing identifier for the hardware. For example, if a computer has more than one memory card with the same Class and `HardwareName`, each memory card is assigned an `Occurrence` value (0, 1, 2…).
`HardwareClassID`	*Type:* integer. Key The id for the `HardwareClass` of the object.

HardwareProperty Table

The `HardwareProperty` table provides property names and values for each hardware object. The information is represented in the database as Windows Management Instrumentation (WMI) properties.

Table 812: Database columns for `HardwareProperty` table

Database Column	Details
`HardwarePropertyID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`Property`	*Type:* text (max 256 characters). Key The hardware property. A single hardware object can have many properties.

HardwareValue Table

The value of a specified `HardwareProperty` of the specified `HardwareObject`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 813: Database columns for `HardwareValue` table

Database Column	Details
`HardwareObjectID`	*Type:* integer. Key Object.
`HardwarePropertyID`	*Type:* integer. Key Property.
`Value`	*Type:* text. Nullable Property value.

InventoryReport Table

The `InventoryReport` table contains a record of every user and computer that has reported hardware or software inventory. It details the date and time when the hardware or software tracking was performed.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 814: Database columns for `InventoryReport` table

Database Column	Details
`ComputerID`	*Type:* integer. Key The computer that the inventory was tracked on. This is a foreign key into the `Computer` table.
`UserID`	*Type:* integer. Key The user for whom inventory was tracked. For computer inventory, the `UserID` is 1 (system user). This is a foreign key into the `User` table.
`SWDate`	*Type:* datetime. Nullable The time software was tracked, or is NULL if no tracking is recorded.
`HWDate`	*Type:* datetime. Nullable The time hardware was tracked, or is NULL if no tracking is recorded.
`FilesDate`	*Type:* datetime. Nullable The time files were tracked, or is NULL if no tracking is recorded.
`ServicesDate`	*Type:* datetime. Nullable The time Oracle services were tracked, or is NULL if no tracking is recorded.

Database Column	Details
`VMwareServicesDate`	*Type:* datetime. Nullable The time VMware services were tracked, or is NULL if no tracking is recorded.
`SequenceNumber`	*Type:* integer. Nullable Used when generating a differential inventory.
`OVMMDate`	*Type:* datetime. Nullable The time Oracle VM manager was interrogated, or is NULL if no interrogation is recorded.
`AccessDate`	*Type:* datetime. Nullable Access time information was tracked, or is NULL if no tracking is recorded.
`AgentVersion`	*Type:* text (max 32 characters). Nullable The version of the agent which generated the inventory package.

ServiceComponent Table

A software component installed to implement a `ServiceProvider`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 815: Database columns for `ServiceComponent` table

Database Column	Details
`ServiceComponentID`	*Type:* integer. Key. Generated ID Unique ID for the service component.
`Name`	*Type:* text (max 128 characters). Key The name of the service component.
`Version`	*Type:* text (max 32 characters). Key The version of the service component.
`Publisher`	*Type:* text (max 128 characters). Key The publisher of the service component.
`Edition`	*Type:* text (max 128 characters). Key The edition of the service component.

ServiceProvider Table

The inventoried providers of services.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 816: Database columns for `ServiceProvider` table

Database Column	Details
`ServiceProviderID`	*Type:* integer. Key. Generated ID Unique ID for the service provider.
`ComputerID`	*Type:* integer. Key The `Computer` this service provider is hosted by.
`ParentServiceProviderID`	*Type:* integer. Key. Nullable The `ServiceProvider` this provider is parented by.
`EnterpriseManager ServiceProviderID`	*Type:* integer. Key. Nullable The `ServiceProvider` of the OEM instance that manages this provider.
`ServiceProviderTypeID`	*Type:* integer. Key The `ServiceProviderType` of the service provider.
`ServiceProviderNameID`	*Type:* integer. Key The `ServiceProviderName` of the service provider.
`LastInventoryDate`	*Type:* datetime The date and time that the service provider was last inventoried.
`LastInventoryResult`	*Type:* integer. Nullable The error code returned when the service provider was last inventoried.
`LastInventoryError`	*Type:* text (max 256 characters). Nullable The error message returned when the service provider was last inventoried.
`CreationDate`	*Type:* datetime. Nullable The date and time that the service provider was created.
`AuditEvidence`	*Type:* binary. Nullable The Oracle LMS audit evidence in zip archive.

ServiceProviderApplicationOracle Table

An Oracle application.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 817: Database columns for `ServiceProviderApplicationOracle` table

Database Column	Details
`ServiceProvider ApplicationOracleID`	*Type:* integer. Key. Generated ID Unique ID for the Oracle application.
`ServiceProviderID`	*Type:* integer. Key Unique ID for the service provider.
`Name`	*Type:* text (max 240 characters). Key The application name.
`Users`	*Type:* integer The number of users.
`ApplicationID`	*Type:* integer. Key The ID of the application as assigned by Oracle.

ServiceProviderApplicationUsagePerMonth Table

A count of oracle application usage items per month.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 818: Database columns for `ServiceProviderApplicationUsagePerMonth` table

Database Column	Details
`ServiceProvider ApplicationUsagePer MonthID`	*Type:* integer. Key. Generated ID Unique ID for the Oracle per month summary count.
`ServiceProviderID`	*Type:* integer. Key Unique ID for the service provider.

Database Column	Details
ServiceProvider ApplicationUsageTypeID	*Type:* integer. Key The ServiceProviderApplicationUsageType of the service provider application usage.
YearMonth	*Type:* datetime. Key The year and month of the count.
ItemsUsed	*Type:* integer The number of items used.

ServiceProviderApplicationUsageType Table

The types of inventoried ServiceProviderApplicationUsagePerMonth items.

Table 819: Database columns for ServiceProviderApplicationUsageType table

Database Column	Details
ServiceProvider ApplicationUsageTypeID	*Type:* integer. Key. Generated ID Unique ID for the service provider application usage item type.
Type	*Type:* text (max 128 characters). Key The type of a service provider application usage item.

ServiceProviderApplicationUserOracle Table

An Oracle applications User.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 820: Database columns for ServiceProviderApplicationUserOracle table

Database Column	Details
ServiceProvider ApplicationUserOracleID	*Type:* integer. Key. Generated ID Unique ID for the Oracle application user.
ServiceProvider ApplicationOracleID	*Type:* integer. Key The application this user is associated with.

Database Column	Details
UserID	*Type:* integer. Key The application users user ID.
UserName	*Type:* text (max 100 characters) The application users user name.
Description	*Type:* text (max 240 characters). Nullable The application users description.
EMail	*Type:* text (max 240 characters). Nullable The application users email address.

ServiceProviderComponent Table

A software component installed to implement a ServiceProvider.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 821: Database columns for ServiceProviderComponent table

Database Column	Details
ServiceProviderID	*Type:* integer. Key The ServiceProvider this component is associated with.
ServiceComponentID	*Type:* integer. Key The ServiceComponent this provider is associated with.

ServiceProviderComponentProperty Table

The ServiceProviderComponentProperty table provides property names and values for each service component on a provider.

Table 822: Database columns for ServiceProviderComponentProperty table

Database Column	Details
ServiceProvider ComponentPropertyID	*Type:* integer. Key. Generated ID Auto-generated identity number

Database Column	Details
`Property`	*Type:* text (max 256 characters). Key The service component property. A single service component on a provider can have many properties.

ServiceProviderComponentValue Table

The value of a specified `ServiceProviderComponentProperty` of the specified `ServiceProviderComponent`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 823: Database columns for `ServiceProviderComponentValue` table

Database Column	Details
`ServiceProviderID`	*Type:* integer. Key Service provider.
`ServiceComponentID`	*Type:* integer. Key Service component.
`ServiceProvider ComponentPropertyID`	*Type:* integer. Key Property.
`Value`	*Type:* text (max 256 characters). Nullable Property value.

ServiceProviderName Table

The names of inventoried `ServiceProvider`s.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 824: Database columns for `ServiceProviderName` table

Database Column	Details
`ServiceProviderNameID`	*Type:* integer. Key. Generated ID Unique ID for the service provider name.

Database Column	Details
Name	*Type:* text (max 128 characters). Key The name of a service provider.

ServiceProviderProperty Table

The ServiceProviderProperty table provides property names and values for each service provider.

Table 825: Database columns for ServiceProviderProperty table

Database Column	Details
ServiceProviderPropertyID	*Type:* integer. Key. Generated ID Auto-generated identity number
Property	*Type:* text (max 256 characters). Key The service provider property. A single service provider can have many properties.

ServiceProviderType Table

The types of inventoried ServiceProviders.

Table 826: Database columns for ServiceProviderType table

Database Column	Details
ServiceProviderTypeID	*Type:* integer. Key. Generated ID Unique ID for the service provider type.
Type	*Type:* text (max 128 characters). Key The type of a service provider.

ServiceProviderValue Table

The value of a specified ServiceProviderProperty of the specified ServiceProvider.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 827: Database columns for `ServiceProviderValue` table

Database Column	Details
`ServiceProviderID`	*Type:* integer. Key Service provider.
`ServiceProviderPropertyID`	*Type:* integer. Key Property.
`Value`	*Type:* text (max 256 characters). Nullable Property value.

ServiceUser Table

A user that uses a `ServiceProvider`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 828: Database columns for `ServiceUser` table

Database Column	Details
`ServiceUserID`	*Type:* integer. Key. Generated ID Unique ID for the service user.
`ServiceProviderID`	*Type:* integer. Key The `ServiceProvider` this user is associated with.
`Name`	*Type:* text (max 128 characters). Key The name of the service user.
`Description`	*Type:* text (max 256 characters). Nullable A textual description of the service user.
`AccountStatus`	*Type:* text (max 256 characters). Nullable Current status of user account.
`CreationDate`	*Type:* datetime. Nullable Date and time when user was created.
`LastLogonDate`	*Type:* datetime. Nullable Date and time when user last logged on.

ServiceUserOracle Table

A specific kind of `ServiceUser`, specifically an Oracle user.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 829: Database columns for `ServiceUserOracle` table

Database Column	Details
`ServiceUserOracleID`	*Type:* integer. Key. Generated ID Unique ID for the Oracle service user.
`ServiceUserID`	*Type:* integer. Key The service user this user is associated with.
`DefaultTablespace`	*Type:* text (max 256 characters). Nullable The default tablespace for the user.
`TempTablespace`	*Type:* text (max 256 characters). Nullable The temporary tablespace for the user.

SoftwareDetails Table

The `SoftwareDetails` table contains a record of detailed data for each `SoftwareOccurrence` found.

Table 830: Database columns for `SoftwareDetails` table

Database Column	Details
`SoftwareDetailsID`	*Type:* integer. Key. Generated ID The id for the software details. This is automatically generated by SQL Server.
`RawSoftwareName`	*Type:* text (max 128 characters). Key The name of the software defined by the vendor, unprocessed by FlexNet Manager Suite.
`RawVersion`	*Type:* text (max 32 characters). Key The version of the software defined by the vendor, unprocessed by FlexNet Manager Suite.
`Publisher`	*Type:* text (max 256 characters). Key The publisher of the software defined by the vendor.

Database Column	Details
ProductID	*Type:* text (max 256 characters). Key The MSI product ID of the software defined by the vendor.

SoftwareFile Table

The SoftwareFile table contains a record for each file associated with an application on each computer. File tracking is not enabled by default. For more information on configuring which files to track, see the section about the Inventory Agent.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 831: Database columns for SoftwareFile table

Database Column	Details
SoftwareFileID	*Type:* big integer. Key. Generated ID The id for the software file. This is automatically generated by SQL Server.
ComputerID	*Type:* integer. Key The computer on which the file was tracked. This is a foreign key into the Computer table.
SoftwareID	*Type:* integer. Key. Nullable The software containing the file that was tracked. This is a foreign key into the SoftwareVersion table.
SoftwareIsoTagFileID	*Type:* integer. Key. Nullable The software ID tag content of the file. This is a foreign key into the SoftwareIsoTagFile table.
Version	*Type:* text (max 32 characters). Nullable The version of the software file defined by the vendor.
MD5	*Type:* text (max 32 characters) The file's MD5 digest.
Size	*Type:* integer The file's size in bytes.
DateTime	*Type:* datetime. Nullable The last date and time the file was modified on the computer.

Database Column	Details
`FileVersion`	*Type:* text (max 256 characters). Nullable The file version of the software file defined by the vendor.
`FileDescription`	*Type:* text (max 256 characters). Nullable The file description of the software file defined by the vendor.
`Language`	*Type:* text (max 256 characters). Nullable The language of the software file defined by the vendor.
`CompanyName`	*Type:* text (max 256 characters). Nullable The company name of the software file defined by the vendor.
`SoftwareFilePathID`	*Type:* integer. Key. Nullable The full path to the file that was tracked, minus the filename. This is a foreign key into the `SoftwareFilePath` table.
`SoftwareFileNameID`	*Type:* integer. Key. Nullable The name of the file that was tracked, minus the path. This is a foreign key into the `SoftwareFileName` table.

SoftwareFileName Table

The `SoftwareFileName` table contains a record for each unique file name for files captured in inventory.

Table 832: Database columns for `SoftwareFileName` table

Database Column	Details
`SoftwareFileNameID`	*Type:* integer. Key. Generated ID The id for the software file name. This is automatically generated by SQL Server.
`Name`	*Type:* text (max 400 characters). Key The name of a file captured in inventory, minus the path.
`CreationDate`	*Type:* datetime. Key The creation date of the SoftwareFileName which will be used to cleanup the older unused records.

SoftwareFilePath Table

The `SoftwareFilePath` table contains a record for each unique file path for files captured in inventory.

Table 833: Database columns for `SoftwareFilePath` table

Database Column	Details
`SoftwareFilePathID`	*Type:* integer. Key. Generated ID The id for the software file path. This is automatically generated by SQL Server.
`Path`	*Type:* text (max 400 characters). Key The full path to a file captured in inventory, minus the filename.
`CreationDate`	*Type:* datetime. Key The creation date of the SoftwareFilePath which will be used to cleanup the older unused records.

SoftwareIsoTagEntity Table

The `SoftwareIsoTagEntity` table provides property names and values for each unique entities on software ID tags.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 834: Database columns for `SoftwareIsoTagEntity` table

Database Column	Details
`SoftwareIsoTagEntityID`	*Type:* integer. Key. Generated ID The `SoftwareIsoTagEntity` table unique ID for each records.
`RegID`	*Type:* text (max 200 characters). Key The unique registration ID value of an entity in an software ID tag.
`Name`	*Type:* text (max 200 characters). Key The entity name value in a software ID tag.

SoftwareIsoTagFile Table

The `SoftwareIsoTagFile` table provides property names and values for each Software ID Tag in a normalized manner.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 835: Database columns for `SoftwareIsoTagFile` table

Database Column	Details
`SoftwareIsoTagFileID`	*Type:* integer. Key. Generated ID The `SoftwareIsoTagFile` that this property belongs to
`MD5`	*Type:* text (max 32 characters). Key The MD5 propery value of software ID tag file.
`TagContent`	*Type:* text The actual content of the software id tag file.
`EntitlementRequiredIndicator`	*Type:* boolean. Nullable The entitlement required indicator value of the software ID tag.
`SoftwareIsoTagSoftwareVersionID`	*Type:* integer. Key. Nullable The product version and name identifier for this software. This is a foreign key into the `SoftwareIsoTagSoftwareVersion` table.
`SoftwareCreatorEntityID`	*Type:* integer. Key. Nullable The software creator related data for software ID tag. This is a foreign key into the `SoftwareIsoTagEntity` table.
`SoftwareLicensorEntityID`	*Type:* integer. Key. Nullable The software licensor related data for software ID tag. This is a foreign key into the `SoftwareIsoTagEntity` table.
`TagCreatorEntityID`	*Type:* integer. Key. Nullable The tag creator related data for software ID tag. This is a foreign key into the `SoftwareIsoTagEntity` table.
`OriginalArpGuid`	*Type:* text (max 200 characters). Nullable The original GUID of add-remove programs values of a repackaged software.
`OriginalArpPublisher`	*Type:* text (max 200 characters). Nullable The original publisher of add-remove programs values of a repackaged software.
`OriginalArpDisplayName`	*Type:* text (max 200 characters). Nullable The original display name of add-remove programs values of a repackaged software.
`OriginalArpDisplayVersion`	*Type:* text (max 200 characters). Nullable The original display version of add-remove programs values of a repackaged software.
`CurrentArpGuid`	*Type:* text (max 200 characters). Nullable The current GUID of add-remove programs values of a repackaged software.

Database Column	Details
`CurrentArpPublisher`	*Type:* text (max 200 characters). Nullable The current publisher of add-remove programs values of a repackaged software.
`CurrentArpDisplayName`	*Type:* text (max 200 characters). Nullable The current display name of add-remove programs values of a repackaged software.
`CurrentArpDisplayVersion`	*Type:* text (max 200 characters). Nullable The current display version of add-remove programs values of a repackaged software.
`AdminStudioAppCatalogID`	*Type:* text (max 200 characters). Nullable Application catalog ID of a repackaged application in AdminStudio.
`IsValidSchema`	*Type:* boolean. Nullable Whether the software id tag has valid schema.
`IsValidSignature`	*Type:* boolean. Nullable Whether the software id tag has valid digital signature.
`ActivationStatus`	*Type:* text (max 50 characters). Nullable The activation status value of software ID tag.
`ChannelType`	*Type:* text (max 200 characters). Nullable The channel type value of software ID tag.
`SerialNumber`	*Type:* text (max 200 characters). Nullable The serial number value of software ID tag.
`ParseErrorMessage`	*Type:* text (max 1000 characters). Nullable The message of the error occured while reading the software iso tag file.

SoftwareIsoTagSoftwareVersion Table

The `SoftwareIsoTagSoftwareVersion` table provides property names and values for each software ID tag unique product related data.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 836: Database columns for `SoftwareIsoTagSoftwareVersion` table

Database Column	Details
`SoftwareIsoTagSoftwareVersionID`	*Type:* integer. Key. Generated ID The `SoftwareIsoTagSoftwareVersion` table unique ID for each records.
`TagCreatorEntityID`	*Type:* integer. Key The tag creator related data for software ID tag. This is a foreign key into the `SoftwareIsoTagEntity` table.
`TagSoftwareUniqueID`	*Type:* integer. Key. Nullable The software unique ID related data for software ID tag. This is a foreign key into the `SoftwareIsoTagUnique` table.
`ProductTitle`	*Type:* text (max 200 characters). Key The product title value for software ID tag.
`ProductVersionName`	*Type:* text (max 200 characters). Key The product version name value for software ID tag.
`ProductVersionMajor`	*Type:* integer. Key The major version value of software ID tag.
`ProductVersionMinor`	*Type:* integer. Key The minor version value of software ID tag.
`ProductVersionBuild`	*Type:* integer. Key The build version value of software ID tag.
`ProductVersionReview`	*Type:* integer. Key The review version value of software ID tag.

SoftwareIsoTagUnique Table

The `SoftwareIsoTagUnique` table provides property names and values for each unique id on software ID tags.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 837: Database columns for `SoftwareIsoTagUnique` table

Database Column	Details
`SoftwareIsoTagUniqueID`	*Type:* integer. Key. Generated ID The `SoftwareIsoTagUniqueID` table unique ID for each records.

Database Column	Details
UniqueID	*Type:* text (max 200 characters). Key The unique ID value of a software ID tag.

SoftwareOccurrence Table

The SoftwareOccurrence table contains the list (by computer and user) of applications that are installed. The applications may not have been installed through FlexNet Manager Suite. The information is obtained from managed devices from:

- FlexNet Manager Suite packages cache
- Add/Remove Programs registry entries
- Microsoft Installer
- ProductVersion resource strings in program files, if files are tracked

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 838: Database columns for SoftwareOccurrence table

Database Column	Details
SoftwareOccurrenceID	*Type:* integer. Key. Generated ID The id for the software occurrence. This is automatically generated by SQL Server.
ComputerID	*Type:* integer. Key The computer on which the software was tracked. For user inventory, this is the computer that the user was logged on to at the time of the Generate Inventory event. This is a foreign key into the Computer table.
UserID	*Type:* integer. Key User for whom the SoftwareVersion was installed. This is a foreign key to the User table.
SoftwareID	*Type:* integer. Key The software that has been tracked. This is a foreign key to the SoftwareVersion table.
SoftwareDetailsID	*Type:* integer. Key The details that have been tracked. This is a foreign key to the SoftwareDetails table.

Database Column	Details
`Evidence`	*Type:* text (max 32 characters). Nullable An indication of how the software was determined to be on the managed device. The valid entries are: • msi • managesoft • uninstall • exehdr (for file tracking only) • dllhdr (for file tracking only)
`PackagePathID`	*Type:* integer. Key. Nullable FlexNet Manager Suite `PackageFullName` if known (not always!).
`PolicyGUID`	*Type:* binary (max 16 bytes). Nullable FlexNet Manager Suite `Policy` GUID if known.
`InstallationDate`	*Type:* datetime. Nullable The date and time that the software was installed.

SoftwareOccurrenceSoftwareIsoTagFile Table

The `SoftwareOccurrenceSoftwareIsoTagFile` table is link table joining records in `SoftwareOccurrence` and `SoftwareIsoTagFile` tables.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 839: Database columns for `SoftwareOccurrenceSoftwareIsoTagFile` table

Database Column	Details
`SoftwareOccurrence SoftwareIsoTagFileID`	*Type:* integer. Key. Generated ID The `SoftwareOccurrenceSoftwareIsoTagFile` table unique ID for each records.
`SoftwareOccurrenceID`	*Type:* integer. Key This is a foreign key into the `SoftwareOccurrence` table.

Database Column	Details
SoftwareIsoTagFileID	*Type:* integer. Key This is a foreign key into the SoftwareIsoTagFile table.

SoftwareProperty Table

The SoftwareProperty table contains a record for each unique property name captured in inventory.

Table 840: Database columns for SoftwareProperty table

Database Column	Details
SoftwarePropertyID	*Type:* integer. Key. Generated ID The id for the software property. This is automatically generated by SQL Server.
Property	*Type:* text (max 256 characters). Key The software property. A single software object can have many properties.

SoftwareValue Table

The value of a specified SoftwareProperty of the specified SoftwareOccurrence.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 841: Database columns for SoftwareValue table

Database Column	Details
SoftwareOccurrenceID	*Type:* integer. Key Object.
SoftwarePropertyID	*Type:* integer. Key Property.
Value	*Type:* text (max 256 characters). Nullable Property value.

SoftwareVersion Table

The SoftwareVersion table contains a record for each software name/version combination returned through inventory. The software names and versions are gathered from places such as Add/Remove Programs on managed

devices. They do not represent package names and versions from the software library, although correlation is likely.

Table 842: Database columns for `SoftwareVersion` table

Database Column	Details
`SoftwareID`	*Type:* integer. Key. Generated ID The id for the software version. This is automatically generated by SQL Server.
`SoftwareName`	*Type:* text (max 128 characters). Key The name of the software defined by the vendor.
`Version`	*Type:* text (max 32 characters). Key The version of the software defined by the vendor.
`CreationDate`	*Type:* datetime. Key The creation date of the SoftwareVersion which will be used to cleanup the older unused records.

VirtualDesktopAccess Table

A VDI device a `User` has accessed on an end-point.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 843: Database columns for `VirtualDesktopAccess` table

Database Column	Details
`ComputerID`	*Type:* integer. Key The end-point ComputerID. This is a foreign key into the `Computer` table.
`UserID`	*Type:* integer. Key The ID for the user accessing the VDI device. This is a foreign key into the `User` table.
`MachineName`	*Type:* text (max 64 characters). Key Computer name of the VDI device.
`MachineDomain`	*Type:* text (max 256 characters). Key. Nullable Fully qualified domain of the VDI device.
`VDITemplateName`	*Type:* text (max 256 characters). Key The template from which the VDI device was cloned.

Database Column	Details
`Type`	*Type:* text (max 64 characters). Key The type of VDI.
`LogonTime`	*Type:* datetime. Key The time the user logged on to the VDI device.
`VirtualDesktopAccessID`	*Type:* integer. Key. Generated ID The ID of the user session to the VDI device.

VirtualDesktopApplicationUsage Table

A virtualized application is used from VDI.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 844: Database columns for `VirtualDesktopApplicationUsage` table

Database Column	Details
`VirtualDesktop ApplicationUsageID`	*Type:* integer. Key. Generated ID The ID of the application usage record.
`VirtualDesktopAccessID`	*Type:* integer. Key The ID of the corresponding VDI access record. This is a foreign key into the `VirtualDesktopAccess` table.
`Name`	*Type:* text (max 64 characters). Key The display name of the virtual application.
`Version`	*Type:* text (max 16 characters). Key The version of the virtual application.
`PackageGUID`	*Type:* unique identifier. Key The GUID of the package that the virtual application is associated with.
`LastLaunchOnSystem`	*Type:* datetime The last date and time that the virtual application was launched.
`AccessMode`	*Type:* text (max 100 characters). Key The access mode for the application.

VirtualDesktopGroupAccess Table

A user with access to a particular VDI Group for a given site.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 845: Database columns for `VirtualDesktopGroupAccess` table

Database Column	Details
`VDISiteName`	*Type:* text (max 256 characters). Key the VDI Site.
`VDIGroupName`	*Type:* text (max 256 characters). Key The name of the VDI Group.
`Sid`	*Type:* text (max 512 characters). Key. Nullable The Sid of the user.
`VDIBrokerType`	*Type:* text (max 64 characters). Key The type of VDI infrastructure.

VirtualDesktopGroupAccessScan Table

The last scan time of the VDI to retrieve ACL information

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 846: Database columns for `VirtualDesktopGroupAccessScan` table

Database Column	Details
`VDIBrokerType`	*Type:* text (max 64 characters). Key The type of VDI.
`VDISiteName`	*Type:* text (max 256 characters). Key The VDI Site.
`ScanTime`	*Type:* datetime The VDI Site.

Licensing Tables

The complete set of database tables documented here includes:

- LicenseAllocation table (see LicenseAllocation Table)
- LicenseModel table (see LicenseModel Table)
- LicensePurchase table (see LicensePurchase Table)
- ProductContainsSoftware table (see ProductContainsSoftware Table)
- SoftwareProduct table (see SoftwareProduct Table)
- SoftwarePublisher table (see SoftwarePublisher Table)
- SoftwareReseller table (see SoftwareReseller Table)

LicenseAllocation Table

The `LicenseAllocation` table specifies the allocation of licenses for each organizational unit. The same licensable product definition (from `SoftwareProduct`) may have license allocations for more than one organizational unit.

Table 847: Database columns for `LicenseAllocation` table

Database Column	Details
`AllocationID`	*Type:* integer. Key. Generated ID Unique identifier for the license allocation record. This is automatically generated by SQL Server.
`SoftwareProductID`	*Type:* integer. Key The license that maps to an application. This is a foreign key into the `SoftwareProduct` table.
`OrganizationID`	*Type:* integer. Key Id of the organizational unit to which the software is allocated.
`UnitsAllocated`	*Type:* integer. Nullable The number of units allocated for the application.
`Expiry`	*Type:* datetime. Nullable The date and time that the license allocation expires.

LicenseModel Table

The `LicenseModel` table defines the license models available (for example, Site license). Each licensable product (listed in `SoftwareProduct`) is assigned a license model. A license model may apply to multiple licensable products.

Table 848: Database columns for `LicenseModel` table

Database Column	Details
`ModelID`	*Type:* integer. Key. Generated ID The unique identifier for a license model.
`Name`	*Type:* text (max 256 characters). Key The name of the license model.

LicensePurchase Table

`LicensePurchase` records details of purchases of licenses for a specified `SoftwareProduct`.

Table 849: Database columns for `LicensePurchase` table

Database Column	Details
`SoftwareProductID`	*Type:* integer. Key The `SoftwareProduct` purchased.
`ResellerID`	*Type:* integer. Key The `Reseller` from which the software product was purchased.
`OrganizationID`	*Type:* integer. Key The organizational unit that owns the license for the product.
`Purchased`	*Type:* datetime. Key When the purchase was made.
`Expires`	*Type:* datetime. Nullable When the license expires.
`Price`	*Type:* integer. Nullable The price paid for the license.
`Quantity`	*Type:* integer Number of units licensed.
`OrderNumber`	*Type:* text (max 32 characters). Key Cross-reference to customer's purchase order number.

ProductContainsSoftware Table

The `ProductContainsSoftware` table lists the applications returned by inventory (in the `SoftwareVersion` table) that are covered by licensable products (listed in `SoftwareProduct`). A license can map to multiple applications: if

any of these applications is installed, a license is required.

Table 850: Database columns for `ProductContainsSoftware` table

Database Column	Details
`SoftwareProductID`	*Type:* integer. Key The license that maps to an application. This is a foreign key into the `SoftwareProduct` table.
`SoftwareVersionID`	*Type:* integer. Key The application maps to the license. This is a foreign key into the `SoftwareVersion` table.

SoftwareProduct Table

The `SoftwareProduct` table contains all of the licensable products (license definitions) for an organization. It represents all of the license agreements available for monitoring.

Table 851: Database columns for `SoftwareProduct` table

Database Column	Details
`SoftwareProductID`	*Type:* integer. Key. Generated ID This is a unique identifier for the software product.
`ProductName`	*Type:* text (max 256 characters). Key The name of the license. This normally corresponds to the name of the software product as defined by the vendor.
`ModelID`	*Type:* integer. Key Reference to the Licensing model for FlexNet Manager Suite
`TrackedByID`	*Type:* integer In what units are Licences counted?
`PublisherID`	*Type:* integer. Key Reference to publisher
`Agreement`	*Type:* text (max 256 characters) A URL to the license agreement for the product.[Comments]
`Comments`	*Type:* text. Nullable Additional comments

SoftwarePublisher Table

The `SoftwarePublisher` table lists application publishers (for example, Microsoft). Each licensable product (listed in `SoftwareProduct`) is assigned a publisher. A publisher may be assigned to multiple licensable products.

Table 852: Database columns for `SoftwarePublisher` table

Database Column	Details
`PublisherID`	*Type:* integer. Key. Generated ID The unique identifier for a publisher.
`Name`	*Type:* text (max 256 characters). Key The name of the publisher.
`SupportURL`	*Type:* text (max 256 characters) The support URL.
`SupportPhone`	*Type:* text (max 256 characters) The support phone number.
`ContactName`	*Type:* text (max 256 characters) The name of the contact.
`Comments`	*Type:* text (max 512 characters) An arbitrary comment about the publisher.

SoftwareReseller Table

The `SoftwareReseller` table lists application resellers (usually the organization listed on the purchase order for the product). Each licensable product (listed in `SoftwareProduct`) is assigned an application reseller. A reseller may be assigned to multiple licensable products.

Table 853: Database columns for `SoftwareReseller` table

Database Column	Details
`ResellerID`	*Type:* integer. Key. Generated ID Auto-generated identifier of `Reseller`
`Name`	*Type:* text (max 256 characters). Key The name of the reseller.
`ContactName`	*Type:* text (max 256 characters) The name of the sales contact.

Database Column	Details
`ContactPhone`	*Type:* text (max 256 characters) The contact phone number.
`Comments`	*Type:* text (max 512 characters) An arbitrary comment about the reseller.

ManageSoft Tables

The complete set of database tables documented here includes:

- DatabaseConfiguration table (see DatabaseConfiguration Table)

DatabaseConfiguration Table

The `DatabaseConfiguration` table contains configuration properties for the FlexNet Manager Suite database tables, which are used for ongoing maintenance of the database.

Table 854: Database columns for `DatabaseConfiguration` table

Database Column	Details
`Property`	*Type:* text (max 32 characters). Key The name of the property.
`Value`	*Type:* text (max 256 characters) The value of the property.
`Created`	*Type:* datetime The date and time the property was created.
`LastUpdate`	*Type:* datetime The date and time the property was last updated.

Networking Tables

The complete set of database tables documented here includes:

- NetworkLocation table (see NetworkLocation Table)
- Subnet table (see Subnet Table)

NetworkLocation Table

The Location table contains data about Locations

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 855: Database columns for `NetworkLocation` table

Database Column	Details
`NetworkLocationID`	*Type:* integer. Key. Generated ID The ID for the Location
`Name`	*Type:* text (max 256 characters). Key The name of the Location
`DN`	*Type:* text (max 1024 characters). Key. Nullable The Distinguished name of the Location
`AutoPopulated`	*Type:* boolean Specifies whether the row was populated automatically(1) or manually(0).
`Enabled`	*Type:* boolean Specifies whether the row will be used when mapping domains and devices to Locations
`DomainID`	*Type:* integer. Key DomainID of the domain in which the NetworkLocation resides

Subnet Table

The Subnet table contains data about subnets in a location.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 856: Database columns for `Subnet` table

Database Column	Details
`SubnetID`	*Type:* integer. Key. Generated ID The ID for the Subnet

Database Column	Details
`IPSubnet`	*Type:* text (max 64 characters). Key The IPSubnet of the Subnet
`IPSubnetMask`	*Type:* text (max 64 characters). Key The IPSubnetMask of the Subnet
`NetworkLocationID`	*Type:* integer. Key NetworkLocationID of the NetworkLocation in which the Subnet resides
`AutoPopulated`	*Type:* boolean Specifies whether the row was populated automatically(1) or manually(0).
`Enabled`	*Type:* boolean Specifies whether the row will be used when mapping domains and devices to Locations

Packaging Tables

The complete set of database tables documented here includes:

- Architecture table (see Architecture Table)
- FileNameMap table (see FileNameMap Table)
- Media table (see Media Table)
- MediaContainsPackagePath table (see MediaContainsPackagePath Table)
- MediaContainsPackageVersion table (see MediaContainsPackageVersion Table)
- MediaType table (see MediaType Table)
- PackageFamily table (see PackageFamily Table)
- PackagePath table (see PackagePath Table)
- PackagePathType table (see PackagePathType Table)
- PackageProvides table (see PackageProvides Table)
- PackageRequires table (see PackageRequires Table)
- PackageState table (see PackageState Table)
- PackageVersion table (see PackageVersion Table)
- PackageVersionArchitecture table (see PackageVersionArchitecture Table)
- PackageVersionEnvironment table (see PackageVersionEnvironment Table)
- PackageVersionInState table (see PackageVersionInState Table)

- PackageVersionLocale table (see PackageVersionLocale Table)

Architecture Table

`Architecture` identifies a target CPU (ABI), used to identify on what type of computer a package may be installed.

Table 857: Database columns for `Architecture` table

Database Column	Details
`ArchitectureID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`ArchitectureName`	*Type:* text (max 64 characters). Key Name of the computer architecture

FileNameMap Table

Stores mappings from a file on disk to a filename that will be used on the managed device. Currently used by the HPUX wizards to rename files that have non-Windows conforming names.

Table 858: Database columns for `FileNameMap` table

Database Column	Details
`MediaID`	*Type:* integer. Key The `Media` that the `SourceFile` exists on.
`SourceFile`	*Type:* text (max 256 characters). Key The file to be renamed.
`DestFile`	*Type:* text (max 256 characters) The final file name.
`IsFile`	*Type:* boolean Boolean field that specifies whether the row refers to a file or a directory.

Media Table

Packages are stored on `Media` identified in this table.

Table 859: Database columns for `Media` table

Database Column	Details
`MediaID`	*Type:* integer. Key. Generated ID Auto-generated identity number, 1 = local administration server.
`Title`	*Type:* text (max 128 characters). Key Name of media (empty for local administration server).
`MediaTypeID`	*Type:* integer. Key What type of media?
`Location`	*Type:* text (max 256 characters). Nullable Where on the media?

MediaContainsPackagePath Table

This table identifies which `Media` contains which `PackagePath`. A record exists here at least for every `PackagePath` currently in the local administration server.

Table 860: Database columns for `MediaContainsPackagePath` table

Database Column	Details
`MediaID`	*Type:* integer. Key What `Media` contains the package?
`PackagePathID`	*Type:* integer. Key What `PackagePath`?

MediaContainsPackageVersion Table

This table identifies which `Media` contains which `PackageVersion`. A record exists here at least for every `PackageVersion` currently in the local administration server.

Table 861: Database columns for `MediaContainsPackageVersion` table

Database Column	Details
`MediaID`	*Type:* integer. Key What `Media` contains the package?
`PackageVersionID`	*Type:* integer. Key What `PackageVersion`?

MediaType Table

Packages are stored on `Media` of various types. This table contains a record for each type.

Table 862: Database columns for `MediaType` table

Database Column	Details
`MediaTypeID`	*Type:* integer. Key. Generated ID Auto-generated identity number, 1 = Warehouse (administration server).
`Description`	*Type:* text (max 128 characters). Key `Media` type name (for example: Warehouse, Filesystem, CD).

PackageFamily Table

`PackageFamily` is a short name used by the client to decide where a package to be downloaded to and whether it's an upgrade or downgrade of a previous package. Only one package version of a family may be installed in a given context.

Table 863: Database columns for `PackageFamily` table

Database Column	Details
`PackageFamilyID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`PackageName`	*Type:* text (max 64 characters). Key Package family name

PackagePath Table

Package Path identifies a filesystem path where the package will be stored in the software library and in transit. As such, the administration server and each distribution server may only contain a single package version having a given Path

Table 864: Database columns for `PackagePath` table

Database Column	Details
`PackagePathID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`PackageFullName`	*Type:* text (max 256 characters). Key. Nullable Package Path string

Database Column	Details
ParentPathID	*Type:* integer. Key. Nullable When a package is a variant of a parent package, this contains a reference to the parent package's path.
PackagePathTypeID	*Type:* integer The type of PackagePath that this row represents.

PackagePathType Table

This table contains the list of different types of packages, which also corresponds to the main areas in the Software Library.

Table 865: Database columns for PackagePathType table

Database Column	Details
PackagePathTypeID	*Type:* integer. Key. Generated ID Auto-generated identity number
Description	*Type:* text (max 128 characters). Key This describes the type of the package

PackageProvides Table

PackageProvides is used when a package can satisfy a virtual dependency, like "web-browser".

Table 866: Database columns for PackageProvides table

Database Column	Details
PackageProvidesID	*Type:* integer. Key. Generated ID Auto-generated identity number
PackageVersionID	*Type:* integer. Key The package which provides the interface
PackageFamilyID	*Type:* integer. Key The (virtual) package which is provided
Version	*Type:* text (max 32 characters). Key. Nullable The version provided, if necessary

PackageRequires Table

`PackageRequires` is used when a package requires another package or some other configuration, like a piece of hardware for example.

Table 867: Database columns for `PackageRequires` table

Database Column	Details
`PackageVersionID`	*Type:* integer. Key The `PackageVersion` which has the requirement.
`RequiredType`	*Type:* text (max 8 characters). Key Requirement type: for example, software, hardware.
`RequiredObject`	*Type:* text (max 64 characters). Key Required object: for example, `PackageFamily` name.
`Strength`	*Type:* integer. Nullable Strength of the requirement.
`Property`	*Type:* text (max 64 characters). Nullable The required property of the object (for example, package version).
`Value`	*Type:* text (max 64 characters). Nullable The value of the required property.
`Match`	*Type:* integer. Key How to match the required value.

PackageState Table

This table contains the package states that may be assigned to a package in the software library. The default set of states are based on ITIL release management processes.

Table 868: Database columns for `PackageState` table

Database Column	Details
`PackageStateID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`Name`	*Type:* text (max 64 characters). Key Package State Name
`CanAddToPolicy`	*Type:* boolean Whether a package in this state can be added to policy

PackageVersion Table

The PackageVersion table contains information about all of the packages in the software library. It is primarily used to map between Installation and PackageApplies for the purpose of comparing what users and computers should have versus what they actually have installed. This table only stores the details of one version of each package. This will change in future releases.

Table 869: Database columns for PackageVersion table

Database Column	Details
PackageVersionID	*Type:* integer. Key. Generated ID Auto-generated identity number
PackagePathID	*Type:* integer. Key Reference to Path (Full name) of Package
Version	*Type:* text (max 32 characters). Key The version number of the package. The Installation table also has PackageName and Version columns. This value can be used to find the corresponding PackageFullName so that Installation can be mapped to PackageApplies.
Update	*Type:* text (max 64 characters). Key The current update (or patch) number of the package
PackageFamilyID	*Type:* integer. Key A managed device may only have one PackageVersion in a family.
Title	*Type:* text (max 64 characters). Nullable The friendly name for the package.
MD5	*Type:* text (max 40 characters). Nullable The MD5 digest of the project file (.ndp) for the package. This is updated in the database when the package is packed or distributed.
Size	*Type:* integer. Nullable If set, contains the size in bytes of the distributable form of the package
Category	*Type:* text (max 128 characters). Nullable A category or class used to group packages

PackageVersionArchitecture Table

PackageVersionArchitecture specifies all the architectures that a particular package version applies to.

Table 870: Database columns for `PackageVersionArchitecture` table

Database Column	Details
`PackageVersionID`	*Type:* integer. Key Foreign key into the `PackageVersion` table.
`ArchitectureID`	*Type:* integer. Key Foreign key into the `Architecture` table.

PackageVersionEnvironment Table

`PackageVersionEnvironment` specifies all the environments (operating systems) that a particular package version applies to.

Table 871: Database columns for `PackageVersionEnvironment` table

Database Column	Details
`PackageVersionID`	*Type:* integer. Key Foreign key into the `PackageVersion` table.
`Environment`	*Type:* text (max 128 characters). Key Name of the environment that is used in the package. This refers to the environments used in the Packer.

PackageVersionInState Table

This table contains a history of changes made to the state of a package. Note that the username is recorded as a nvarchar rather than a foreign key to the user table so that if a user is deleted, there is still a record of the changes that were made.

Table 872: Database columns for `PackageVersionInState` table

Database Column	Details
`PackageVersionID`	*Type:* integer. Key The package that has been changed
`PackageStateID`	*Type:* integer. Key The state that was set
`UserName`	*Type:* text (max 64 characters). Key The user that made the state change

Database Column	Details
`Changed`	*Type:* datetime. Key The date/time that the change was made
`Comments`	*Type:* text (max 256 characters) A user defined set of comments relating to the state change

PackageVersionLocale Table

`PackageVersionLocale` specifies all the locales (language and country combinations) that a particular package version applies to.

Table 873: Database columns for `PackageVersionLocale` table

Database Column	Details
`PackageVersionID`	*Type:* integer. Key Foreign key into the `PackageVersion` table.
`LocaleCode`	*Type:* text (max 6 characters). Key Foreign key into the `Locale` table.

ReferenceData Tables

The complete set of database tables documented here includes:

- Country table (see Country Table)
- Language table (see Language Table)
- Locale table (see Locale Table)

Country Table

Stores country information, including their ISO country code and English names.

Table 874: Database columns for `Country` table

Database Column	Details
`CountryCode`	*Type:* text (max 2 characters). Key The two letter country code.

Database Column	Details
Name	*Type:* text (max 128 characters). Key The english name of the country.

Language Table

Stores language information, including their English names, and various forms of language id.

Table 875: Database columns for Language table

Database Column	Details
LangCode3	*Type:* text (max 3 characters). Key The three letter language code.
LangCode2	*Type:* text (max 2 characters). Nullable The two letter language code.
EnglishName	*Type:* text (max 128 characters). Key The english name of the language.
LocalName	*Type:* text (max 128 characters). Nullable The name of the language, written in the local language.
MSLanguageID	*Type:* integer. Nullable The Microsoft language id, as specified in winnt.h in the Platform SDK.

Locale Table

Stores locale information, which consists of country and language combinations. Use the LocaleCode column as the foreign key into this table.

Table 876: Database columns for Locale table

Database Column	Details
LocaleCode	*Type:* text (max 6 characters). Key A combination of the language code and country code, separated by a hyphen. If there is no country code, then there will be no hyphen added. This column MUST have the correct value when inserted, based on the values of the language and country codes.
LangCode3	*Type:* text (max 3 characters). Key The three letter language code.

Database Column	Details
`CountryCode`	*Type:* text (max 2 characters). Key. Nullable The two letter country code.
`LocaleName`	*Type:* text (max 128 characters) The name of the locale. For example, “English (United States)”.
`MSLocaleID`	*Type:* integer. Nullable The Microsoft identifier for the locale. For example, 1033 for English (United States).

Rights Tables

The complete set of database tables documented here includes:

- ActionClass table (see ActionClass Table)
- PartitionType table (see PartitionType Table)
- Resource table (see Resource Table)

ActionClass Table

The types of action on a `Resource` for which rights may be granted or denied.

Table 877: Database columns for `ActionClass` table

Database Column	Details
`ActionClassID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`ActionClassName`	*Type:* text (max 16 characters). Key The name of the `ActionClass`.

PartitionType Table

Some secured `Resources` may be partitioned. Partitions are used to grant rights to one part of a `Resource` excluding other parts, for example limiting rights so that the operator can access only certain distribution servers, organizational units, or areas in the software library. There are three types of partitioning, defined by entries in this table.

Table 878: Database columns for `PartitionType` table

Database Column	Details
`PartitionTypeID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`PartitionTypeName`	*Type:* text (max 32 characters). Key Name of the `PartitionType`.

Resource Table

Access rights are granted to the `Resources` defined in this table.

Table 879: Database columns for `Resource` table

Database Column	Details
`ResourceID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`ResourceName`	*Type:* text (max 16 characters). Key Name of the `Resource`.
`PartitionTypeID`	*Type:* integer. Nullable If not NULL, the type of partitioning used with this `Resource`.

ScriptResult Tables

The complete set of database tables documented here includes:

- ComputerScriptResult table (see ComputerScriptResult Table)

ComputerScriptResult Table

This table are used to store recognition rules and their results

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 880: Database columns for `ComputerScriptResult` table

Database Column	Details
`ComputerID`	*Type:* integer. Key The computer that the installation event occurred on. This is a foreign key into the `Computer` table.
`RecognitionRule`	*Type:* text (max 256 characters). Key The recognition rule.
`Revision`	*Type:* integer. Nullable The revision number of the recognition rule.
`InventoryDate`	*Type:* datetime The date the recognition rule ran.
`Result`	*Type:* text. Nullable The result of the recognition rule script.

Status Tables

The complete set of database tables documented here includes:

- AMTEventLog table (see AMTEventLog Table)

AMTEventLog Table

Records the entries in the AMT event log for a `NetworkDevice`.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 881: Database columns for `AMTEventLog` table

Database Column	Details
`AMTEventLogID`	*Type:* integer. Key. Generated ID Auto-generated identity number.
`DeviceID`	*Type:* integer. Key. Nullable `NetworkDevice` identity number.
`Reported`	*Type:* datetime Date and time the event log entry was reported at.

Database Column	Details
`PETDeviceAddress`	*Type:* small integer The device address from the PET message format.
`PETEventSensorType`	*Type:* small integer The event sensor type from the PET message format.
`PETEventType`	*Type:* small integer The event type from the PET message format.
`PETEventOffset`	*Type:* small integer The event offset from the PET message format.
`PETEventSourceType`	*Type:* small integer The event source type from the PET message format.
`PETEventSeverity`	*Type:* small integer The event severity from the PET message format.
`PETSensorNumber`	*Type:* small integer The sensor number from the PET message format.
`PETEntity`	*Type:* small integer The entity from the PET message format.
`PETEntityInstance`	*Type:* small integer The entity instance address from the PET message format.
`PETEventData`	*Type:* text (max 32 characters) The event data from the PET message format.

Targeting Tables

The complete set of database tables documented here includes:

- TargetType table (see TargetType Table)

TargetType Table

The `TargetType` table contains a row for each type of object that can be targeted in FlexNet Manager Suite.

Table 882: Database columns for `TargetType` table

Database Column	Details
`TargetTypeID`	*Type:* integer. Key. Generated ID The ID for the target type: • Computers • Users • Group • DistributionLocation • DistributionServer • Organization • Assets • Contracts • Purchase orders • Software licenses • Software titles • Compliance computers • Compliance users • Operators • SAP system landscapes • SAP systems • SAP rule sets • Discovered devices • Beacon • Vendor • Device • Rule • Inventory connection • FNMP Server • Fast Import • OLE DB Connection • ORACLE Connection

Database Column	Details
	• XML • Intermediate File • ADSI Connection • Web Service • SQL Connection • Software Title Evidence • FNMEA Agent • Installed Software • Baseline Import • Available Package • Client ARL
`TargetTypeName`	*Type:* text (max 256 characters). Key The name of the target type.

Tenants Tables

The complete set of database tables documented here includes:

- FlexeraLicense table (see FlexeraLicense Table)
- Tenant table (see Tenant Table)

FlexeraLicense Table

The `FlexeraLicense` table contains the encoded contents of the Flexera Software licenses required for the tenants in the system. This table is also used by the system in the single-tenant setup where there is only one tenant.

Table 883: Database columns for `FlexeraLicense` table

Database Column	Details
`TenantUID`	*Type:* text (max 40 characters). Key The unique identifier of a tenant. A reference to the `Tenant` to which this license is attached.

Database Column	Details
License	*Type:* text The encoded contents of the Flexera Software license attached to a particular Tenant.
LicenseChecksum	*Type:* integer. Key The check sum of the license.
LicenseDetails	*Type:* XML. Nullable XML definition of the license details

Tenant Table

The Tenant table contains the details of each tenant in multitenant FlexNet Manager Suite database tables.

Table 884: Database columns for Tenant table

Database Column	Details
TenantID	*Type:* integer. Key. Generated ID The tenant ID in a multi-tenant database.
TenantUID	*Type:* text (max 40 characters). Key The unique identifier of a tenant. This identifier is used to identify the tenant in environments where tenant information is stored on multiple databases.
TenantName	*Type:* text (max 256 characters). Key The name of the tenant.
TenantDomain	*Type:* text (max 20 characters). Nullable The sub-domain to use for the tenant.
Comments	*Type:* text. Nullable Operator comments about this tenant record.
CreationUser	*Type:* text (max 128 characters). Nullable The operator who created the tenant record.
CreationDate	*Type:* datetime The date the tenant record was created.
UpdatedUser	*Type:* text (max 128 characters). Nullable The name of the operator who last updated the tenant record.
UpdatedDate	*Type:* datetime. Nullable The date the tenant record was last updated.

Database Column	Details
`OrganizationID`	*Type:* integer. Nullable The unique Flexera organization ID.

Usage Tables

The complete set of database tables documented here includes:

- ComputerUsage table (see ComputerUsage Table)
- SoftwareFileUsage table (see SoftwareFileUsage Table)
- SoftwareUsagePerWeek table (see SoftwareUsagePerWeek Table)

ComputerUsage Table

Each time usage information is received, the `ComputerUsage` table is updated with the current day's time-stamp.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 885: Database columns for `ComputerUsage` table

Database Column	Details
`ComputerID`	*Type:* integer. Key The id of the computer this information applies to. This id is a foreign key to the `Computer` table. It forms part of the unique index that identifies each row of data.
`UserID`	*Type:* integer. Key The id of the user context in which the application was detected. This is a foreign key to the `User` table. It forms part of the unique index that identifies each row of data.
`LastReported`	*Type:* datetime. Nullable The date that the user last reported usage information from the specified computer.

SoftwareFileUsage Table

This table contains information about each file relevant to reporting software usage information on each computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the*

database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.

Table 886: Database columns for `SoftwareFileUsage` table

Database Column	Details
`SoftwareFileUsageID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`ComputerID`	*Type:* integer. Key The id of the computer this information applies to. This id is a foreign key to the `Computer` table. It forms part of the unique index that identifies each row of data.
`UserID`	*Type:* integer. Key The id of the user context in which the application was detected. This is a foreign key to the `User` table. It forms part of the unique index that identifies each row of data.
`Version`	*Type:* text (max 32 characters). Key The version of the software file defined by the vendor.
`SoftwareFileNameID`	*Type:* integer. Key The name of the file that was tracked, minus the path. This is a foreign key into the `SoftwareFileName` table.
`LongName`	*Type:* text (max 4000 characters). Nullable The full path and file that was tracked.
`CompanyName`	*Type:* text (max 50 characters). Key The company name of the software.
`Description`	*Type:* text (max 1024 characters). Key The file description of the software.
`ProductName`	*Type:* text (max 50 characters). Key The product name of the software file.
`ProductVersion`	*Type:* text (max 32 characters). Key The version of the product of the software file defined by the vendor.

SoftwareUsagePerWeek Table

Software usage information is stored in weekly batches. Information received by the server is stored in the `SoftwareUsagePerWeek` table. Each row in the table represents usage information received from a specified user, on a specified managed device, regarding usage of specified software, during the week where the Monday is the specified date.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 887: Database columns for `SoftwareUsagePerWeek` table

Database Column	Details
`SoftwareUsagePerWeekID`	*Type:* integer. Key. Generated ID Auto-generated identity number
`ComputerID`	*Type:* integer. Key The id of the computer this information applies to. This id is a foreign key to the `Computer` table. It forms part of the unique-clustered-index that identifies each row of data.
`UserID`	*Type:* integer. Key The id of the user context in which the application was detected. This id is a foreign key to the `User` table. It forms part of the unique-clustered-index that identifies each row of data.
`SoftwareID`	*Type:* integer. Key The id of the software that was used. This is a foreign key to the `SoftwareVersion` table. It forms part of the unique-clustered-index that identifies each row of data.
`SoftwareFileUsageID`	*Type:* integer. Key. Nullable The id of the software file usage that was used. This is a foreign key to the `SoftwareFileUsage` table. It forms part of the unique-clustered-index that identifies each row of data.
`StartOfWeek`	*Type:* datetime. Key The first day for the week. This date identifies the week that usage data applies to.
`Duration`	*Type:* integer. Nullable The total duration, in seconds, that the application was run. It represents the total spanning across many sessions.
`ActiveTime`	*Type:* integer. Nullable The total active time, in seconds, that the application was in the foreground. It represents the total spanning across many sessions.
`Sessions`	*Type:* integer. Nullable The number of sessions the in which the application was used within the week.
`Days`	*Type:* integer. Nullable The number of distinct days the application was used within the week.

WakeOnLAN Tables

The complete set of database tables documented here includes:

- WakeOnLANDistributionJob table (see WakeOnLANDistributionJob Table)
- WakeOnLANStatus table (see WakeOnLANStatus Table)
- WakeOnLANTask table (see WakeOnLANTask Table)

WakeOnLANDistributionJob Table

Wake on LAN distribution jobs control the distribution of a Wake on LAN task to the nearest distribution server for the targeted managed devices. The status of these distribution jobs is stored in the `WakeOnLANDistributionJob` table. Each row in the table represents a Wake on LAN job, which is any Wake on LAN task (or a subset of a Wake on LAN task), that has been distributed to a distribution server. Be aware: There can be multiple distribution jobs for a given Wake on LAN task.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 888: Database columns for `WakeOnLANDistributionJob` table

Database Column	Details
`DistJobUID`	*Type:* binary (max 16 bytes). Key A unique identifier for this distribution job.
`TaskUID`	*Type:* binary (max 16 bytes). Key A unique identifier for the task that created this distribution job. This is a foreign key linked to the `TaskUID` in the `WakeOnLANTask` table.
`ServerUID`	*Type:* binary (max 16 bytes). Key A unique identifier for the distribution server that this distribution job targets. This foreign key links to the `ServerUID` in the `DistributionServer` table.
`State`	*Type:* text (max 16 characters) The state of this distribution job. This can be one of the following values: + Pending + Failed + Success

WakeOnLANStatus Table

All managed devices targeted by a Wake on LAN task have a status associated with them. The status of the managed devices is stored in the `WakeOnLANStatus` table. Each row in the table represents a managed device to be woken by a Wake on LAN task from a distribution job.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 889: Database columns for `WakeOnLANStatus` table

Database Column	Details
`DistJobUID`	*Type:* binary (max 16 bytes). Key A unique identifier for a distribution job. This foreign key links to the `DistJobUID` in the `WakeOnLANDistributionJob` table. It forms part of the unique index that identifies each row of data.
`ComputerID`	*Type:* integer. Key The id for the managed device. It forms part of the unique index that identifies each row of data.
`State`	*Type:* text (max 16 characters) The state of this managed device. This can be one of the following values: • Pending • Failed • Woken • Awake

WakeOnLANTask Table

Wake on LAN tasks control any targeted managed devices. The details of these tasks are stored in the `WakeOnLANTask` table. Each row in the table represents a Wake on LAN task.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 890: Database columns for `WakeOnLANTask` table

Database Column	Details
`TaskUID`	*Type:* binary (max 16 bytes). Key A unique identifier for the task that created a Wake on LAN job.
`FriendlyName`	*Type:* text (max 400 characters) The descriptive name assigned to the Wake on LAN task.

Database Column	Details
`StartTime`	*Type:* datetime. Nullable The time at which the managed devices will be woken.

WorkFlow Tables

The complete set of database tables documented here includes:

- Action table (see Action Table)
- ActionApplies table (see ActionApplies Table)
- ActionState table (see ActionState Table)
- Job table (see Job Table)
- Task table (see Task Table)
- TaskSchedule table (see TaskSchedule Table)
- TaskType table (see TaskType Table)

Action Table

An `Action` arising from a `Task`, to be applied (possibly repeatedly) by an actor (often a distribution server) to a set of target devices.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 891: Database columns for `Action` table

Database Column	Details
`ActionUID`	*Type:* binary (max 16 bytes). Key The unique identifier for the `Action`.
`TaskID`	*Type:* integer. Key The `Task` which gave rise to this `Action`.
`ServerUID`	*Type:* binary (max 16 bytes). Key. Nullable True if this `Action` has been delegated to a distribution server.
`JobUID`	*Type:* binary (max 16 bytes). Key. Nullable The `Job` which instructed the DS to perform the `Action`, if the `Job` still exists.

Database Column	Details
ActionStateID	*Type:* integer .One of the action states defined in the ActionState table.
PackageVersionID	*Type:* integer. Nullable If Task is of type Distribution, a PackageVersion applies.
FailureReason	*Type:* text. Nullable If not empty, text describing the reason the Action failed.
LastUpdate	*Type:* datetime The last time that the ActionState was updated. This value is the UTC date time of the event.
DSVersion	*Type:* text (max 32 characters). Nullable The version of the DS used to execute the Action.

ActionApplies Table

An action applies/applied to this computer, which can be identified by its computer id, device id, DNS, IP or MAC address. One of the five related cross-references must be non-null. If more than one is non-null, precedence is applied top to bottom in the order documented below.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 892: Database columns for ActionApplies table

Database Column	Details
ActionAppliesID	*Type:* integer. Key. Generated ID Auto-generated identity number
ActionUID	*Type:* binary (max 16 bytes). Key The Action which applies.
ComputerID	*Type:* integer. Key. Nullable The computer id of the device to which the Action applies. Index into the Computer table.
DeviceID	*Type:* integer. Key. Nullable Index into the NetworkDevice table for this device.

Database Column	Details
`MACAddress`	*Type:* text (max 18 characters). Key. Nullable The network hardware address of the device.
`DNSName`	*Type:* text (max 128 characters). Key. Nullable The DNS name of the device.
`IPAddress`	*Type:* text (max 64 characters). Key. Nullable The IP Address of the device.
`ActionStateID`	*Type:* integer One of the action states defined in the `ActionState` table.
`FailureReason`	*Type:* text. Nullable If not empty, text describing the reason the action failed.
`LastUpdate`	*Type:* datetime The last time that the state of this action was updated. This value is the UTC date-time of the event.

ActionState Table

All possible states for an action are reflected in a record here.

Table 893: Database columns for `ActionState` table

Database Column	Details
`ActionStateID`	*Type:* integer. Key. Generated ID The id for the action state.

Database Column	Details
`ActionStateName`	*Type:* text (max 32 characters). Key The name for the action state. Possible id-name pairs are: • 1 = Created • 2 = DistributionInProgress • 3 = DistributionFailed • 4 = Distributed • 5 = SchedulePending • 6 = ScheduledFailed • 7 = Scheduled • 8 = Applied • 9 = ApplyFailed • 10 = CancelPending • 11 = CancelFailed • 12 = Cancelled • 13 = NotSupported

Job Table

This table stores the information about the jobs.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 894: Database columns for `Job` table

Database Column	Details
`JobUID`	*Type:* binary (max 16 bytes). Key The unique id for the job.
`TaskID`	*Type:* integer. Key The id for the task.
`ServerUID`	*Type:* binary (max 16 bytes). Key The unique id for the server.

Task Table

This table stores the information about the tasks.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 895: Database columns for `Task` table

Database Column	Details
`TaskID`	*Type:* integer. Key. Generated ID The id of the task.
`TaskUID`	*Type:* binary (max 16 bytes). Key. Nullable The id of the task.
`TaskTypeID`	*Type:* integer The id for the task type.
`TaskName`	*Type:* text (max 128 characters). Key The name for the task.
`PackagePathID`	*Type:* integer. Key. Nullable For a distribution task, which package.
`TaskScheduleID`	*Type:* integer The id for the task schedule.
`MinimumVersion`	*Type:* text (max 16 characters). Nullable The minimum version required to execute the task.

TaskSchedule Table

This table stores the required information about the task schedule, such as the start and finish times number of retries, delays and other related information.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database* `TenantID` *has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 896: Database columns for `TaskSchedule` table

Database Column	Details
`TaskScheduleID`	*Type:* integer. Key. Generated ID The id for the task schedule.
`StartTime`	*Type:* datetime. Nullable The time that the scheduled task must start.
`EndTime`	*Type:* datetime. Nullable The time that the scheduled task must end.
`RetryCount`	*Type:* integer. Nullable Number of times for task retries.
`MinRetryDelay`	*Type:* integer. Nullable Number of seconds before a retry occurs in case of a failure.
`RepeatDelay`	*Type:* integer. Nullable Number of seconds before the task is repeated.
`NumParallelTasks`	*Type:* integer. Nullable Number of tasks that can be run in parallel.
`SleepBetweenTasks`	*Type:* integer. Nullable Amount of time before the next task can start.

TaskType Table

This table stores the information about different types of tasks and their associated IDs.

Table 897: Database columns for `TaskType` table

Database Column	Details
`TaskTypeID`	*Type:* integer. Key. Generated ID The id for the task.
`TaskTypeName`	*Type:* text (max 32 characters). Key The name of the task.

7

License Portal Database Schema

This chapter describes additions made to the database schema for FlexNet Manager Suite to accommodate a separate licensing portal. With the entire product now presented in a web interface, this separation is entirely historical. The tables described in this chapter continue to appear in the database for all implementations.

Information Structure

The following information is provided about database tables. Items appear only when relevant to the database column, and are suppressed where they do not apply. Two of these items (shown bold) are columns in the following pages, and the remainder are displayed within the **Details**.

Item	Comment
Database Column	The name of the column in the SQL table.
Type	The data type of the contents of the database column.
Size	For types that have a maximum capacity, the upper limit is provided in parentheses.
Key	The word “Key” appears when a column is a unique key field within the table. It is possible for several database columns to be part of the key, so that this indicator may appear for several columns in a table.
Generated ID	This indicates that a numeric ID is assigned by the database.
Nullable	If this indicator is present, the database column permits nulls.
Computed	This indicator appears for columns that are automatically computed by the database.
Default	If a column has a default value declared in the schema, this is specified at the end of the first set of details for the column.
Details	Describes the data stored in the database column, including many of the indicators described above.

Compliance.ECM.Logic Tables

The complete set of database tables documented here includes:

- ComplianceActionHistory table (see ComplianceActionHistory Table)
- ComplianceActionHistoryResource table (see ComplianceActionHistoryResource Table)
- EcmSettings table (see EcmSettings Table)
- SoftwareLicenseUsageHistory table (see SoftwareLicenseUsageHistory Table)
- TrackGroup table (see TrackGroup Table)
- TrackSoftwareLicenseUsage table (see TrackSoftwareLicenseUsage Table)
- TrackSoftwareTitle table (see TrackSoftwareTitle Table)
- TrackSoftwareTitleUsage table (see TrackSoftwareTitleUsage Table)

ComplianceActionHistory Table

`ComplianceActionHistory` records actions performed in the Compliance portal on a contract or software license, including usage activation/deactivation.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 898: Database columns for `ComplianceActionHistory` table

Database Column	Details
`ComplianceActionHistoryID`	*Type:* integer. Key. Generated ID Unique identifier for the record.
`ComplianceAction HistoryResourceID`	*Type:* integer. Key Identifies the type of action performed. Foreign key to the `ComplianceActionHistoryResource` table.
`History`	*Type:* text Detailed information about the action performed.
`HistoryParameters`	*Type:* text Details of parameters changed and their changed values.
`AssociatedObjectID`	*Type:* integer The ID of the contract or license associated with the action.

Database Column	Details
`AssociatedObjectName`	*Type:* text (max 512 characters) The name of the contract or license associated with the action.
`Comment`	*Type:* text (max 1024 characters) Comments recorded about the change by the operator.
`CreationUser`	*Type:* text (max 512 characters) The username of the operator who made the change.
`CreationDate`	*Type:* datetime The date of the change.

ComplianceActionHistoryResource Table

`ComplianceActionHistoryResource` table stores string resources required by the `ComplianceActionHistory` table.

Table 899: Database columns for `ComplianceActionHistoryResource` table

Database Column	Details
`ComplianceAction HistoryResourceID`	*Type:* integer. Key. Generated ID Unique identifier for each record. Possible values and the corresponding default strings that may be written into a history list are: • 1 = Payment made • 2 = Payment edited • 3 = Payment cancelled • 4 = Activated application usage tracking for contract • 5 = Deactivated application usage tracking for contract • 6 = Activated application usage tracking for software license • 7 = Deactivated application usage tracking for software license • 8 = Modified application usage tracking for software license • 9 = Modified application usage tracking for contract • 10 = Not defined • 11 = Obligated to pay: (amount) • 12 = Actual amount was set to: (amount) • 13 = Actual amount currency rate was set to: (rate) • 14 = Estimated amount was set to: (amount) • 15 = Estimated amount currency rate was set to: (rate) • 16 = Budgeted amount was set to: (amount) • 17 = Budgeted amount currency rate was set to: (amount) • 18 = Payment status was set to: (status) • 19 = Payment amount: (amount); Payment date: (date) • 20 = Payment date was set to: (date) • 21 = Software license: (license name) • 22 = Software title: (application name) • 23 = Contract: (contract name) • 24 = Tracked: (yes/no); Track group: (group); Track start date: (date); Track end date: (date) • 25 = Applications tracked: (number).

Database Column	Details
ResourceName	*Type:* text (max 256 characters). Key The name of the resource that determines the text to display on the user interface.
DefaultValue	*Type:* text (max 512 characters) The default value to display if there is no resource string available to define the history action.

EcmSettings Table

EcmSettings stores operator-specific settings for the Compliance portal.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database TenantID has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 900: Database columns for EcmSettings table

Database Column	Details
EcmSettingID	*Type:* integer. Key. Generated ID A unique identifier for the record.
ComplianceOperatorID	*Type:* integer. Key An operator of the Compliance portal. Foreign key to the ComplianceOperator table.
SettingKey	*Type:* text (max 512 characters). Key A resource describing the operator setting.
SettingType	*Type:* text (max 512 characters) The data type of the operator setting.
SettingValueString	*Type:* text Serialized value of the operator setting.
LastUpdated	*Type:* datetime Date and time when this setting was last updated.

SoftwareLicenseUsageHistory Table

SoftwareLicenseUsageHistory records snapshots of software license utilization.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 901: Database columns for `SoftwareLicenseUsageHistory` table

Database Column	Details
`SoftwareLicenseUsageHistoryID`	*Type:* integer. Key. Generated ID A unique identifier for each record in this table.
`SnapshotDate`	*Type:* datetime Date that the snapshot was recorded and the projected usage was calculated.
`SoftwareLicenseID`	*Type:* integer. Key SoftwareLicenseID that identifies the software license. This field is a foreign key to the `SoftwareLicense` table.
`NumberPurchased`	*Type:* integer Total number of licenses purchased, as of the Snapshot Date.
`NumberInstalled`	*Type:* integer Total number of installations for the license, as of the Snapshot Date.
`NumberUsedActual`	*Type:* integer. Nullable Total consumption of the license, as of the Snapshot Date. If application usage is not being tracked, this field is blank.
`NumberUsedProjected`	*Type:* integer. Nullable The projected usage calculated for this license, based on patterns of usage over time.

TrackGroup Table

The `TrackGroup` table contains a list of the different tracking groups that tracked computer belong to.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 902: Database columns for `TrackGroup` table

Database Column	Details
`TrackGroupID`	*Type:* integer. Key. Generated ID A unique identifier for each `TrackGroup`. Possible values and the corresponding default strings are: • 1 = Sample • 2 = Enterprise.
`ResourceName`	*Type:* text (max 50 characters). Nullable The name of the resource that determines the text to display on the user interface.
`GroupName`	*Type:* text (max 64 characters). Key The default name of the `TrackGroup`. This is the value displayed if there is no resource string available to define the `TrackGroup`.

TrackSoftwareLicenseUsage Table

`TrackSoftwareLicenseUsage` keeps track of usage for each license.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 903: Database columns for `TrackSoftwareLicenseUsage` table

Database Column	Details
`TrackSoftwareLicenseUsageID`	*Type:* integer. Key. Generated ID Unique identifier for each record.
`SoftwareLicenseID`	*Type:* integer. Key Identifies a license. This field is a foreign key to the `SoftwareLicense` table.
`TrackGroupID`	*Type:* integer. Key. Nullable Identifies the track group associated with the license. This field is a foreign key to the `TrackGroup` table.
`SampleSize`	*Type:* integer. Nullable Number of computers in sample group.

Database Column	Details
`UsedPercentage`	*Type:* decimal. Nullable Percentage of computers within the tracking group that reported use of applications associated with this license.
`LastUpdated`	*Type:* datetime Date and time when software license usage was updated.

TrackSoftwareTitle Table

`TrackSoftwareTitle` stores details related to tracking software usage for a software title.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 904: Database columns for `TrackSoftwareTitle` table

Database Column	Details
`TrackSoftwareTitleID`	*Type:* integer. Key. Generated ID Unique identifier for each record. This field is a foreign key to the `SoftwareTitle` table.
`SoftwareTitleID`	*Type:* integer. Key. Nullable Identifies the application for which usage is being tracked. This field is a foreign key to the `SoftwareTitle` table.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable Identifies the license associated with the application. This field is a foreign key to the `SoftwareLicense` table.
`TrackGroupID`	*Type:* integer. Key Identifies if usage tracking has been activated for the Sample or Enterprise tracking group. This field is a foreign key to the `TrackGroup` table.
`LastTrackStartDate`	*Type:* datetime. Nullable Date that tracking was last turned on.
`LastTrackEndDate`	*Type:* datetime. Nullable Date that tracking was last turned off. This field may be null if the operator cleared the end date when activating application usage.

Database Column	Details
`TrackEndDueDate`	*Type:* datetime. Nullable Date that the current tracking period ends. Should be null when `IsTracked` is `False`.
`IsTracked`	*Type:* boolean. Key Indicates whether usage tracking is enabled for this application entry.

TrackSoftwareTitleUsage Table

`TrackSoftwareTitleUsage` keeps track of whether licensed software is being used on a computer.

Note: *To cater for multi-tenant mode, this table may contain data for multiple tenants. Access requires that the database `TenantID` has been set in the SQL Server connection context information. That setting filters an underlying table to produce this view of data for the single, selected tenant.*

Table 905: Database columns for `TrackSoftwareTitleUsage` table

Database Column	Details
`TrackSoftwareTitleUsageID`	*Type:* integer. Key. Generated ID Unique identifier for each record.
`ComplianceComputerID`	*Type:* integer. Key Identifies the computer on which usage tracking details were recorded. This field is a foreign key to the `ComplianceComputer` table.
`SoftwareTitleID`	*Type:* integer. Key Identifier for the application that was installed on the computer. This field is a foreign key to the `SoftwareTitle` table.
`SoftwareLicenseID`	*Type:* integer. Key. Nullable Identifier for the license associated with the installed application on the computer. This field is a foreign key to the `SoftwareLicense` table.
`TrackGroupID`	*Type:* integer. Key. Nullable Identifies the track group to which the computer has been assigned.
`IsUsed`	*Type:* boolean. Nullable Indicates whether the application is used on the computer.
`LastUsed`	*Type:* datetime. Nullable Date and time when software was last used on computer.

8

Inventory Spreadsheet Templates

In contrast with other chapters in this document, this chapter takes a different approach: rather than documenting the schema of the central database for FlexNet Manager Suite, it describes the formats acceptable for spreadsheet (`.xslx`) or comma-separated value (`.csv`) files that can be used to import various kinds of inventory information into the central database. For each data element, it shows which database table, and which column in that table, is the final destination for the imported data. (For details about importing inventory as spreadsheets or CSV files, see the chapter *Importing Inventory Spreadsheets and CSV Files* in the companion volume, *FlexNet Manager Suite System Reference*.)

Such spreadsheet (including CSV) files can be imported through two different paths:

- Using the web interface for FlexNet Manager Suite, the data may be uploaded directly to the central application server(s) as a one-time upload
- Optionally with a repeatable schedule, the data may also be uploaded through an inventory beacon.

The same templates are used for inventory imports through either of these channels.

Information Structure for Spreadsheet Inventory Imports

The following information is provided about the structure of spreadsheet (`.xslx`) and comma-separated value (`.csv`) template files that can be prepared as a data source for importing inventory. The items listed below appear only when relevant to the spreadsheet column, and are suppressed where they do not apply. Four of these items (shown bold) are columns in the following pages, and the remainder are displayed within the **Details** column.

Below this key is a mapping between:

- The file name of the downloaded template
- The prompt in the web interface of FlexNet Manager Suite for upload of the completed spreadsheet
- The topic below that covers this data (topic names are driven by the underlying database schema).

Remember: *The template files are fixed format. While adding data to each file, you may not change:*

- *The file name*

- *The names of columns*
- *The number of columns*
- *The order of columns.*

Item	Comment
Column	The name of the column in the spreadsheet template (and uploaded data file). ***Important:*** *Some column names are long, and must be wrapped over more than one line in this document. In all cases, the wrapped text should be continuous on a single line without white space in the template column names.*
Example values	Some sample data, or in some cases the list of supported values. When such a list is present, ensure that each row has a value that is an exact match for one of the available values (except that the validation is case insensitive).
Details	Describes the data required in the spreadsheet column, including many of the indicators described below.
Type	The data type of the contents of the spreadsheet column.
max	For types that have a maximum capacity, the upper limit is provided in parentheses.
Key	The word “Key” appears when a column is a unique key field for data matching between the row of the spreadsheet and the data in the central database table (the destination for the data). Keep in mind that a single spreadsheet may include data destined for multiple database tables; and even within a single database table, it is possible for several database columns to be part of the key. For these reasons, this indicator may appear in several rows in the documentation list.
Nullable	If this indicator is present, the spreadsheet column may be left blank (and the target database entity allows nulls). Be careful about spaces in a cell of your spreadsheet: white space is a valid value, and is not equivalent to a null.
Destination	Where the imported data is eventually saved in the central database for FlexNet Manager Suite. This is given with a dot separating the database table and the column name within the table, in the format `tableName.columnName`. For further details on these database tables and columns, see the other chapters in this volume. ***Tip:*** *A single value in the imported spreadsheet may update data in more than one database column. Where that happens, this* ***Destination*** *listing shows the multiple destinations for the individual row.*

Mapping templates to topics

The following table relates the template names (and the related prompts in the web interface) to the topics in this section that describe the individual columns within the templates. Templates are listed alphabetically. The naming of

the following topics is driven by the related table names in the underlying database schema, so this list helps map the real world presentation to the database.

Tip: *Templates are provided in matching pairs of XLSX and CSV files. As these are structurally identical, only the base file name (without an extension) is listed here.*

Template file name	Web prompt	See topic
`Cluster`	**Cluster evidence**	*ConsolidatedCluster Template*
`ClusterGroup`	**Cluster group data**	*ConsolidatedClusterGroup Template*
`ClusterHostAffinityRule`	**Cluster host affinity rule data**	*ConsolidatedClusterHostAffinityRule Template*
`Computer`	**Computers and VMs**	*ConsolidatedComputer Template*
`FileEvidence`	**File evidence**	*ConsolidatedFileEvidence Template*
`InstallerEvidence`	**Installation evidence**	*ConsolidatedInstallerEvidence Template*
`OracleDatabaseUser`	**Oracle Database user**	*ConsolidatedOracleDatabaseUser Template*
`RemoteAccessFile`	**Access shown by file evidence**	*ConsolidatedRemoteAccessFile Template*
`RemoteAccessInstaller`	**Access shown by installer evidence**	*ConsolidatedRemoteAccessInstaller Template*
`VMPool`	**Virtual machine pool data**	*ConsolidatedVMPool Template*
`WMIEvidence`	**WMI evidence**	*ConsolidatedWMIEvidence Template*

Compliance.InventoryReader.Logic Tables

The complete set of database tables documented here includes:

- ConsolidatedAccessEvidence table (see ConsolidatedAccessEvidence Template)
- ConsolidatedCluster table (see ConsolidatedCluster Template)
- ConsolidatedClusterGroup table (see ConsolidatedClusterGroup Template)
- ConsolidatedClusterHostAffinityRule table (see ConsolidatedClusterHostAffinityRule Template)
- ConsolidatedComputer table (see ConsolidatedComputer Template)
- ConsolidatedFileEvidence table (see ConsolidatedFileEvidence Template)
- ConsolidatedInstallerEvidence table (see ConsolidatedInstallerEvidence Template)
- ConsolidatedOracleDatabaseUser table (see ConsolidatedOracleDatabaseUser Template)

- ConsolidatedRemoteAccessFile table (see ConsolidatedRemoteAccessFile Template)
- ConsolidatedRemoteAccessInstaller table (see ConsolidatedRemoteAccessInstaller Template)
- ConsolidatedVMPool table (see ConsolidatedVMPool Template)
- ConsolidatedWMIEvidence table (see ConsolidatedWMIEvidence Template)

ConsolidatedAccessEvidence Template

ConsolidatedAccessEvidence provides a simpler interface to specify client access happening on application installed on server computers. It combines the server computer, and its access evidence details into a single row.

Table 906: Columns included with `ConsolidatedAccessEvidence` templates

Column	Details

`ComputerID`	*Type:* big integer. Key
	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
	Destination:
	`ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID`
	`ImportedClientAccessedAccessEvidence.ExternalServer ComputerID`
	`ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID`

`ProductName`	*Type:* text (max 256 characters). Key
	The product name of the software as reported by the access evidence.
	Destination:
	`ImportedClientAccessEvidence.ExternalAccessEvidenceID`
	`ImportedClientAccessEvidence.ProductName`
	`ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID`
	`ImportedClientAccessedAccessEvidence.ExternalAccess EvidenceID`
	`ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID`

Column	Details

`Version`	*Type:* text (max 72 characters). Key. Nullable
	The version of the software as reported by the access evidence.
	Destination:
	`ImportedClientAccessEvidence.ExternalAccessEvidenceID`
	`ImportedClientAccessEvidence.Version`
	`ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID`
	`ImportedClientAccessedAccessEvidence.ExternalAccess EvidenceID`
	`ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID`

`Edition`	*Type:* text (max 50 characters). Key. Nullable
	The edition of the software as reported by the access evidence.
	Destination:
	`ImportedClientAccessEvidence.ExternalAccessEvidenceID`
	`ImportedClientAccessEvidence.Edition`
	`ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID`
	`ImportedClientAccessedAccessEvidence.ExternalAccess EvidenceID`
	`ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID`

`AccessingDeviceIPAddress`	*Type:* text (max 256 characters). Key. Nullable
	IP Address of the accessing device.
	Destination:
	`ImportedAccessingDevice.ExternalAccessingDeviceID`
	`ImportedAccessingDevice.IPAddress`
	`ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID`
	`ImportedClientAccessedAccessEvidence.ExternalAccessing DeviceID`
	`ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID`

Column	Details
AccessingDevice ComputerName	*Type:* text (max 256 characters). Key. Nullable IP Address of the device accessing the product. Destination: ImportedAccessingDevice.ExternalAccessingDeviceID ImportedAccessingDevice.ComputerName ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID ImportedClientAccessedAccessEvidence.ExternalAccessing DeviceID ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID
AccessingDeviceSerialNo	*Type:* text (max 100 characters). Nullable Serial number of the device accessing the product. Destination: ImportedAccessingDevice.SerialNo
AccessingDeviceDomain	*Type:* text (max 100 characters). Key. Nullable Domain name of the device accessing the product. Destination: ImportedAccessingDevice.ExternalAccessingDeviceID ImportedAccessingDevice.Domain ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID ImportedClientAccessedAccessEvidence.ExternalAccessing DeviceID ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID

Column	Details

`AccessingUser`	*Type:* text (max 128 characters). Key. Nullable
	The DOMAIN/SAMAccountName of the user accessing the product.
	Destination:
	`ImportedAccessingUser.ExternalAccessingUserID`
	`ImportedAccessingUser.UserName (Element 2 after splitting on '\')`
	`ImportedAccessingUser.DomainName (Element 1 after splitting on '\')`
	`ImportedClientAccessedAccessEvidence.ImportedClient AccessedAccessEvidenceID`
	`ImportedClientAccessedAccessEvidence.ExternalAccessingUserID`
	`ImportedClientAccessedAccessOccurrence.ImportedClient AccessedAccessEvidenceID`

`AccessDate`	*Type:* datetime. Key. Nullable
	The date that the product was accessed. The date must be specified in the following format: 'yyyyMMdd'.
	Possible values:
	• yyyy/MM/dd
	• yyyy/MM/dd HH:mm:Ss
	• yyyy/MM/dd HH:mm
	• yyyy-MM-dd
	• yyyy-MM-dd HH:mm:Ss
	• yyyy-MM-dd HH:mm
	• yyyyMMdd
	• yyyyMMdd HH:mm:Ss
	• yyyyMMdd HH:mm
	Destination:
	`ImportedClientAccessedAccessOccurrence.AccessDate`
	`ImportedClientAccessedAccessOccurrence.LicenseDate`

`AccessCount`	*Type:* integer. Nullable Number of times the product was accessed on the given access date. Destination: `ImportedClientAccessedAccessOccurrence.AccessCount`

Column	Details
`InventoryDate`	*Type:* datetime. Nullable The date (and optionally time) the access evidence record was inventoried. Possible values: • yyyy/MM/dd • yyyy/MM/dd HH:mm:Ss • yyyy/MM/dd HH:mm • yyyy-MM-dd • yyyy-MM-dd HH:mm:Ss • yyyy-MM-dd HH:mm • yyyyMMdd • yyyyMMdd HH:mm:Ss • yyyyMMdd HH:mm Destination: `ImportedClientAccessedAccessOccurrence.InventoryDate`
`ClientAccessSource`	*Type:* text (max 100 characters). Nullable The source type of the access evidence. Destination: `ImportedClientAccessedAccessEvidence.ClientAccessSource`

ConsolidatedCluster Template

The Cluster spreadsheet provides a simple interface for defining server clustering. It is useful when combined with the ClusterGroup and ClusterHostAffinityRule spreadsheets.

Table 907: Columns included with `ConsolidatedCluster` templates

Column	Details
`ClusterID`	*Type:* big integer. Key The unique identifier for this imported cluster. This may be a string or an integer. Destination: `ImportedCluster.ExternalID`

Column	Details

`ClusterName`	*Type:* text (max 128 characters)
	The name of the cluster in the external cluster management system.
	Destination:
	`ImportedCluster.ExternalName`
	`ImportedCluster.Name`

`Namespace`	*Type:* text (max 256 characters). Nullable Where the cluster is contained: + The fully-qualified domain name (for HyperV clusters) - example: 'france.thc.myenterprise.com' + The datacenter name (for VMWare clusters) - example: 'MelProdDataCenter' Destination: `ImportedCluster.Namespace`
`ClusterType`	*Type:* text (max 128 characters) The kind of cluster. The value must be an exact case-insensitive match to one of the permitted values. Possible values: • vMotion Cluster • Hyper-V Cluster • Host Affinity Group • VM Affinity Group • Oracle VM Destination: `ImportedCluster.ClusterTypeID`

Column	Details
`InventoryDate`	*Type:* datetime. Nullable The date (with optional time) that the cluster last had inventory reported. The date must be entered in one of the supported formats. Possible values: • yyyy/MM/dd • yyyy/MM/dd HH:mm:Ss • yyyy/MM/dd HH:mm • yyyy-MM-dd • yyyy-MM-dd HH:mm:Ss • yyyy-MM-dd HH:mm • yyyyMMdd • yyyyMMdd HH:mm:Ss • yyyyMMdd HH:mm Destination: `ImportedCluster.InventoryDate`
`InventoryAgent`	*Type:* text (max 64 characters). Nullable The name of the person or tool that performed the last inventory. For imported spreadsheets, you may wish to include the name of the person preparing the data, in case there is subsequent follow-up required. Destination: `ImportedCluster.InventoryAgent`
`DRS`	*Type:* boolean. Nullable Whether Distributed Resource Scheduler (DRS) is enabled on the cluster. Possible values: true, false, 0 or 1 Destination: `ImportedCluster.DRS`
`DPM`	*Type:* boolean. Nullable Whether Distributed Power Management (DPM) is enabled on the cluster. Possible values: true, false, 0 or 1 Destination: `ImportedCluster.DPM`

ConsolidatedClusterGroup Template

The ClusterGroup spreadsheet uses data from the Cluster spreadsheet and defines groups of servers as well as computers that are members of these groups.

Table 908: Columns included with `ConsolidatedClusterGroup` templates

Column	Details
`ClusterID`	*Type:* big integer. Key The unique identifier for the imported cluster. This may be a string or an integer and must match a value for the ClusterID in the cluster spreadsheet. Destination: `ImportedClusterGroup.ClusterExternalID`

`ClusterGroupID`	*Type:* big integer. Key
	The unique identifier for this cluster group. This may be a string or an integer.
	Destination:
	`ImportedClusterGroup.ExternalID`
	`ImportedClusterGroupMember.ClusterGroupExternalID`

`ClusterGroupName`	*Type:* text (max 128 characters). Nullable The name of the cluster group. Depending on the value of the ClusterGroupType this will be a group of hosts or virtual machines. Destination: `ImportedClusterGroup.Name`
`ClusterGroupType`	*Type:* text (max 128 characters) The kind of cluster included in the group. The value must be an exact case-insensitive match to one of the permitted values. Possible values: • vMotion Cluster • Hyper-V Cluster • Host Affinity Group • VM Affinity Group • Oracle VM Destination: `ImportedClusterGroup.ClusterTypeID`

Column	Details
`ComputerID`	*Type:* big integer. Key The identifier used in the 'Computer' spreadsheet for a computer which is a member of the group. To identify all the members of the group, repeat as many lines as required in your spreadsheet where the other values in the row are identical, and only the 'ComputerID' value changes. Values in this column must match a ComputerID in the computer spreadsheet or the row will be skipped. Destination: `ImportedClusterGroupMember.ComputerExternalID`

ConsolidatedClusterHostAffinityRule Template

The ClusterHostAffinity spreadsheet defines the groups of virtual machines which may run on groups of host servers.

Table 909: Columns included with `ConsolidatedClusterHostAffinityRule` templates

Column	Details
`ClusterID`	*Type:* big integer. Key The unique identifier for the imported cluster, to which this affinity rule applies. This may be a string or an integer and must match a ClusterID from the cluster spreadsheet. Destination: `ImportedClusterHostAffinityRule.ClusterExternalID`
`Name`	*Type:* text (max 128 characters). Nullable The name of the cluster host affinity rule. Destination: `ImportedClusterHostAffinityRule.Name`
`ClusterHostGroupName`	*Type:* big integer. Key The name of the group of hosts that the ClusterVMGroupName virtual machines may run on. Destination: `ImportedClusterHostAffinityRule.ClusterHostGroupExternalID`
`ClusterVMGroupName`	*Type:* big integer. Key The name of the virtual machine group that may run on the ClusterHostGroupName hosts. Destination: `ImportedClusterHostAffinityRule.ClusterVMGroupExternalID`

Column	Details
`ClusterHostAffinity RuleType`	*Type:* text (max 128 characters) The type of affinity rule. The value must be an exact case-insensitive match to one of the permitted values. Possible values: • must run on • must not run on Destination: `ImportedClusterHostAffinityRule.ClusterHostAffinityRule TypeID`

ConsolidatedComputer Template

'ConsolidatedComputer' consolidates data for the Computer, VirtualMachine, Domain, User and Cluster objects, providing a simpler way to populate this information. Any spreadsheet row that includes a 'HostComputerID' is making that row a virtual machine, and the import process expects that virtualization data will be provided.

Table 910: Columns included with `ConsolidatedComputer` templates

Column	Details

`ComputerID`	*Type:* big integer. Key
	The unique identifier for a computer (either physical or virtual). This identifier can either be an integer or a string. Keep this consistent across multiple imports: it is used to track the computer over time.
	Destination:
	`ImportedComputer.ExternalID`
	`ImportedVirtualMachine.VMComputerID`
	`ImportedClusterNode.ComputerExternalID`
	`ImportedCloudServiceInstance.ExternalComputerID`

`ComputerName`	*Type:* text (max 256 characters) The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by `GetComputerName()`. For UNIX, it is the host name of the machine, as returned by `gethostname(2)`. Destination: `ImportedComputer.ComputerName`

Column	Details
`DomainFlatName`	*Type:* text (max 100 characters). Key. Nullable The flatname of the domain of the computer. Example: ‘mycompany’. Destination: `ImportedDomain.FlatName`

`DomainQualifiedName`	*Type:* text (max 100 characters). Key. Nullable
	The fully qualified domain name for the computer. Example: ‘prod.mycompany.eu’.
	Destination:
	`ImportedComputer.Domain`
	`ImportedDomain.QualifiedName`

`BIOSUUID`	*Type:* unique identifier. Nullable The BIOS UUID of the computer (physical or virtual), as provided by the operating system. Possible values: 93B5BE3B-88B0-450E-9F75-F6294210DFA0 Destination: `ImportedComputer.UUID`

`OperatingSystem`	*Type:* text (max 128 characters). Nullable
	The operating system of the computer. For virtual machines, it is the configured operating system of the guest. Note that this operating system identification is not used for licensing.
	Destination:
	`ImportedComputer.OperatingSystem`
	`ImportedVirtualMachine.GuestFullName`

`ServicePack`	*Type:* text (max 128 characters). Nullable The service pack installed for the operating system. Destination: `ImportedComputer.ServicePack`
`EmailAddress`	*Type:* text (max 256 characters). Nullable The email address associated with the device. Typically used for mobile devices. Destination: `ImportedComputer.EmailAddress`
`PhoneNumber`	*Type:* text (max 128 characters). Nullable The phone number of the device. Used for mobile devices. Destination: `ImportedComputer.PhoneNumber`

Column	Details

`Manufacturer`	*Type:* text (max 128 characters). Nullable
	The manufacturer of the computer hardware. Some examples include:
	• On Windows, the SMBios manufacturer (the WMI Manufacturer property of the 'Win32_ComputerSystem' class).
	• On Linux, 'Manufacturer' in the 'System Information' section resulting from the 'dmidecode' command. Sample command: 'dmidecode -s system-manufacturer'
	• On Solaris x86, as for Linux, with failovers first to 'sysinfo SI_HW_PROVIDER' and then to 'ModelNo'.
	• On Solaris SPARC, the 'sysinfo SI_HW_PROVIDER'. Typically this value is 'Sun_Microsystems' or, more recently, 'Oracle Corporation'. Failover to the 'ModelNo'.
	• On HP-UX, the string literal 'HP'.
	• On AIX, the 'modelname' system attribute preceding the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use 'IBM'. This value is typically 'IBM'.
	Destination:
	`ImportedComputer.Manufacturer`
	`ImportedVirtualMachine.Manufacturer`

Column	Details

`ModelNo`	*Type:* text (max 128 characters). Nullable
	The model of the computer hardware or the virtual machine. This value is defined for the context of the current execution environment, rather than the physical server that may be hosting a virtual machine or partition. Examples:
	• On Windows, the SMBios product name. The WMI Model property of the Win32_ComputerSystem class.
	• On Linux, the SMBios product name read using the command 'dmidecode -s system-product-name'. Specifically, the 'System Information' section and the 'Product Name' in that section is used.
	• On Solaris x86, as for Linux, with failover to the 'sysinfo SI_PLATFORM', stripping 'SUNW', and replacing hyphen characters with space characters.
	• On Solaris SPARC, the 'openprom' "banner-name" value read from '/dev/openprom'. Failover to the 'sysinfo SI_PLATFORM', stripping 'SUNW', and replacing hyphen characters with space characters.
	• On HP-UX, the 'confstr _CS_MACHINE_MODEL'.
	• On AIX, the 'modelname' system attribute following the comma character. For example, if the 'modelname' system attribute is 'IBM,8202-E4B', then use '8202-E4B'.
	Destination:
	`ImportedComputer.ModelNo`
	`ImportedVirtualMachine.ModelNo`

Column	Details
`SerialNo`	*Type:* text (max 100 characters). Nullable The hardware serial number of the computer. The goal of this value is to be tied to the physical hardware, partition or virtual machine and to be as unique as possible across all computers in the organization. This is due to its use in tracking computers, particularly after an operating system rebuild. This value is also used to socialize computer inventory from different inventory sources, and is used to map virtual machine guest operating system inventory to the VM host on which the virtual machine is running. Example sources: • On Windows, the SMBios serial number. The WMI 'SerialNumber' property of the 'Win32_BIOS' class. Can fail over to the 'SerialNumber' property of the 'Win32_SystemEnclosure' class which is typically the same value. • On Linux, the SMBios serial number read using the command 'dmidecode -s system-serial-number'. Specifically, the 'System Information' section and the 'Serial Number' in that section is used. • On Solaris 10 8/07 or later, for a non-global zone, the UUID value from the /etc/zones/index file. For a global zone, the same as Solaris 10 releases earlier than 8/07. • For Solaris 10 releases earlier than 8/07, the hexadecimal version of 'SI_HW_SERIAL' with an appended hyphen character followed by the Zone's name. For example, '838bfc7b-global' or '838bfc7b-myzone'. • For Solaris 8 and 9, The hexadecimal version of 'SI_HW_SERIAL'. • For Mac OS X, the serial number of the machine as printed on the packaging and found in "About this Mac" from the desktop. • For HP-UX, the 'confstr _CS_PARTITION_IDENT' partition identifier if it is an nPar or vPar, or '_CS_MACHINE_IDENT' if not; with a failover to the machine serial number, and a final failover to the 'uname' machine identification number. • For AIX, the 'id_to_partition' system attribute, starting from the third character (strips a '0X' from the start). For example, if the 'id_to_partition' system attribute is '0X0473409002F7B201' then use '0473409002F7B201'. Destination: `ImportedComputer.SerialNo`

Column	Details
`ChassisType`	*Type:* text (max 128 characters). Nullable The type of case of the computer. The value must be a (case insensitive) exact match for one of the values shown. Note that some license types use this information to optimize the licensing position, particularly with desktop and laptop computers. Destination: `ImportedComputer.ChassisType`
`TotalMemory`	*Type:* big integer. Nullable The total RAM in the computer, in bytes. Destination: `ImportedComputer.TotalMemory`
`NumberOfDisplayAdapters`	*Type:* integer. Nullable The number of graphics cards in the computer. Destination: `ImportedComputer.NumberOfDisplayAdapters`
`VirtualMachineUUID`	*Type:* text (max 256 characters). Nullable The unique identifier of the virtual machine provided by the virtualization infrastructure. (This may have the same value as the 'BIOSUUID', or have byte order reversed, or be altogether different.) Destination: `ImportedVirtualMachine.UUID`
`IMEI`	*Type:* text (max 256 characters). Nullable IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types. Destination: `ImportedComputer.IMEI`

`NumberOfProcessors`	*Type:* integer. Nullable
	The total number of physical processors (CPU) in the computer. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
	Destination:
	`ImportedComputer.NumberOfProcessors`
	`ImportedVirtualMachine.NumberOfProcessors`

Column	Details

`ProcessorType`	*Type:* text (max 256 characters). Nullable
	The descriptive string of the processor(s) in the computer. This may be a comma-separated list in the case where there is more than one physical processor in the system. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
	Destination:
	`ImportedComputer.ProcessorType`
	`ImportedVirtualMachine.ProcessorType`

`MaxClockSpeed`	*Type:* integer. Nullable The maximum clock speed of the fastest processor in the computer in megahertz. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position. Destination: `ImportedComputer.MaxClockSpeed`

`NumberOfCores`	*Type:* integer. Nullable
	The total number of cores in the computer. If there is more than one physical processor in the computer, then this would be the sum of the core counts for all the processors. For example, in a computer with two quad-core processors, this value would be 8. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
	Destination:
	`ImportedComputer.NumberOfCores`
	`ImportedCloudServiceInstance.CoreCount`

`NumberOfSockets`	*Type:* integer. Nullable The number of physical sockets into which a processor may be placed in the computer. It is rare that an inventory source can know this value. If unset, it is typically approximated by the number of processors. Destination: `ImportedComputer.NumberOfSockets`

Column	Details
`NumberOfLogicalProcessors`	*Type:* integer. Nullable The number of logical processors in the computer. This is the number of 'execution contexts' the operating system has access to. It will commonly be equivalent to the number processors in a single core, non-multi-threaded processor architecture, to the number of cores in a multi-core single threaded processor architecture, and to the number of threads in a multi-threaded processor architecture. For example, in a two processor, quad-core and hyper-threaded computer, this value would be 16. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position. Destination: `ImportedComputer.NumberOfLogicalProcessors`
`PartialNumberOfProcessors`	*Type:* decimal. Nullable Used in processor-based licensing, this is the non-integer number of cores allocated to this partition or virtual machine. When this property is null, the 'NumberOfCores' is used. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position. Possible values: 120.45 Destination: `ImportedComputer.PartialNumberOfProcessors`

`NumberOfHardDrives`	*Type:* integer. Nullable
	The number of physical hard drives in the computer. While the intent is physical drives, often this can end up being the number of disk partitions.
	Destination:
	`ImportedComputer.NumberOfHardDrives`
	`ImportedVirtualMachine.NumberOfHardDrives`

`TotalDiskSpace`	*Type:* big integer. Nullable The total size of all hard drives in the computer in bytes. Note that this can be a very large number on modern systems. The maximum value for a bigint is 9,223,372,036,854,775,807, which can represent about 9.2 exabyte. While in practice it is unlikely that this size of storage capacity is reached for a single system, some systems can end up with large values through virtualized drives. Therefore, it is worth considering capping values when calculating total disk space, particularly when converting values from kilobytes or megabytes to bytes. Destination: `ImportedComputer.TotalDiskSpace`

Column	Details

`NumberOfNetworkCards`	*Type:* integer. Nullable
	The number of network cards in the computer.
	Destination:
	`ImportedComputer.NumberOfNetworkCards`
	`ImportedVirtualMachine.NumberOfNetworkCards`

`IPAddress`	*Type:* text (max 256 characters). Nullable The IP address of the computer in IPv4 or IPv6 format. This may be a comma-separated list if there is more than one active network adapter in the system. Do not include inactive network adapters and network adapters with invalid IP addresses. Examples: • ‘69.89.31.226’ • ‘2002:4559:1FE2::4559:1FE2’ Destination: `ImportedComputer.IPAddress`

`MACAddress`	*Type:* text (max 256 characters). Nullable
	The MAC address of the computer. This may be a comma-separated list if there is more than one active network adapter in the system. Do not include inactive network adapters and network adapters with invalid MAC addresses.
	Destination:
	`ImportedComputer.MACAddress`
	`ImportedCloudServiceInstance.MACAddress`

`LastLoggedOnUser`	*Type:* text (max 128 characters). Key. Nullable
	The DOMAIN/SAMAccountName of the user last logged onto the computer.
	Destination:
	`ImportedComputer.LastLoggedOnUser`
	`ImportedUser.ExternalID`
	`ImportedUser.UserName (Element 2 after splitting on '\')`
	`ImportedUser.Domain (Element 1 after splitting on '\')`
	`ImportedUser.SAMAccountName (Element 2 after splitting on '\')`

Column	Details
`LastLogonDate`	*Type:* datetime. Nullable The date and time when the user last logged on to the computer. The date must be entered in one of the supported formats. Possible values: • yyyy/MM/dd • yyyy/MM/dd HH:mm:Ss • yyyy/MM/dd HH:mm • yyyy-MM-dd • yyyy-MM-dd HH:mm:Ss • yyyy-MM-dd HH:mm • yyyyMMdd • yyyyMMdd HH:mm:Ss • yyyyMMdd HH:mm Destination:
`CalculatedUser`	*Type:* text (max 128 characters). Nullable The domain/SAMAccountName of the calculated user. Some inventory systems calculate the user who owns a computer. For example, it might be the user who, over the last ten logins, logged in most often. Destination: `ImportedComputer.CalculatedUser`

`HostComputerID`	*Type:* text (max 256 characters). Key. Nullable
	The ComputerID of the server this virtual machine is hosted on. This may be a string or an integer and must match the ComputerID for another computer in this spreadsheet.
	Destination:
	`ImportedVirtualMachine.HostComputerID`
	`ImportedCloudServiceInstance.HostComputerID`

Column	Details
`VirtualMachineType`	*Type:* text (max 100 characters). Nullable The type of the virtual machine. If present, the value must be a (case insensitive) exact match to one of the values shown. Possible values: • VMware • HyperV • LPAR • WPAR • nPar • vPar • SRP • Zone • Unknown • Oracle VM • AWS EC2 • Linux KVM Destination: `ImportedVirtualMachine.VirtualMachineType`

`VMEnabledState`	*Type:* text (max 128 characters). Nullable
	The operational state of the virtual machine. If present, the value must be a (case insensitive) exact match to one of the values shown.
	Possible values:
	• Started
	• Stopped
	• Suspended
	• Unknown
	• Terminated
	Destination:
	`ImportedVirtualMachine.VMEnabledStateID`
	`ImportedCloudServiceInstance.VMEnabledStateID`

Column	Details
`AffinityEnabled`	*Type:* boolean. Nullable Set this to `true` (or `1`) if this VM has affinity for its current host (so that it is unable to move to different host computers). Leave blank (null) for a physical computer. Possible values: true, false, 0 or 1 Destination: `ImportedVirtualMachine.AffinityEnabled`
`CPUAffinity`	*Type:* text (max 256 characters). Nullable Contains a comma-separated list of processor numbers (Host Logical Processors) or ranges for which this virtual machine has affinity. Example: `1,3-5,8` Destination: `ImportedVirtualMachine.CPUAffinity`
`CoreAffinity`	*Type:* text (max 256 characters). Nullable Contains a comma-separated list of core numbers (or ranges) for which this virtual machine has affinity. Cores are numbered sequentially up the sequence of processors. Example: `1,5-8,10` Destination: `ImportedVirtualMachine.CoreAffinity`

Column	Details
`ComplianceComputerType`	*Type:* text (max 128 characters). Nullable If you know that the computer is a virtual machine or VM host, record that data here. If you are unsure, leave this cell empty (NULL): this allows the system to infer the computer type (for example, a computer with VMs linked to it is inferred to be a VM host). If data comes from multiple inventory sources, leaving this value as null also allows the value to be inserted from another source. So, unless there is a very good reason, do not just specify 'Computer', but allow the inference rules to help. Possible values: • Computer • VM Host • Virtual Machine • Remote Device • Mobile Device • VDI Template Destination: `ImportedComputer.ComplianceComputerTypeID`
`HostIdentifyingNumber`	*Type:* text (max 128 characters). Nullable A physical server may have an identifier that is unique only across that hardware model, and may be less unique than the true hardware serial number, for example. This value is typically set for physical machines only, which include virtualization hosts, partitioned server hosts, and standalone machines. For a partitioned server, this value can be reported by each of the partitions on that server, such that a record of the physical computer can be created using one of the instances of this value. This value is used for matching computers. Destination:

Column	Details
`HostType`	*Type:* text (max 128 characters). Nullable The type of the physical host computer. This value is similar to the model number, but it is always for the physical server that an execution context may be running on. Therefore, this will generally be a known value for standalone machines and partitions, but it will not be known for virtual machines. This value is used for matching computers. Examples: • 'i86pc' • 'Sun-Fire-T1000' • 'rx7620' • '785' (for a 9000/785/C3700) • '8202' (for an IBM,8202-E4B). Destination:
`VMLocation`	*Type:* text (max 256 characters). Nullable Location of the virtual machine on the file system. Destination: `ImportedVirtualMachine.VMLocation`
`PoolName`	*Type:* text (max 100 characters). Nullable The name of the pool that the virtual machine belongs to. Destination: `ImportedVirtualMachine.PoolName`

Column	Details
`PoolType`	*Type:* text (max 100 characters). Nullable The type of the pool that the virtual machine belongs to. Possible values: • Folder • Datacenter • ComputeResource • HostSystem • ResourcePool • VirtualMachine • PhysicalSharedPool • VirtualSharedPool • LPAR • RSET • ClusterComputeResource • PSET Destination: `ImportedVirtualMachine.PoolType`
`CPUUsage`	*Type:* integer. Nullable The maximum CPU usage of the virtual machine (MHz). Destination: `ImportedVirtualMachine.CPUUsage`
`MemoryUsage`	*Type:* big integer. Nullable The maximum memory usage of the virtual machine (bytes). Destination: `ImportedVirtualMachine.MemoryUsage`

Column	Details

`InventoryDate`	*Type:* datetime. Nullable
	The date (and optionally time) the computer last had inventory reported. This field is generally used for differential updates (that is, if the date/time has not changed since the previous import, the data record is not imported/updated). The date must be entered in one of the supported formats.
	Possible values:
	• yyyy/MM/dd
	• yyyy/MM/dd HH:mm:Ss
	• yyyy/MM/dd HH:mm
	• yyyy-MM-dd
	• yyyy-MM-dd HH:mm:Ss
	• yyyy-MM-dd HH:mm
	• yyyyMMdd
	• yyyyMMdd HH:mm:Ss
	• yyyyMMdd HH:mm
	Destination:
	`ImportedComputer.InventoryDate`
	`ImportedCloudServiceInstance.InventoryDate`

`ClusterID`	*Type:* big integer. Key. Nullable The unique identifier for the cluster containing this computer. This must match the ClusterID used in the Cluster spreadsheet. If both the ClusterID and the ClusterNodeType do not match the data provided in the Cluster spreadsheet then the computer will not be associated with a cluster. Destination: `ImportedClusterNode.ClusterExternalID`

Column	Details
`ClusterNodeType`	*Type:* text (max 128 characters). Nullable The Cluster node type of the computer. Must be a (case insentitive) exact match for one of the values shown. If both the ClusterID and the ClusterNodeType do not match the data provided in the Cluster spreadsheet then the computer will not be associated with a cluster. Possible values: • Active • Passive • Hot • Warm • Cold Destination: `ImportedClusterNode.ClusterNodeTypeID`

`HostID`	*Type:* text (max 100 characters). Nullable
	The HostID hardware property for the server hosting this machine partition (when inventorying a machine partition such as Solaris Zone, AIX lPar, HP-UX nPar/vPar).
	Destination:
	`ImportedComputer.HostID`
	`ImportedCloudServiceInstance.HostID`

`FirmwareSerialNumber`	*Type:* text (max 100 characters). Nullable The Serial number in the system firmware such as BIOS, EEPROM etc. Destination: `ImportedComputer.FirmwareSerialNumber`
`MachineID`	*Type:* text (max 100 characters). Nullable For AIX, it is the System ID. For HP-UX, it is the Machine/Software ID. It is unset for other platforms. Destination: `ImportedComputer.MachineID`
`InstanceCloudID`	*Type:* text (max 256 characters). Key. Nullable The ID of the cloud instance. Destination: `ImportedCloudServiceInstance.InstanceCloudID`

Column	Details
`CloudServiceProvider`	*Type:* text (max 256 characters). Key. Nullable A unique identifier for a cloud service provider record. Destination: `ImportedCloudServiceInstance.CloudServiceProvider`
`InstanceAffinity`	*Type:* text (max 256 characters). Nullable The affinity setting for the instance on the Dedicated Host. Destination: `ImportedCloudServiceInstance.InstanceAffinity`
`ImageID`	*Type:* text (max 256 characters). Nullable The ID of the image used to launch the instance. Destination: `ImportedCloudServiceInstance.ImageID`
`LaunchTime`	*Type:* datetime. Nullable The time the cloud instance was launched or the Reserved Instance started. Possible values: • yyyy/MM/dd • yyyy/MM/dd HH:mm:Ss • yyyy/MM/dd HH:mm • yyyy-MM-dd • yyyy-MM-dd HH:mm:Ss • yyyy-MM-dd HH:mm • yyyyMMdd • yyyyMMdd HH:mm:Ss • yyyyMMdd HH:mm Destination: `ImportedCloudServiceInstance.LaunchTime`
`NetworkID`	*Type:* text (max 256 characters). Nullable The ID of the Virtual Private Cloud. Destination: `ImportedCloudServiceInstance.NetworkID`

Column	Details
`LifecycleMode`	*Type:* text (max 256 characters). Nullable The time the instance was launched. Destination: `ImportedCloudServiceInstance.LifecycleMode`
`Account`	*Type:* text (max 256 characters). Nullable The Account that is used to create the instance. Destination: `ImportedCloudServiceInstance.Account`
`ThreadsPerCore`	*Type:* integer. Nullable The number of thread per core of the instance. Destination: `ImportedCloudServiceInstance.ThreadsPerCore`
`InstanceType`	*Type:* text (max 256 characters). Nullable Cloud provider instance type. Destination: `ImportedCloudServiceInstance.InstanceType`
`Region`	*Type:* text (max 256 characters). Nullable Region of the instance or host. Destination: `ImportedCloudServiceInstance.Region`
`AvailabilityZone`	*Type:* text (max 256 characters). Nullable Location of the instance or host. Destination: `ImportedCloudServiceInstance.AvailabilityZone`
`InstanceTenancy`	*Type:* text (max 256 characters). Nullable Instance tenancy of the instance or host Destination: `ImportedCloudServiceInstance.InstanceTenancy`

ConsolidatedFileEvidence Template

ConsolidatedFileEvidence provides a simpler interface to specify files and their usage on computers. It combines the computer, file evidence and usage details into a single row.

Table 911: Columns included with `ConsolidatedFileEvidence` templates

Column	Details

`ComputerID`	*Type:* big integer. Key
	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
	Destination:
	`ImportedInstalledFileEvidence.ExternalID`
	`ImportedInstalledFileEvidenceUsage.ExternalID`

`FileName`	*Type:* text (max 256 characters). Key
	The name of the file used as evidence of software installation. For unix operating systems include the full path in the file name, including the opening '/'. For Windows operating systems the file path is specified in the FilePath column and this column must only contain the file name.
	Destination:
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.FileName`
	`ImportedInstalledFileEvidence.ExternalFileID`
	`ImportedInstalledFileEvidenceUsage.ExternalFileID`

`FileVersion`	*Type:* text (max 100 characters). Key. Nullable
	The version number of the file used as evidence of software installation.
	Destination:
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.FileVersion`
	`ImportedInstalledFileEvidence.ExternalFileID`
	`ImportedInstalledFileEvidenceUsage.ExternalFileID`

`ProductVersion`	*Type:* text (max 200 characters). Nullable The product version number in the file header. Destination: `ImportedFileEvidence.ProductVersion`
`ProductName`	*Type:* text (max 200 characters). Nullable The product name in the file header. Destination: `ImportedFileEvidence.ProductName`

Column	Details
`FilePath`	*Type:* text (max 400 characters). Nullable The path of the file used as evidence of software installation. Destination: `ImportedFileEvidence.FilePath`

`Company`	*Type:* text (max 100 characters). Key. Nullable
	The company in the file header.
	Destination:
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.Company`
	`ImportedInstalledFileEvidence.ExternalFileID`
	`ImportedInstalledFileEvidenceUsage.ExternalFileID`

`Description`	*Type:* text (max 200 characters). Key. Nullable
	The description in the file header.
	Destination:
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.Description`
	`ImportedInstalledFileEvidence.ExternalFileID`
	`ImportedInstalledFileEvidenceUsage.ExternalFileID`

`FileSize`	*Type:* integer. Key. Nullable
	The size of the file in bytes.
	Destination:
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.FileSize`
	`ImportedInstalledFileEvidence.ExternalFileID`
	`ImportedInstalledFileEvidenceUsage.ExternalFileID`

`Language`	*Type:* text (max 200 characters). Nullable The language in the file header. Destination: `ImportedFileEvidence.Language`

Column	Details

`AccessMode`	*Type:* text (max 128 characters). Key. Nullable
	The access mode of the file evidence. Leave this blank unless this row is a virtualized application. In that case choose one of the values below that matches your application or desktop virtualization infrastructure.
	Possible values:
	• Local
	• App-V
	• XenApp
	• XenDesktop
	• VMware View
	• Office 365
	Destination:
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.AccessModeID`
	`ImportedInstalledFileEvidence.ExternalFileID`
	`ImportedInstalledFileEvidenceUsage.ExternalFileID`

`NumberOfSessions`	*Type:* big integer. Nullable The number of sessions that the file evidence was in use by the user specified in the UserID column during the usage tracking period. If multiple users used the same application on the computer, create one row for each user with usage. Destination: `ImportedInstalledFileEvidenceUsage.NumberOfSessions`
`StartDate`	*Type:* text (max 10 characters). Nullable The start date of the usage. The date must be specified in the following format: 'yyyyMMdd'. Destination: `ImportedInstalledFileEvidenceUsage.StartDate`
`LastUsedDate`	*Type:* text (max 10 characters). Nullable The last used date of the usage. The date must be specified in the following format: 'yyyyMMdd'. Destination: `ImportedInstalledFileEvidenceUsage.LastUsedDate`

Column	Details

`UserID`	*Type:* big integer. Key. Nullable
	The DOMAIN/SAMAccountName for the user that the file evidence was used by. If this software was used by multiple users, create one row for each user of the software on the computer.
	Destination:
	`ImportedInstalledFileEvidenceUsage.ExternalUserID`
	`ImportedUser.ExternalID`
	`ImportedUser.UserName (Element 2 after splitting on '\')`
	`ImportedUser.Domain (Element 1 after splitting on '\')`
	`ImportedUser.SAMAccountName (Element 2 after splitting on '\')`

ConsolidatedInstallerEvidence Template

ConsolidatedInstallerEvidence provides a simpler interface to specify installed applications and their usage on computers. It combines the computer, installer evidence and usage details into a single row.

Table 912: Columns included with `ConsolidatedInstallerEvidence` templates

Column	Details

`ComputerID`	*Type:* big integer. Key
	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
	Destination:
	`ImportedInstalledInstallerEvidence.ExternalComputerID`
	`ImportedInstalledInstallerEvidenceUsage.ExternalID`
	`ImportedInstance.ExternalComputerID`

`InstanceName`	*Type:* big integer. Key. Nullable
	If this installer evidence is an Oracle database, then this field specifies the name of the database instance. If there are multiple instances, create a row for each instance in this spreadsheet.
	Destination:
	`ImportedInstalledInstallerEvidence.ExternalInstanceID`
	`ImportedInstalledInstallerEvidenceUsage.ExternalInstanceID`
	`ImportedInstance.InstanceID`
	`ImportedInstance.ParentInstanceID`
	`ImportedInstance.InstanceName`

Column	Details

`DisplayName`	*Type:* text (max 256 characters). Key
	The display name of the software as reported by the installer evidence.
	Destination:
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.DisplayName`
	`ImportedInstalledInstallerEvidence.ExternalInstaller EvidenceID`
	`ImportedInstalledInstallerEvidenceUsage.ExternalInstallerID`

`Version`	*Type:* text (max 72 characters). Key. Nullable
	The version of the software as reported by the installer evidence.
	Destination:
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.Version`
	`ImportedInstalledInstallerEvidence.ExternalInstaller EvidenceID`
	`ImportedInstalledInstallerEvidenceUsage.ExternalInstallerID`

`Publisher`	*Type:* text (max 200 characters). Key. Nullable
	The publisher of the software as reported by the installer evidence.
	Destination:
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.Publisher`
	`ImportedInstalledInstallerEvidence.ExternalInstaller EvidenceID`
	`ImportedInstalledInstallerEvidenceUsage.ExternalInstallerID`

`Evidence`	*Type:* text (max 32 characters). Key. Nullable
	Identifier for the type of installer evidence.
	Destination:
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.Evidence`
	`ImportedInstalledInstallerEvidence.ExternalInstaller EvidenceID`
	`ImportedInstalledInstallerEvidenceUsage.ExternalInstallerID`

`ProductCode`	*Type:* text (max 55 characters). Nullable The product code of the evidence. This is usually the MSI product code. Destination: `ImportedInstallerEvidence.ProductCode`

Column	Details
AccessMode	*Type:* text (max 128 characters). Key. Nullable The access mode of the installer evidence. Leave this blank unless this row is a virtualized application. In that case choose one of the values below that matches your application or desktop virtualization infrastructure. Possible values: • Local • App-V • XenApp • XenDesktop • VMware View • Office 365 Destination: ImportedInstallerEvidence.ExternalInstallerID ImportedInstallerEvidence.AccessModeID ImportedInstalledInstallerEvidence.ExternalInstaller EvidenceID ImportedInstalledInstallerEvidenceUsage.ExternalInstallerID
InstallDate	*Type:* text (max 10 characters). Nullable The install date of the installer evidence. The date must be specified in the following format: 'yyyyMMdd'. Destination: ImportedInstalledInstallerEvidence.InstallDate
DiscoveryDate	*Type:* text (max 10 characters). Nullable The date that the installer evidence was first seen. The date must be specified in the following format: 'yyyyMMdd'. Destination: ImportedInstalledInstallerEvidence.DiscoveryDate
NumberOfSessions	*Type:* big integer. Nullable The number of sessions that the installer evidence was in use by the user specified in the UserID column during the usage tracking period. If multiple users used the same application on the computer, create one row for each user with usage. Destination: ImportedInstalledInstallerEvidenceUsage.NumberOfSessions

Column	Details
`StartDate`	*Type:* text (max 10 characters). Nullable The start date of the usage. The date must be specified in the following format: 'yyyyMMdd'. Destination: `ImportedInstalledInstallerEvidenceUsage.StartDate`
`LastUsedDate`	*Type:* text (max 10 characters). Nullable The last used date of the usage. The date must be specified in the following format: 'yyyyMMdd'. Destination: `ImportedInstalledInstallerEvidenceUsage.LastUsedDate`

`UserID`	*Type:* big integer. Key. Nullable
	The DOMAIN/SAMAccountName for the user that the installer evidence was used by. If this software was used by multiple users, create one row for each user of the software on the computer.
	Destination:
	`ImportedInstalledInstallerEvidenceUsage.ExternalUserID`
	`ImportedUser.ExternalID`
	`ImportedUser.UserName (Element 2 after splitting on '\')`
	`ImportedUser.Domain (Element 1 after splitting on '\')`
	`ImportedUser.SAMAccountName (Element 2 after splitting on '\')`

ConsolidatedOracleDatabaseUser Template

ConsolidatedOracleDatabaseUser provides a list of the users for each Oracle database instance.

Table 913: Columns included with `ConsolidatedOracleDatabaseUser` templates

Column	Details

`UserID`	*Type:* big integer. Key
	The identifier used in the source connection for the instance end-user. This may be an integer or a string.
	Destination:
	`ImportedInstanceUser.ExternalID`
	`ImportedLicenseUser.ExternalID`

Column	Details

`ComputerID`	*Type:* big integer. Key
	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
	Destination:
	`ImportedInstanceUser.ExternalID`
	`ImportedInstanceUser.ComputerID`
	`ImportedLicenseUser.ExternalID`

`InstanceName`	*Type:* big integer. Key
	This field specifies the name of the database instance. If there are multiple instances, create a row for each instance in this spreadsheet. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID or this row will be skipped.
	Destination:
	`ImportedInstanceUser.ExternalID`
	`ImportedInstanceUser.InstanceID`
	`ImportedLicenseUser.ExternalID`

`Name`	*Type:* text (max 256 characters) The name of the user. Destination: `ImportedLicenseUser.UserName`
`AccountStatus`	*Type:* text (max 256 characters). Nullable The current status of the end-user account. Destination: `ImportedInstanceUser.AccountStatus`

Column	Details
`CreationDate`	*Type:* datetime. Nullable The date and time when the end-user was created. The date must be entered in one of the supported formats. Possible values: • yyyy/MM/dd • yyyy/MM/dd HH:mm:Ss • yyyy/MM/dd HH:mm • yyyy-MM-dd • yyyy-MM-dd HH:mm:Ss • yyyy-MM-dd HH:mm • yyyyMMdd • yyyyMMdd HH:mm:Ss • yyyyMMdd HH:mm Destination: `ImportedInstanceUser.CreationDate`
`LastLogonDate`	*Type:* datetime. Nullable The date and time when the end-user last logged on to the computer. The date must be entered in one of the supported formats. Possible values: • yyyy/MM/dd • yyyy/MM/dd HH:mm:Ss • yyyy/MM/dd HH:mm • yyyy-MM-dd • yyyy-MM-dd HH:mm:Ss • yyyy-MM-dd HH:mm • yyyyMMdd • yyyyMMdd HH:mm:Ss • yyyyMMdd HH:mm Destination: `ImportedInstanceUser.LastLogonDate`

Column	Details
`DefaultTablespace`	*Type:* text (max 256 characters). Nullable The default tablespace for an Oracle end-user. Destination: `ImportedInstanceUser.DefaultTablespace`
`TempTablespace`	*Type:* text (max 256 characters). Nullable The temporary tablespace for an Oracle end-user. Destination: `ImportedInstanceUser.TempTablespace`
`DisplayName`	*Type:* text (max 256 characters). Key The display name of the software as reported by the installer evidence. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID, Version, Publisher and InstanceName or this row will be skipped. Destination: `ImportedInstanceUser.ApplicationID`
`Version`	*Type:* text (max 72 characters). Key The version of the software as reported by the installer evidence. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID, DisplayName, Publisher and InstanceName or this row will be skipped. Destination: `ImportedInstanceUser.ApplicationID`
`Publisher`	*Type:* text (max 200 characters). Key The publisher of the software as reported by the installer evidence. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID, DisplayName, Version and InstanceName or this row will be skipped. Destination: `ImportedInstanceUser.ApplicationID`
`Evidence`	*Type:* text (max 32 characters). Key. Nullable Identifier for the type of installer evidence. Destination: `ImportedInstanceUser.ApplicationID`

Column	Details
`AccessMode`	*Type:* text (max 128 characters). Key. Nullable The access mode of the installer evidence. Leave this blank unless this row is a virtualized application. In that case choose one of the values below that matches your application or desktop virtualization infrastructure. Possible values: • Local • App-V • XenApp • XenDesktop • VMware View • Office 365 Destination: `ImportedInstanceUser.ApplicationID`

ConsolidatedRemoteAccessFile Template

The RemoteAccessFile spreadsheet is used for capturing application virtualization information. Systems such as Microsoft AppV and Citrix XenApp allow a user to access applications that are not installed on a local computer. This object allows you to provide applications that a user may access by specifying the file evidence.

When populating the RemoteAccessFile template, please note that an application can be identified by file evidence. If the evidence does not match the ARL then no application will be created. The evidence not recognised will appear under the ‘Unrecognised Evidence’ screen within Flexnet Manager Suite. From there, you may create applications for any unrecognised evidence as required, and lastly ensure any new application relates to a license. This results in the evidence now being recognised for the new application and may cause license consumption after the next reconciliation. This application virtualization access using files is a special case in application matching. It does not require a mandatory file link to the application and can user a ‘not for recognition’ file to link to an application. This is because application and desktop virtualization systems rarely provide enough file information for more complex application recognition rules to function.

If entering file evidence, you must provide the following key identifier fields. + 1 = FileName

The following identifier fields are typically requried for matching evidence in the ARL, however are not mandatory. + 1 = Company + 2 = FileVersion + 3 = Description + 4 = FileSize

File evidence does not have to be specified in the FileEvidence spreadsheet as well as here.

Table 914: Columns included with `ConsolidatedRemoteAccessFile` templates

Column	Details
`ServerID`	*Type:* big integer. Key This is the ComputerID of the server that publishes this virtual application. The ComputerID must match a computer from the Computer spreadsheet, and that computer must have an installation of the application this file is part of. If the server does not have an installation of an appropriate application then the user will not be shown as having access to that application. This is a mandatory field. Destination: `ImportedRemoteUserToApplicationAccess.ExternalServerID`

`FileName`	*Type:* text (max 256 characters). Key
	The name of the file used as evidence of software installation. For unix operating systems include the full path in the file name, including the opening '/'. For Windows operating systems the file path is specified in the FilePath column and this column must only contain the file name.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalFileID`
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.FileName`

`FileVersion`	*Type:* text (max 100 characters). Key. Nullable
	The version number of the file used as evidence of software installation.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalFileID`
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.FileVersion`

`ProductVersion`	*Type:* text (max 200 characters). Nullable The product version number in the file header. Destination: `ImportedFileEvidence.ProductVersion`
`ProductName`	*Type:* text (max 200 characters). Nullable The product name in the file header. Destination: `ImportedFileEvidence.ProductName`
`FilePath`	*Type:* text (max 400 characters). Nullable The path of the file used as evidence of software installation. Destination: `ImportedFileEvidence.FilePath`

Column	Details

`Company`	*Type:* text (max 100 characters). Key. Nullable
	The company in the file header.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalFileID`
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.Company`

`Description`	*Type:* text (max 200 characters). Key. Nullable
	The description in the file header.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalFileID`
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.Description`

`FileSize`	*Type:* integer. Key. Nullable
	The size of the file in bytes.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalFileID`
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.FileSize`

`Language`	*Type:* text (max 200 characters). Nullable The language in the file header. Destination: `ImportedFileEvidence.Language`

`UserID`	*Type:* big integer. Key
	The UserID must be populated with the fully qualified name e.g. Mydomain\ JohnSmith. If not then a User is not created.
	If fully qualified then this field populates the following user related fields. + 1 = The user name of the end-user from the text following the “". + 2 = The login name (SAM account name) of the end-user from the text following the “". + 3 = The domain name of the end-user from the text before the “".
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalUserID`
	`ImportedUser.ExternalID`
	`ImportedUser.UserName (Element 2 after splitting on '\')`
	`ImportedUser.Domain (Element 1 after splitting on '\')`
	`ImportedUser.SAMAccountName (Element 2 after splitting on '\')`

Column	Details

`AccessMode`	*Type:* text (max 128 characters). Key. Nullable
	The AccessMode states how an application has been accessed.
	Possible values:
	• Local
	• App-V
	• XenApp
	• XenDesktop
	• VMware View
	• Office 365
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalFileID`
	`ImportedRemoteUserToApplicationAccess.AccessModeID`
	`ImportedFileEvidence.ExternalFileID`
	`ImportedFileEvidence.AccessModeID`

ConsolidatedRemoteAccessInstaller Template

The RemoteAccessInstaller spreadsheet is used for capturing application virtualization information. Systems such as Microsoft AppV and Citrix XenApp allow a user to access applications that are not installed on a local computer. This object allows you to provide applications that a user may access by specifying the installer evidence.

When populating the RemoteAccessInstaller, please note that an application can be identified by installer evidence. If the evidence does not match the ARL then no application will be created. The evidence not recognised will appear under the ‘Unrecognised Evidence’ screen within Flexnet Manager Suite. From there, you may create applications for any unrecognised evidence as required, and lastly ensure any new application relates to a license. This results in the evidence now being recognised for the new application and may cause license consumption after the next reconciliation.

If entering installer evidence, you must provide the following key identifier fields. + 1 = DisplayName

The following identifier fields are typically requried for matching evidence in the ARL, however are not mandatory. + 1 = Version + 2 = Publisher + 3 = Evidence

Installer evidence does not have to be specified in the InstallerEvidence spreadsheet as well as here.

Table 915: Columns included with `ConsolidatedRemoteAccessInstaller` templates

Column	Details

`DisplayName`	*Type:* text (max 256 characters). Key
	The display name of the software as reported by the installer evidence and is part of the unique identifier for installer evidence.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalInstaller EvidenceID`
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.DisplayName`

`Version`	*Type:* text (max 72 characters). Key
	The version of the software as reported by the installer evidence and is part of the unique identifier for installer evidence.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalInstaller EvidenceID`
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.Version`

`Publisher`	*Type:* text (max 200 characters). Key
	Publishers of software applications (for example, “Microsoft”) as reported by the installer evidence and publisher is part of the unique identifier for installer evidence.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalInstaller EvidenceID`
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.Publisher`

`Evidence`	*Type:* text (max 32 characters). Key
	The evidence type of the software as reported by the installer evidence and is part of the unique identifier for installer evidence.
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalInstaller EvidenceID`
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.Evidence`

Column	Details
`ProductCode`	*Type:* text (max 55 characters). Nullable The product code of the evidence. This is usually the MSI product code and is not part of the unique identifier. Destination: `ImportedInstallerEvidence.ProductCode`

`UserID`	*Type:* big integer. Key
	The UserID must be populated with the fully qualified name e.g. Mydomain\ JohnSmith. If not then a User is not created.
	If fully qualified then this field populates the following user related fields. + 1 = The user name of the end-user from the text following the “". + 2 = The login name (SAM account name) of the end-user from the text following the “". + 3 = The domain name of the end-user from the text before the “".
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalUserID`
	`ImportedUser.ExternalID`
	`ImportedUser.UserName (Element 2 after splitting on '\')`
	`ImportedUser.Domain (Element 1 after splitting on '\')`
	`ImportedUser.SAMAccountName (Element 2 after splitting on '\')`

`AccessMode`	*Type:* text (max 128 characters). Key. Nullable
	The AccessMode states how an application has been accessed.
	Possible values:
	• Local
	• App-V
	• XenApp
	• XenDesktop
	• VMware View
	• Office 365
	Destination:
	`ImportedRemoteUserToApplicationAccess.ExternalInstaller EvidenceID`
	`ImportedRemoteUserToApplicationAccess.AccessModeID`
	`ImportedInstallerEvidence.ExternalInstallerID`
	`ImportedInstallerEvidence.AccessModeID`

ConsolidatedVMPool Template

The VMPool spreadsheet provides a simple method to associate virtual machines with groups (pools) on their host.

Table 916: Columns included with `ConsolidatedVMPool` templates

Column	Details
`PoolName`	*Type:* text (max 100 characters). Key The name of the pool. Destination: `ImportedVMPool.PoolName`
`ParentName`	*Type:* text (max 100 characters). Nullable The name of the parent pool. Destination: `ImportedVMPool.ParentName`
`PoolFriendlyName`	*Type:* text (max 256 characters) The friendly name of the pool. Destination: `ImportedVMPool.PoolFriendlyName`
`HostComputerID`	*Type:* big integer. Key The identifier used in the source connection for the computer which is hosting the pool. The HostComputerID should match the ComputerID in the Computer spreadsheet. Otherwise the record will be ignored. Destination: `ImportedVMPool.HostComputerID`

Column	Details
`ObjectType`	*Type:* text (max 256 characters). Key. Nullable The type of pool. Possible values: • Folder • Datacenter • ComputeResource • HostSystem • ResourcePool • VirtualMachine • PhysicalSharedPool • VirtualSharedPool • LPAR • RSET • ClusterComputeResource • PSET Destination: `ImportedVMPool.ObjectType`
`ComplianceConnectionID`	*Type:* integer. Key. Nullable The identifier for a data source connection in the `ComplianceConnection` table. Destination: `ImportedVMPool.ComplianceConnectionID`
`ParentObjectType`	*Type:* text (max 256 characters). Nullable The type of pool of the parent. Destination: `ImportedVMPool.ParentObjectType`
`NumberOfProcessors`	*Type:* decimal. Nullable The number of processors in this pool. Possible values: 120.45 Destination: `ImportedVMPool.NumberOfProcessors`

Column	Details
NumberOfCores	*Type:* decimal. Nullable The number of cores in this pool. Possible values: 120.45 Destination: ImportedVMPool.NumberOfCores

ConsolidatedWMIEvidence Template

ConsolidatedWMIEvidence provides a simpler interface to specify Windows Management Instrumentation (WMI) properties on computers. Other Web-Based Enterprise Management (WBEM) properties are supported from Unix computers as well. The most important data to provide in this spreadsheet is operating system installs. The 'Win32_OperatingSystem' class and the 'Name' property contains this data.

Table 917: Columns included with ConsolidatedWMIEvidence templates

Column	Details
ComputerID	*Type:* big integer. Key The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored. Destination: ImportedInstalledWMIEvidence.ExternalComputerID
ClassName	*Type:* text (max 50 characters). Key The WMI class name of the evidence. An example is 'Win32_OperatingSystem'. Destination: ImportedWMIEvidence.ExternalEvidenceID ImportedWMIEvidence.ClassName ImportedInstalledWMIEvidence.ExternalEvidenceID
PropertyName	*Type:* text (max 50 characters). Key The WMI property name of the WMI evidence. An example is 'Name'. Destination: ImportedWMIEvidence.ExternalEvidenceID ImportedWMIEvidence.PropertyName ImportedInstalledWMIEvidence.ExternalEvidenceID

Column	Details

`PropertyValue`	*Type:* text (max 256 characters). Key
	The value of the property of the WMI evidence. An example is 'Microsoft Windows 7 Enterprise'
	Destination:
	`ImportedWMIEvidence.ExternalEvidenceID`
	`ImportedWMIEvidence.PropertyValue`
	`ImportedInstalledWMIEvidence.ExternalEvidenceID`

`InstanceName`	*Type:* text (max 256 characters). Key. Nullable
	The name of the WMI class instance. This is important when there a multiple instances of a WMI class on a computer. An example is the 'Win32_VideoController' class that may have many instances with the same properties. In this case you need to specify the name of the instance here, 'Intel(R) HD Graphics Family' or 'NVIDIA Quadro K2100M' for example.
	Destination:
	`ImportedWMIEvidence.ExternalEvidenceID`
	`ImportedInstalledWMIEvidence.ExternalEvidenceID`
	`ImportedInstalledWMIEvidence.InstanceName`

9

Flexera Data Models

FlexNet Manager Suite includes Flexera Analytics, a technology that enables you to create reports and to customize dashboards, either for your enterprise or for personal use. You can build and/or customize reports and dashboards using data contained in the Flexera data models.

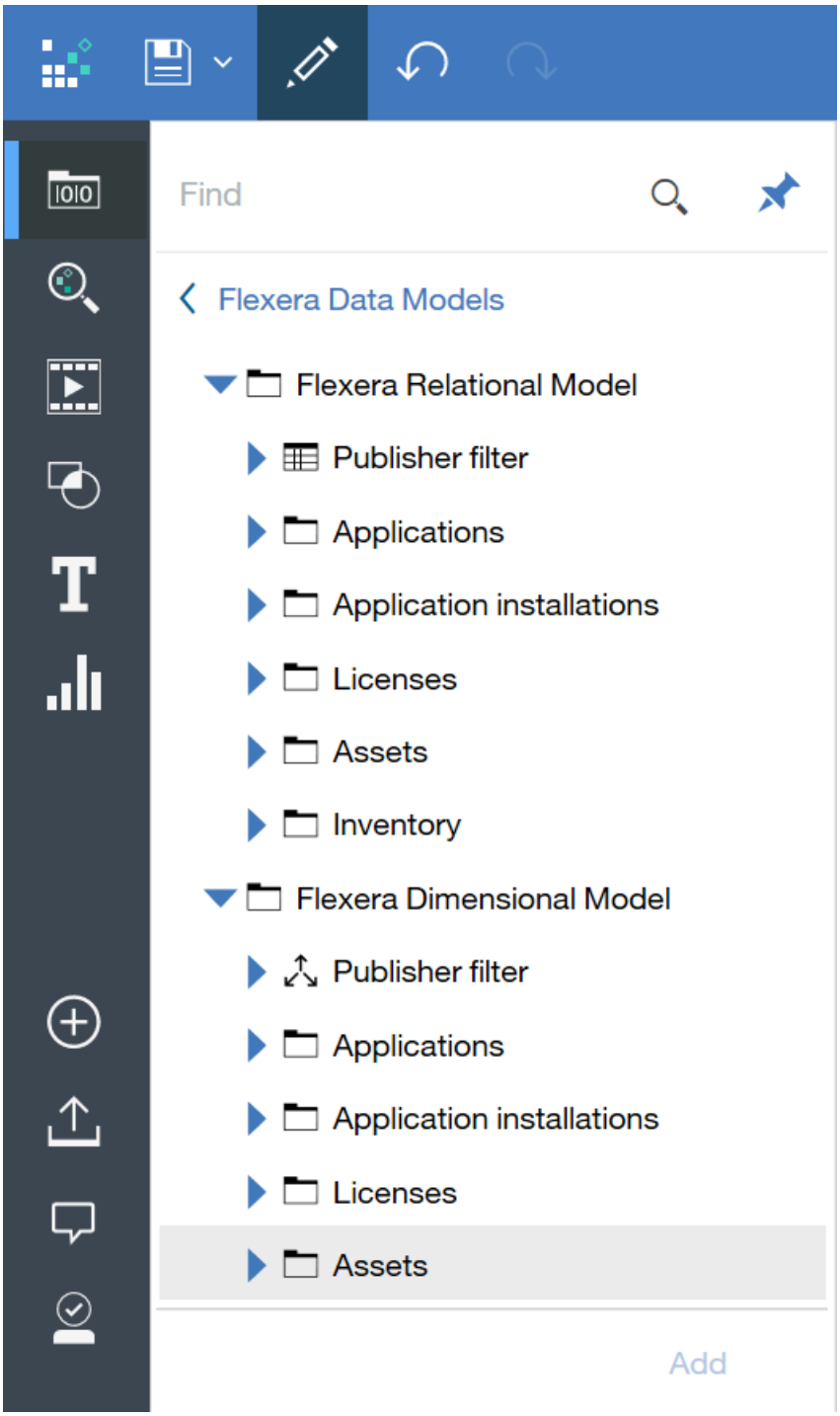
To help you use this data when customizing dashboards, widgets, and reports, Flexera Analytics provides two data models that organize your asset-management data and define how the data relates to each another. Within the data models, folders organize and structure the data into subject-area categories. Each folder in the subject area contains two types of data: measures and attributes. Measures represent numbers and facts, and attributes represent categories of descriptive data.

The two data models are:

- **Relational model** (see Relational model)
- **Dimensional model** (see Dimensional model)

Note: *It is recommended that you use data from one data model or the other when creating widgets and dashboards. You need to keep this in mind and plan ahead before you begin to create a dashboard.*

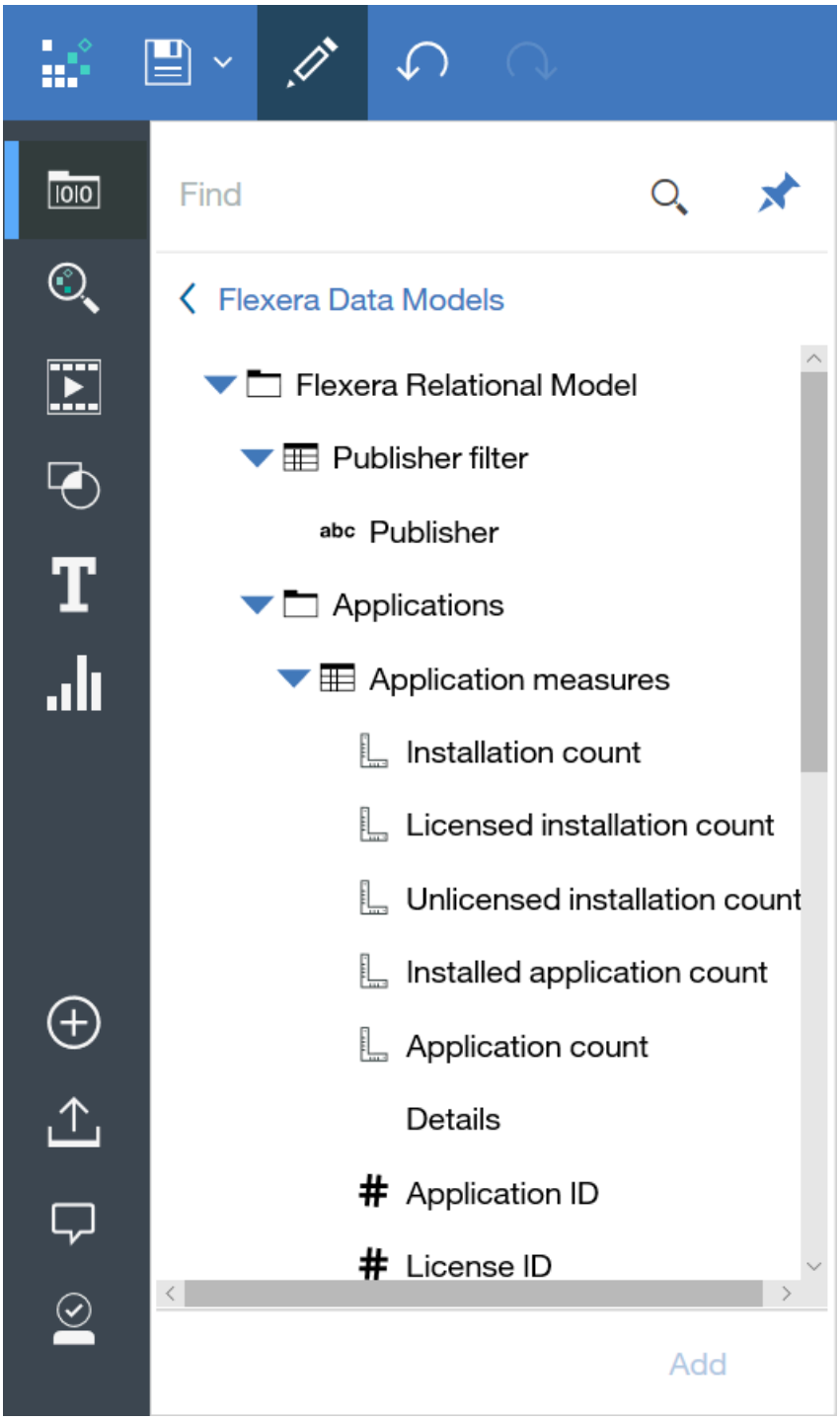
Figure 9: Flexera data models



Relational model

The relational model organizes data using measures and attributes but uses a flat structure. There is no ability to drill up or drill down on units of data to see how other data relates to them. There are also some attributes and measures that are specific to the relational model, such as the Inventory subject area.

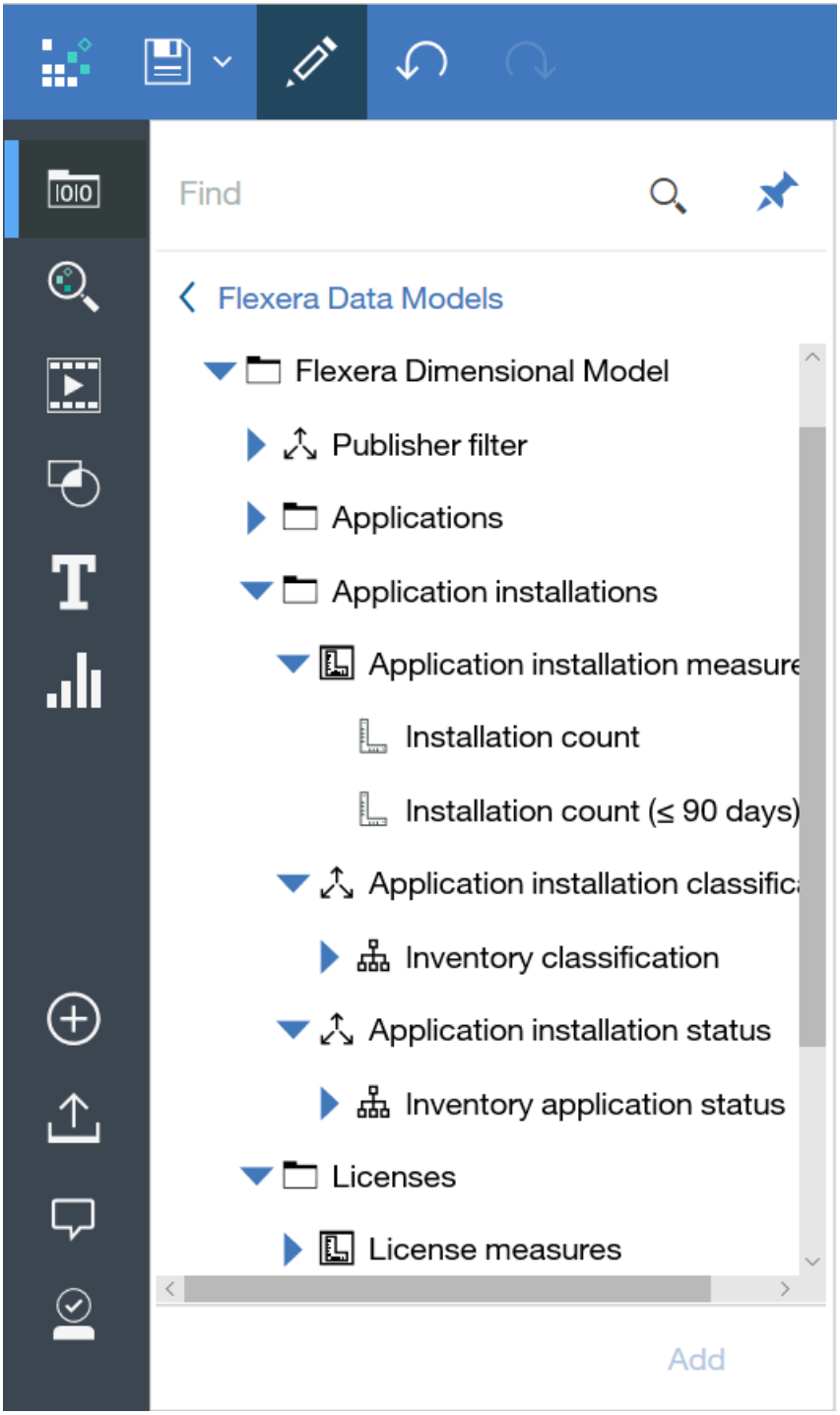
Figure 10: Relational model



Dimensional model

The dimensional model organizes data using measures and attributes but uses a hierarchy structure that enables you to drill up and drill down to see how data relates to other functionality.

Figure 11: Dimensional model



Relational Model Categories

The set of data categories in the relational model documented here includes:

- Publisher Filter (see Publisher Filter)

- Applications (see Applications)
- Application Installation (see Application Installation)
- Licenses (see Licenses)
- Assets (see Assets)
- Inventory (see Inventory)
- Contracts (see Contracts)
- Purchases (see Purchases).

Publisher Filter

Use this attribute when creating and customizing the Publisher filter widget in Flexera Analytics.

Folder	Measure/Attribute	Description
Publisher filter	**Publisher filter**	An attribute that describes the Publisher name of an application.

Applications

Use these measures and attributes when creating and customizing widgets related to applications in Flexera Analytics.

Folder	Measure/ Attribute	Description
Application measures	**Installation count**	A measure that defines the number of installations of a single application.
	Licensed installation count	A measure that defines the number of installations of a single application that has an associated license.
	Unlicensed installation count	A measure that defines the number of installations of a single application that do not have an associated license.
	Installed application count	A measure that defines the number of applications that report a valid installation.
	Application count	A measure that defined the total number of applications.
Application	**Publisher**	An attribute that describes the Publisher name of an application.
	Publisher and Product	A combined attribute that describes the Publisher name and Product name of an application.

Folder	Measure/ Attribute	Description
	Product	An attribute that describes the Product name of an application.
	Edition	An attribute that describes the Edition name of an application.
	Version	An attribute that describes the Version name of an application.
	Name	An attribute that describes the name of an application.
	Is managed	A Boolean attribute that describes a managed application that is authorized for use in the enterprise or not authorized for use. A value of 1 denotes managed applications (including authorized and unauthorized applications).
	Start of life	An attribute that describes the start date of an application.
	Release date	An attribute that describes the date an application was released.
	End of sales	An attribute that describes the date an application will not be sold after.
	End of support	An attribute that describes the date an application will be supported until.
	End of extended support	An attribute that describes the date an application will be supported until in extended cases.
	End of life	An attribute that describes the end date of an application. The vendor stops marketing, selling, or rework sustaining it.
Application category	**Application category**	An attribute that describes the category of an application. For example, "Software," "File Versioning," "Data Management," etc.
Application classification	**Application classification**	The measure/attribute column should also be Application classification and not Application category.
Application status	**Application status**	An attribute that describes the status of an application. For example, "Authorized," "Unauthorized," "Unmanaged," etc.
EOSL Filter	**End of extended support (90 days)**	A measure that defines applications whose end of extended support is within 90 days.
	End of extended support (180 days)	A measure that defines applications whose end of extended support is within 180 days.
	End of extended support (270 days)	A measure that defines applications whose end of extended support is within 270 days.

Folder	Measure/ Attribute	Description
	End of extended support (360 days)	A measure that defines applications whose end of extended support is within 360 days.
	End of life (90 days)	A measure that defines applications whose end of life is within 90 days.
	End of life (180 days)	A measure that defines applications whose end of life is within 180 days.
	End of life (270 days)	A measure that defines applications whose end of life is within 270 days.
	End of life (360 days)	A measure that defines applications whose end of life is within 360 days.
	End of sales (90 days)	A measure that defines applications whose end of sales is within 90 days.
	End of sales (180 days)	A measure that defines applications whose end of sales is within 180 days.
	End of sales (270 days)	A measure that defines applications whose end of sales is within 270 days.
	End of sales (360 days)	A measure that defines applications whose end of sales is within 360 days.
	End of support (90 days)	A measure that defines applications whose end of support is within 90 days.
	End of support (180 days)	A measure that defines applications whose end of support is within 180 days.
	End of support (270 days)	A measure that defines applications whose end of support is within 270 days.
	End of support (360 days)	A measure that defines applications whose end of support is within 360 days.

Application Installation

Use these measures and attributes when creating and customizing widgets related to application installations in Flexera Analytics.

Folder	Measure/ Attribute	Description
Application installation measures	**Installation count**	A measure that defines the number of installations for all applications.
	Installation count (<= 90 days)	A measure that defines the number of installations for all applications in the last 90 days.
Application installation classification	**Inventory classification**	An attribute that describes the classification of an application installation. For example, Commercial, Freeware, Component, etc.
Application installation status	**Inventory application status**	An attribute that describes the status of an application installation. For example, "Authorized," "Unauthorized," "Unmanaged," etc.

Licenses

Use these measures and attributes when creating and customizing widgets related to licenses in Flexera Analytics.

Folder	Measure/ Attribute	Description
License measures	**Entitlement count**	A measure that defines the total number of entitlements.
	Consumption count	A measure that defines the number of entitlements consumed.
	Installation count	A measure that defines the number of installations by device.
	Over-consumption count	A measure that defines the number of licenses at risk to expire or exceed entitlement.
	Unit price	A measure that defines the value of a single license in local currency.
	Financial risk	A measure that defines the dollar amount associated with the licenses at risk to expire or exceed entitlement.
	Risk %	A measure that defines the percentage of entitlements at risk.
	Consumption %	A measure that defines the percentage of entitlements in use.

Folder	Measure/ Attribute	Description
	License count	A measure that defines the number of licenses.
License	**License type**	An attribute that describes the type of a license. For example, Enterprise, Device, User, OEM, etc.
	Publisher	An attribute that describes the Publisher name of a license.
	Publisher and product	A combined attribute that describes the Publisher name and Product name of a license.
	Product	An attribute that describes the Product name of a license.
	Edition	An attribute that describes the Edition name of a license.
	Version	– An attribute that describes the Version name of a license.
	Is multi-product	An attribute that describes whether the license has more than one product.
Compliance status	**Compliance status**	An attribute that describes the compliance status of a license. For example, "Compliant," "At Risk," "Not tracked," etc.
License classification	**License classification**	An attribute that describes the classification of a license. For example, "Commercial," "Freeware," "Component," etc.
License status	**License status**	An attribute that describes the status of a license. For example, "Active," "Retired," "Purchased," etc.

Assets

Use these measures and attributes when creating and customizing widgets related to assets in Flexera Analytics.

Folder	Measure/Attribute	Description
Asset measures	**Asset count**	A measure that defines the number of hardware assets.
Asset status	**Asset status**	An attribute that describes the status of a hardware asset. For example, "Installed," "Disposed," "In Storage," "Purchased," etc.
Asset type	**Asset type**	An attribute that describes the type of a hardware asset. For example, "Workstation," "Laptop," "Server," etc.
Asset activity measures	**Date**	An attribute that describes the asset reported date.
	Age (days)	An attribute that describes the number of days relative to the asset date.
	New asset count	A measure that defines the number of new hardware assets created.
	Reported inventory count	A measure that defines the number of hardware assets last reported per asset date.

Folder	Measure/Attribute	Description
	New asset count (<= 30 days)	A measure that defines the number of hardware assets acquired in the last 30 days.
	New asset count (>30 days)	A measure that defines the number of hardware assets acquired more than 30 days ago.
	Reported inventory count (<= 30 days)	A measure that defines the number of hardware assets reported in inventory in the last 30 days.
	Reported inventory count (> 30 days)	A measure that defines the number of hardware assets reported in inventory more than 30 days ago.

Inventory

Use these measures and attributes when creating and customizing widgets related to inventory in Flexera Analytics.

Folder	Measure/Attribute	Description
Duplicate device name	**Inventory device name**	An attribute that describes the duplicate host name for a device.
	Duplicate count	A measure that defines the number of duplicate devices.
Duplicate serial number	**Serial number**	An attribute that describes the duplicate serial number for a device.
	Duplicate count	A measure that defines the number of duplicate devices.
Discovered devices activity	**Date**	An attribute that describes the devices activity date.
	Age (days)	An attribute that describes the number of days relative to the devices activity date.
	Missing inventory count	A measure that defines the total number of devices missing an inventory count.
	Missing inventory count (<= 90 days)	A measure that defines the number of devices missing an inventory count in the last 90 days.

Contracts

Use these measures and attributes when creating and customizing widgets related to contracts in Flexera Analytics.

Folder	Measure/Attribute	Description
Contract measures	**Contract amount**	A measure that defines the amount of the contract, in dollars.
	Contract count	A measure that defines the total number of contracts.

Folder	Measure/Attribute	Description
Contract	**Contract**	Attribute that defines the contract ID or number.
	Contract name	An attribute that defines the name of the contract.
	Is evergreen	An attribute that defines whether the contract will have an expiry date. If there is no expiry date, the contract is considered evergreen.
	Contract expiry date	An attribute that defines the date the contract expires.
	Contract review date	An attribute that defines the date the contract will be reviewed.
	Contract renewal date	An attribute that defines that date that the contract will be renewed.
	Contract with linked licenses	An attribute that defines whether the contract is linked to a license.
Contract status	**Contract status**	An attribute that defines the status of a contract, such as "Active," "Cancelled," or "Expired," etc.
Contract type	**Contract type**	An attribute that defines the contract type such as "Hardware maintenance and support," "Software license," "Software maintenance and support, etc."
Contract event filter	**Contracts expiring within (30 days)**	A measure that defines contracts that expire in 30 days.
	Contracts expiring within (45 days)	A measure that defines contracts that expire in 45 days.
	Contracts expiring within (90 days)	A measure that defines contracts that expire in 90 days.
	Contracts for renewal within (30 days)	A measure that defines contracts that will be renewed in 30 days.
	Contracts for renewal within (45 days)	A measure that defines contracts that will be renewed in 45 days.
	Contracts for renewal within (90 days)	A measure that defines contracts that will be renewed in 90 days.
	Contracts for review within (30 days)	A measure that defines contracts that will be reviewed in 30 days.
	Contracts for review within (45 days)	A measure that defines contracts that will be reviewed in 45 days.

Folder	Measure/Attribute	Description
	Contracts for review within (90 days)	A measure that defines contracts that will be reviewed in 90 days.

Purchases

Use these measures and attributes when creating and customizing widgets related to purchases in Flexera Analytics.

Folder	Measure/Attribute	Description
Latest purchase measures	**Purchase amount**	A measure that defines the purchase amount over the last 30 days, in dollars.
	Effective quantity	A measure that defines the total number of license entitlements brought in by a purchase.
	Purchase count	A measure that defines the total number of purchases over the last 30 days.
Purchase spend measures	**Purchase spent**	A measure that defines the purchase amount, in dollars, of purchases that have been bought in the last 12 months.
Unprocessed purchases	**Available entitlements**	A measure that defines the number of entitlements that are available with this purchase.
	Unprocessed purchase count	A measure that defines the number of unprocessed purchases.
Purchase	**Purchase No**	An attribute that defines the purchase number.
	Purchase description	An attribute that describes the product, such as "Windows Web Server," "Outlook 2016," etc.
	Purchase date	An attribute that defines the date of the purchase.
	Creation date	An attribute that defines the date the purchase record was created.
	Publisher name	An attribute that defines the publisher name of the purchase such as "Adobe," "IBM," and "Microsoft," etc.
	Vendor name	An attribute that defines the vendor name of the purchase, such as "Adobe," "IBM," and "Microsoft," etc.
Purchase type	**Purchase type**	An attribute that defines the purchase type, such as "Hardware," "Service," or "Software," etc.
Purchase status	**Purchase status**	An attribute that defines the status of the purchase such as "Cancelled," "Completed," "New," or "Pending."

Dimensional Model Categories

The set of data categories in the dimensional model documented here includes:

- Publish Filter (see Publisher Filter)
- Applications (see Applications)
- Application Installations (see Application Installation)
- Licenses (see Licenses)
- Assets (see Assets)
- Contracts (see Contracts)
- Purchases (see Purchases).

Publisher Filter

Use this attribute when creating and customizing the Publisher filter widget in Flexera Analytics.

Folder	Measure/Attribute	Description
Publisher filter	**Publisher filter**	An attribute that describes the Publisher name of an application.

Applications

Use these measures and attributes when creating and customizing widgets related to applications in Flexera Analytics.

Folder	Measure/ Attribute	Description
Application measures	**Installation count**	A measure that defines the number of installations of a single application.
	Licensed installation count	A measure that defines the number of installations of a single application that has an associated license.
	Unlicensed installation count	A measure that defines the number of installations of a single application that do not have an associated license.
	Installed application count	A measure that defines the number of applications that report a valid installation.
	Application count	A measure that defined the total number of applications.

Folder	Measure/ Attribute	Description
Application	**Publisher**	An attribute that describes the Publisher name of an application.
	Publisher and Product	A combined attribute that describes the Publisher name and Product name of an application.
	Product	An attribute that describes the Product name of an application.
	Edition	An attribute that describes the Edition name of an application.
	Version	An attribute that describes the Version name of an application.
	Name	An attribute that describes the name of an application.
	Is managed	A Boolean attribute that describes a managed application that is authorized for use in the enterprise or not authorized for use. A value of 1 denotes managed applications (including authorized and unauthorized applications).
	Start of life	An attribute that describes the start date of an application.
	Release date	An attribute that describes the date an application was released.
	End of sales	An attribute that describes the date an application will not be sold after.
	End of support	An attribute that describes the date an application will be supported until.
	End of extended support	An attribute that describes the date an application will be supported until in extended cases.
	End of life	An attribute that describes the end date of an application. The vendor stops marketing, selling, or rework sustaining it.
Application category	**Application category**	An attribute that describes the category of an application. For example, "Software," "File Versioning," "Data Management," etc.
Application classification	**Application classification**	The measure/attribute column should also be Application classification and not Application category.
Application status	**Application status**	An attribute that describes the status of an application. For example, "Authorized," "Unauthorized," "Unmanaged," etc.
EOSL Filter	**End of extended support (90 days)**	A measure that defines applications whose end of extended support is within 90 days.
	End of extended support (180 days)	A measure that defines applications whose end of extended support is within 180 days.

Folder	Measure/ Attribute	Description
	End of extended support (270 days)	A measure that defines applications whose end of extended support is within 270 days.
	End of extended support (360 days)	A measure that defines applications whose end of extended support is within 360 days.
	End of life (90 days)	A measure that defines applications whose end of life is within 90 days.
	End of life (180 days)	A measure that defines applications whose end of life is within 180 days.
	End of life (270 days)	A measure that defines applications whose end of life is within 270 days.
	End of life (360 days)	A measure that defines applications whose end of life is within 360 days.
	End of sales (90 days)	A measure that defines applications whose end of sales is within 90 days.
	End of sales (180 days)	A measure that defines applications whose end of sales is within 180 days.
	End of sales (270 days)	A measure that defines applications whose end of sales is within 270 days.
	End of sales (360 days)	A measure that defines applications whose end of sales is within 360 days.
	End of support (90 days)	A measure that defines applications whose end of support is within 90 days.
	End of support (180 days)	A measure that defines applications whose end of support is within 180 days.
	End of support (270 days)	A measure that defines applications whose end of support is within 270 days.
	End of support (360 days)	A measure that defines applications whose end of support is within 360 days.

Application Installation

Use these measures and attributes when creating and customizing widgets related to application installations in Flexera Analytics.

Folder	Measure/ Attribute	Description
Application installation measures	**Installation count**	A measure that defines the number of installations for all applications.
	Installation count (<= 90 days)	A measure that defines the number of installations for all applications in the last 90 days.
Application installation classification	**Inventory classification**	An attribute that describes the classification of an application installation. For example, Commercial, Freeware, Component, etc.
Application installation status	**Inventory application status**	An attribute that describes the status of an application installation. For example, "Authorized," "Unauthorized," "Unmanaged," etc.

Licenses

Use these measures and attributes when creating and customizing widgets related to licenses in Flexera Analytics.

Folder	Measure/ Attribute	Description
License measures	**Entitlement count**	A measure that defines the total number of entitlements.
	Consumption count	A measure that defines the number of entitlements consumed.
	Installation count	A measure that defines the number of installations by device.
	Over-consumption count	A measure that defines the number of licenses at risk to expire or exceed entitlement.
	Unit price	A measure that defines the value of a single license in local currency.
	Financial risk	A measure that defines the dollar amount associated with the licenses at risk to expire or exceed entitlement.
	Risk %	A measure that defines the percentage of entitlements at risk.
	Consumption %	A measure that defines the percentage of entitlements in use.

Folder	Measure/ Attribute	Description
	License count	A measure that defines the number of licenses.
License	**License type**	An attribute that describes the type of a license. For example, Enterprise, Device, User, OEM, etc.
	Publisher	An attribute that describes the Publisher name of a license.
	Publisher and product	A combined attribute that describes the Publisher name and Product name of a license.
	Product	An attribute that describes the Product name of a license.
	Edition	An attribute that describes the Edition name of a license.
	Version	– An attribute that describes the Version name of a license.
	Is multi-product	An attribute that describes whether the license has more than one product.
Compliance status	**Compliance status**	An attribute that describes the compliance status of a license. For example, "Compliant," "At Risk," "Not tracked," etc.
License classification	**License classification**	An attribute that describes the classification of a license. For example, "Commercial," "Freeware," "Component," etc.
License status	**License status**	An attribute that describes the status of a license. For example, "Active," "Retired," "Purchased," etc.

Assets

Use these measures and attributes when creating and customizing widgets related to assets in Flexera Analytics.

Folder	Measure/Attribute	Description
Asset measures	**Asset count**	A measure that defines the number of hardware assets.
Asset status	**Asset status**	An attribute that describes the status of a hardware asset. For example, "Installed," "Disposed," "In Storage," "Purchased," etc.
Asset type	**Asset type**	An attribute that describes the type of a hardware asset. For example, "Workstation," "Laptop," "Server," etc.
Asset activity measures	**Date**	An attribute that describes the asset reported date.
	Age (days)	An attribute that describes the number of days relative to the asset date.
	New asset count	A measure that defines the number of new hardware assets created.
	Reported inventory count	A measure that defines the number of hardware assets last reported per asset date.

Folder	Measure/Attribute	Description
	New asset count (<= 30 days)	A measure that defines the number of hardware assets acquired in the last 30 days.
	New asset count (>30 days)	A measure that defines the number of hardware assets acquired more than 30 days ago.
	Reported inventory count (<= 30 days)	A measure that defines the number of hardware assets reported in inventory in the last 30 days.
	Reported inventory count (> 30 days)	A measure that defines the number of hardware assets reported in inventory more than 30 days ago.

Contracts

Use these measures and attributes when creating and customizing widgets related to contracts in Flexera Analytics.

Folder	Measure/Attribute	Description
Contract measures	**Contract amount**	A measure that defines the amount of the contract, in dollars.
	Contract count	A measure that defines the total number of contracts.
Contract	**Contract**	Attribute that defines the contract ID or number.
	Contract name	An attribute that defines the name of the contract.
	Is evergreen	An attribute that defines whether the contract will have an expiry date. If there is no expiry date, the contract is considered evergreen.
	Contract expiry date	An attribute that defines the date the contract expires.
	Contract review date	An attribute that defines the date the contract will be reviewed.
	Contract renewal date	An attribute that defines that date that the contract will be renewed.
	Contract with linked licenses	An attribute that defines whether the contract is linked to a license.
Contract status	**Contract status**	An attribute that defines the status of a contract, such as "Active," "Cancelled," or "Expired," etc.
Contract type	**Contract type**	An attribute that defines the contract type such as "Hardware maintenance and support," "Software license," "Software maintenance and support, etc."
Contract event filter	**Contracts expiring within (30 days)**	A measure that defines contracts that expire in 30 days.

Folder	Measure/Attribute	Description
	Contracts expiring within (45 days)	A measure that defines contracts that expire in 45 days.
	Contracts expiring within (90 days)	A measure that defines contracts that expire in 90 days.
	Contracts for renewal within (30 days)	A measure that defines contracts that will be renewed in 30 days.
	Contracts for renewal within (45 days)	A measure that defines contracts that will be renewed in 45 days.
	Contracts for renewal within (90 days)	A measure that defines contracts that will be renewed in 90 days.
	Contracts for review within (30 days)	A measure that defines contracts that will be reviewed in 30 days.
	Contracts for review within (45 days)	A measure that defines contracts that will be reviewed in 45 days.
	Contracts for review within (90 days)	A measure that defines contracts that will be reviewed in 90 days.

Purchases

Use these measures and attributes when creating and customizing widgets related to purchases in Flexera Analytics.

Folder	Measure/Attribute	Description
Latest purchase measures	**Purchase amount**	A measure that defines the purchase amount over the last 30 days, in dollars.
	Effective quantity	A measure that defines the total number of license entitlements brought in by a purchase.
	Purchase count	A measure that defines the total number of purchases over the last 30 days.
Purchase spend measures	**Purchase spent**	A measure that defines the purchase amount, in dollars, of purchases that have been bought in the last 12 months.
Unprocessed purchases	**Available entitlements**	A measure that defines the number of entitlements that are available with this purchase.

Folder	Measure/Attribute	Description
	Unprocessed purchase count	A measure that defines the number of unprocessed purchases.
Purchase	**Purchase No**	An attribute that defines the purchase number.
	Purchase description	An attribute that describes the product, such as "Windows Web Server," "Outlook 2016," etc.
	Purchase date	An attribute that defines the date of the purchase.
	Creation date	An attribute that defines the date the purchase record was created.
	Publisher name	An attribute that defines the publisher name of the purchase such as "Adobe," "IBM," and "Microsoft," etc.
	Vendor name	An attribute that defines the vendor name of the purchase, such as "Adobe," "IBM," and "Microsoft," etc.
Purchase type	**Purchase type**	An attribute that defines the purchase type, such as "Hardware," "Service," or "Software," etc.
Purchase status	**Purchase status**	An attribute that defines the status of the purchase such as "Cancelled," "Completed," "New," or "Pending."

A

B

C

D

E

J

K

L

S

T