Previous features and fixed issues

This build includes many new features and fixed issues, including the following updates from previous releases.

9.0 New features

Introduces new Workspace features

This release of Idera Comparison Toolset introduces new Workspace features, including:

- support for multiple database files
- database storage instead of xml storage used in previous versions
- an increase in the number of stored sessions from 25 to 100
- allowing you to select the Workspace database file you want to use

Supports SQL Server 2019

Idera SQL Comparison Toolset 9.0 adds support for SQL Server 2019 and the latest Azure database.

Supports .NET 4.6

This release of Idera Comparison Toolset required .NET 4.6.

Improved command line features

Command line users will notice:

- a new option in the config file to encrypt sensitive information, such as SQL Server instances, credentials, database names, and more
- more detailed parsing of the config file, which generates more warnings and suggestions
- new samples, located in the installation folder
- improved performance and ease of use
- support for new UTF-32 encoding for output files

Updates to entity filters

Version 9.0 updates entity filters as an exclusionary method only. Previously, filters were used to include and exclude objects, which created some confusion. You still can include objects via filters, but their primary function is to exclude them. ***If you are using entity filters prior to upgrading***, you may have to update your criteria.

9.0 Fixed issues

- A fix in the Data Compare object dictionary makes the name-search for tables and views case-insensitive. Schema portion of the object name is now optional.
- The Data compare Application Settings window contains a new option that specifies the max number of rows in the data grid.

- Data Compare fixes an issue when comparing characters, which, in the ASCII table, sit between the uppercase and the lowercase chars: 90 < char < 97. The previous method generated errors in some specific cases when comparing these characters.
- Data Compare fixes an issue with the mapping rules. In some cases, they didn't save properly or were not applied to a comparison session.

7.5 New features

SQL Server

Provides SQL Server 2017 support

Idera SQL Comparison Toolset 7.5 provides support to compare and synchronize SQL Server 2017 databases. The new features added on SQL Server 2017, are not supported but will be included in a future release.

7.5 Fixed issues

There are no fixed issues in this release.

7.1.7 New features

There are no new features in this release.

7.1.7 Fixed issues

Resolves an issue causing an error during script generation

This release of Idera SQL Data Compare resolves an issue that caused an error to appear when generating synchronization script.

Includes new DATE mapping rules

Idera SQL Data Compare mapping rules include a new mapping between the DATE data type and other similar data types, such as DATETIME, to resolve some compatibility issues.

Corrects a serialization issue

This release fixes an issue with the serialization of the `HierarchyId` data type requiring additional memory. Serialization is a process that is triggered when Idera Data Compare finds a table too big to fit in memory.

7.1.5 New features

There are no new features in this release.

7.1.5 Fixed issues

Resolves an issue causing an error before the comparison completes

This release resolves an issue causing some Idera SQL Schema compare users to see an error message stating that an item with the same key is already added to the database. This error appeared before the comparison was complete and related to database assemblies didn't map properly due to multiple files or multiple file having the same name but a different path or extension.

Corrects an issue caused by conflicting data types

Idera SQL Comparison Toolset 7.1.5 corrects a minor scripting issue with the user-defined types based on SQL Server native types **varchar(max)**, **nvarchar(max)**, and **varbinary(max)**.

7.1 New features

Supports spatial index AUTO_GRID tessellation schemes

Idera SQL Comparison Toolset 7.1 supports the following spatial index auto-tessellation schemes:

- `GEOMETRY_AUTO_GRID` Supported on SQL Server 2012, SQL Server 2014, SQL Server 2016, and Azure v12.
- `GEOGRAPHY_AUTO_GRID` Supported on SQL Server 2012, SQL Server 2014, SQL Server 2016, and Azure v12.

Workspace displays a warning if JavaScript is disabled

The Idera SQL Comparison Toolset Workspace now displays a warning message if JavaScript in Internet Explorer is disabled. JavaScript is not critical, but is needed for some of the Workspace functionalities. A master option in the app.config can hide this warning. The toolset requires JavaScript to be enabled for IE only.

7.1 Fixed issues

Improves data export to Microsoft Excel

Idera SQL Data Compare includes the following resolutions regarding the export functionality:

- **Export to CSV File**. The exported CSV file now is fully compliant with .csv specifications. This includes the correct data separators (depending on the data type), the proper handling of the new-lines embedded in the column values, etc.
- **Export to Text File**. With the new option to export to text, the user can select field and row separators and to include or exclude column name.

7.0 New features

Installation and configuration updates

Idera SQL Comparison Toolset 7.0 contains the following installation and configuration updates:

- The shortcuts for all Idera SQL Comparison Toolset executables now support the option to **Run as administrator**.
- Installation no longer supports Microsoft .NET 2.0.

- The Snapshot Converter utility is no longer included in Idera SQL Comparison Toolset. This utility was used to provide snapshot conversion from Idera SQL Comparison Toolset 2.5 and therefore is no longer necessary.
- The Idera SQL Comparison Toolset version, stored in the registry key **HKLM\SOFTWARE\Idera\SQLtoolbox\SQL comparison toolset**, now reflects version 7.0.0.0.

SQL Schema Compare Microsoft SQL Server 2016 support

Idera SQL Comparison Toolset 7.0 offers support for the following Microsoft SQL Server 2016 features:

Security policies

Idera SQL Schema Compare supports the new securities policy on a regular disc table with no restrictions. Idera SQL Schema Compare handles all synchronization scenarios and can:

- create a policy
- drop a policy.
- alter the policy by adding, dropping, or altering the policy predicates
- alter the policy state
- alter the policy “not for replication” attribute.

In addition to the regular tables, Idera SQL Schema Compare also supports policies on other database objects. Note that each type of policy does contain some restrictions and other subtleties imposed by SQL Server, such as:

- Security policies on memory tables. The predicate in this case can be created only with a natively-compiled function. Being natively-compiled makes the function a non-transactional object. In the context of schema, it means that during the database synchronization, the function is executed outside of the main transaction similar to memory tables. The security policy itself remains transactional.
- Security policies on views. This type of policy supports only the FILTER predicate and not the BLOCK.
- Azure v11 does not support security policies.
- Azure v12 supports security policies.

Idera SQL Schema Compare 7.0 includes the new comparison option **Compare security policies**. This option determines whether the policies and compared and synchronized.

Column master keys

Idera SQL Schema Compare supports the new column master key (CMK). The CMK comparison includes the following limitations:

- The CMK definition contains a component referred to as the “key-store-provider.” This is a software product installed in the client machine that provides access to the “store” containing the CMK. ***If the provider does not exist in the client machine*** , the CMK synchronization fails. Because the provider is a resource outside of the database, Idera SQL Schema Compare cannot check whether it exists. It will however generate a warning so that the user is aware of the issue.

- Another component of the CMK definition is the “key-path”. This is the key, in the key store, that provides the CMK encryption functionality. ***If the key is not found in the client machine*** , the CMK in the database fails. Because this key is a resource outside of the database, Idera SQL Schema Compare cannot verify whether it exists. It will however generate a warning so that the user is aware of the issue.
- Idera SQL Schema Compare cannot synchronize changes between two CMKs that have columns bound to them. This type of change requires a table rebuild, which cannot occur when columns are encrypted.

It is important to mention that Azure v12 supports CMK, but uses a provider that is different from the on-premise SQL Server. Idera SQL Schema Compare compares the CMK on Azure and generates a warning if the provider is not supported by the target database. Note that the warning may not be completely accurate because the list of providers is not fully known and goes beyond the scope of the database. The only Azure provider known at this time is the Azure Key Vault provider.

Idera SQL Schema Compare 7.0 includes the following new options for the CMK:

- **Compare column master keys**. Determines whether the CMK are compared.
- **Compare column master key provider**. Indicates whether the key-store-provider of the CMK is compared.
- **Compare column master key pat**. Indicates whether the key-path component of the CMK is compared.

Column encryption keys

Idera SQL Schema Compare supports the new columns encryption keys (CEK), which work with the CMK to provide column encryption in SQL Server 2016. CEKs are also supported on Azure v12.

The new option **Compare column encryption keys** determines whether the CEKs are compared and synchronized.

Natively-compiled functions

Natively-compiled functions are SQL functions created using the natively-compiled attribute. They are used by memory-object features, such as the security policies on memory tables or default/check constrains on these tables. Idera SQL Schema Compare supports the natively-compiled functions. Notre that these functions are non-transactional, so they are executed outside the main schema transaction when databases are synchronized.

There are no new options for natively-compiled functions. The options for standard functions control the natively-compiled functions as well.

Azure v12 does not support natively-compiled functions.

Natively-compiled triggers

Natively-compiled triggers are SQL triggers created using the natively-compiled attribute. They are used primarily by the memory tables. Idera SQL Schema Compare supports the natively-compiled triggers. Similar to other natively-compiled objects, these triggers are non-transactional.

There are no new options for natively-compiled triggers.

Azure v12 does not support natively-compiled triggers.

Columns

Idera SQL Schema Compare 7.0 supports the following SQL Server 2016 column enhancements.

- **Encryption**. Idera SQL Schema Compare supports the column encryption specified via the clause: `ENCRYPTED WITH (…)` with some restrictions. For example, a schema change that requires a table rebuild is not supported when the table contains encrypted columns. In this case, Idera SQL Schema Compare marks the table “non-comparable.” This type of change is not supported because data transfer, which is part of the rebuilt script, is not possible when the table is encrypted.
  Idera SQL Schema Compare 7.0 includes two new options for the column encryption feature:
  - **Compare column encryption**. Indicates whether the properties used by the column encryption is compared.
  - **Script column encryption**. Indicates whether the encryption is scripted.

  Note that encrypted columns are supported on Azure v12.
- **Dynamic Data Mask**. Idera SQL Schema Compare 7.0 supports the column dynamic data mask specified via the clause: `MASKED WITH (FUNCTION = email()|default()|partial()...)` clause. This feature provides data obfuscation for a column.
  Idera SQL Schema Compare 7.0 includes two new options for the dynamic data mask:
  - **Compare column mask**. Indicates whether the mask is compared.
  - **Script column mask**. Determines whether the mask is scripted.

  Note that dynamic data mask is supported on Azure v12.
- **System-versioning system-time columns**. System-time columns are type DateTime2 columns created with special attributes that can be used by the new system-versioning feature in SQL Server 2016.
  Idera SQL Schema Compare supports the system-time columns, including the “hidden” attribute and the new sys-versioning table clause.

System-versioned tables

System-versioned tables, sometimes referred to as “temporal” tables, are fully supported. Because sys-versioning is a major feature in SQL Server 2016, with many parts and objects, it is hard to fully document it. Idera SQL Schema Compare:

- supports system-time columns created with the attribute `GENERATED ALWAYS AS ROW START|ROW END`.
- supports the HIDDEN attribute for the system-columns.
- maps the system-time columns by the “period-type” instead of name. The period type is the `ROW START|ROW END` attribute.
- because the system-time columns have the same type and do not contain user data, Idera SQL Schema Compare always marks them as equal. They do not trigger a table difference.
- adds the system-time columns even when the target table is not system-versioned. This is done so that the user can enable the system-versioning via the `ALTER TABLE` statement.

- supports the new table embedded clause: `PERIOD FOR SYSTEM_TIME ([ValidFrom], [ValidTo])`.
- supports the new table option: `SYSTEM_VERSIONING = ON (HISTORY_TABLE = <table>, DATA_CONSISTENCY_CHECK = ON)`.
- supports the system-versioning on partition tables. The primary table, the history table or both can be partitioned.
- supports system-versioning on memory tables.

When it comes to scripting, system-versioned tables are notoriously more complicated that the regular tables. Most schema changes do not succeed when the sys-versioning is enabled. When necessary, Idera SQL Schema Compare disables sys-versioning, synchronizes the table, and enables it back. This action triggers multiple warnings to make sure that the user is aware when the sys-versioning changes.

Idera SQL Schema Compare 7.0 adds the following new options for the sys-versioned tables:

- **Compare system versioning**. Determines whether the sys-versioning is compared.
- **Ignore system-versioning history table name**. Indicates whether the name of the history table should be ignored. Because the History table is hidden from the user, its name is not relevant.
- **Script system versioning**. Determines whether the sys-versioning is scripted.
- Drop system-versioning history table. Indicates whether the history table is dropped when the system-versioning is disabled or the primary table is dropped.

Idera SQL Schema Compare supports the system-versioning on Azure v12.

In-Memory tables

SQL Server 2016 introduces many enhancements related to memory tables. Idera SQL Schema Compare supports the following:

- New unique constraints on a memory table.
- New check constraint on a memory table.
- New foreign keys on a memory table. The foreign key references another memory table.
- New `AFTER` triggers on a memory table. Triggers are natively-compiled.
- New system-versioning on memory tables.
- New memory table `IDENTITY`. SQL Server 2016 limits this to `IDENTITY(1,1)`.

Extended properties are supported on all objects. It is important to note that all new objects, including the memory tables, are non-transactional. Idera SQL Schema Compare executes them outside the main transaction when databases are synchronized.

The comparison options for memory tables existed in the previous version of Idera SQL Schema Compare. Version 7.0 adds a new **Compare hash index bucket count** option, which determines whether the `BUCKET_COUNT` property of an hash index is compared. A “hash index” can be added to a memory table type as well.

Memory tables have the following limitations:

- **Memory tables are non-transactional**. This is a SQL Server restriction. However, other database objects that normally are transactional, when associated with the memory tables,

become non-transaction as well. For example, the database schema is a transactional object. When the schema is associated with a memory-table, it changes to non-transactional. The same is true for extended properties, permissions, unique constraints, primary keys, etc.
- **SQL Server 2016 supports memory table indexes on n(var)char columns of any collation**. SQL Server 2014 is more restrictive and requires a `_BIN` collation. This seemingly small change could cause synchronization issues between a SQL Server 2016 and a SQL Server 2014.

Azure v12 does not support memory tables.

Stretch tables

Idera SQL Schema Compare supports the stretched tables or the remote data archiving feature. RDA is supported even on system-versioned tables (the history in this case can be configured as a stretched table).

Schema changes that require a table rebuild are not supported when the table is stretched. Most of the changes on a stretched table require additional script that disables stretching. Idera SQL Schema Compare generates warnings, so that the user is aware when the table stretching changes.

Version 7.0 adds the following new options specifically for the stretched tables:

- **Compare remote data archiving**. Determines whether the RDA feature is compared.
- **Script remote data archiving**. Determines whether the RDA is scripted.
- **Script history remote data archiving**. Determines whether the RDA of a history table is scripted. This applies to the sys-versioned tables only.
- **Abandon remote data when disabling archiving**. Indicates whether the remote data is discarded when the RDA is disabled.

Azure does not support stretched tables. The remote data of a stretched table is on Azure, but the primary stretched table is supported only on-premise databases.

Stretched database

Idera SQL Schema Compare cannot enable a database for stretching. It does however generate a template that can be used by the user to manually, outside the toolset, enable the database stretching. The following T-SQL snippet, commented out, represents the template:

```
/*
--
-- The following template can be used to enable remote data
archiving in the database
--

--1. run sp_configure and turn the 'remote data archive' option on
EXEC sp_configure 'remote data archive' , '1';
GO
RECONFIGURE;
GO
```

```
--2. create the database master key
CREATE MASTER KEY ENCRYPTION BY PASSWORD = '<password>';
GO

--3. create a database credential to use for communication between
the on-premise database and the remote Azure database
CREATE DATABASE SCOPED CREDENTIAL <credential> WITH IDENTITY =
'<identity>' , SECRET = '<password>';
GO

--4. turn on the remote data archiving in the database
ALTER DATABASE <database>
SET REMOTE_DATA_ARCHIVE = ON
(
SERVER = '<azure-server>.database.windows.net',
CREDENTIAL = <credential>
);
GO
*/
```

It is worth mentioning that Idera SQL Schema Compare does not automatically enable the database stretching for two reasons:

1. It cannot create the database master key, since the key password is not available in the database catalog.
2. It cannot create the database-scoped credential used for stretching. The credential identity and the secret, which are an Azure account and a password, are not in the database catalog.

External tables

Idera SQL Schema Compare 7.0 does not support the external tables.

User-defined table types

Idera SQL Schema Compare 7.0 brings the following changes to the user-defined table types:

- In additional to memory table types, indexes are now supported on regular table types.
- Objects associated with a memory table type, including the table type, are non-transactional.
- Idera SQL Schema Compare 7.0 fixes an issue with extended properties on columns. In some cases, they failed to properly synchronize.

Columnstore indexes

Idera SQL Schema Compare 7.0 supports the following new features on columnstore indexes:

- The new columnstore filter predicate, specified via the WHERE clause.
- The new columnstore COMPRESSION_DELAY option.

Note that columnstore indexes are supported on a premium tier of Azure v12. As a result, Idera SQL Schema Compare allows and supports them on all versions of Azure.

New permissions

SQL Server 2016 contains the following new permissions supported by Idera SQL Schema Compare:

- ALTER ANY COLUMN ENCRYPTION KEY (ALCK)
- ALTER ANY COLUMN MASTER KEY (ALCM)
- ALTER ANY DATABASE SCOPED CONFIGURATION (ALDC)
- ALTER ANY EXTERNAL DATA SOURCE (AEDS)
- ALTER ANY EXTERNAL FILE FORMAT (AEFF)
- ALTER ANY SECURITY POLICY (ALSP)
- VIEW ANY COLUMN ENCRYPTION KEY DEFINITION (VWCK)
- VIEW ANY COLUMN MASTER KEY DEFINITION (VWCM)
- ALTER ANY MASK (AAMK)
- UNMASK (UMSK)
- EXECUTE ANY EXTERNAL SCRIPT (EAES)

Azure v12 supports the same permissions.

New comparison options

Below is a summary of the new options supported by Idera SQL Schema Compare 7.0:

- **Compare security policies**. Indicates whether the security policies are compared.
- **Compare column master keys**. Indicates whether the CMK are compared.
- **Compare column master key provider**. Whether the provider of a CMK is compared.
- **Compare column master key path**. Whether the key-path property of a CMK is compared.
- **Compare column encryption keys**. Indicates whether the CEK are compared.
- **Compare column encryption**. Indicates whether the properties related to column encryption, such as the encryption key, the encryption algorithm, are compared.
- **Compare column mask**. Indicates whether the column dynamic data mask is compared.
- **Compare system versioning**. Whether the sys-versioning of the regular or memory tables is compared.
- **Ignore system versioning history table name**. Whether the name of the history table associated with the sys-versioning is ignored.
- **Compare hash index bucket count**. Whether the BUCKET_COUNT property of an hash index is compared.
- **Compare remote data archiving**. Whether the remote data archiving feature (or stretching) is compared.
- **Script column encryption**. Whether the column encryption is scripted.
- **Script column mask**. Whether the column dynamic data mask is scripted.
- **Script system versioning**. Whether the sys-versioning is scripted.
- **Drop system versioning history**. Indicates whether the history table should be dropped when the primary table is dropped or when the sys-versioning is disabled.

- **Script remote data archiving**. Whether the remote data archiving (or stretching) is scripted.
- **Script history remote data archiving**. Whether the remote data archiving (or stretching) is scripted specifically for the history of a system-versioned table.
- **Abandon remote data when disabling archiving**. Whether the Azure remote data should be discarded when the stretching is disabled.

Other changes

Idera SQL Schema Compare 7.0 contains other changes including:

1. The SQL Server Database Selection popup now includes the entity filters. It allows the user to set the filters before comparing databases. Filters are fully saved and retained in the workspace session.
2. The Idera SQL Schema Compare command line <Database> element supports the following new properties:
    a. **ConnectionString**. The entire database connection script.
    b. **PacketSize**. The SQL connection packet-size.
    c. **Pooled**. Indicates whether the SQL connection should be pooled.
    d. **OtherProperties**. Other connection properties.
3. The command line displays operational message on the screen and logs them at the same time; used to be either or in the previous version.
4. The command line /quiet option does not display messages on the screen. Messages are logged regardless of this option.
5. The command line script and log file are not created if databases are equal; used to be empty in the previous version.
6. The command line uses by default UTF-8 encoding for all files.

SQL Schema Compare Command Line wizard

Idera SQL Schema Compare 7.0 contains the new Command Line wizard, which allows the user to create a command line xml file step-by-step, via a simple and intuitive interface.

In addition to the Command Line wizard, Idera SQL Schema Compare includes a new option in the schema compare UI. It allows the user to export a comparison session and create a command line directly from the UI.

Data Compare updates

New features and enhancements

Idera SQL Data Compare 7.0 supports the following new features:

- Compares the data on system-versioned tables, both disc and memory tables.
- Can disable, via an option, the system-versioning before synchronizing the data.
- Compares the stretched tables.
- Can set, via an option, the stretch table query scope. This can be set to either LOCAL_ONLY or LOCAL_AND_REMOTE (default).
- Supports the new memory table foreign keys and natively-compiled triggers. The objects are non-transactional.

Idera SQL Data Compare 7.0 changes the transactional isolation level from “SERILIZED” to “READ COMMITTED” in order to supported the script that disables the foreign keys on memory tables.

Idera SQL Data Compare does not support encrypted tables.

New Compare options

Idera SQL Data Compare 7.0 contains the following new options:

- **Compare system versioned tables**. Indicates whether the sys-versioned tables (or temporal tables) are compared.
- **Disable system versioning**. Whether the sys-versioning is disabled before synchronizing the data.
- **Compare stretched tables**. Whether the stretched tables are compared.
- **Stretched table query scope**. Whether the query against a stretched table selects only the local data (LOCAL_ONLY) or both local and remote data (LOCAL_AND_REMOTE).

Data Compare Command Line Wizard

Idera SQL Data Compare 7.0 brings the new command line wizard. The wizard is similar to schema and creates an xml file specified for the data compare command line. In addition the wizard, the user can create the config file directly from the data compare UI.

7.0 Fixed Issues

Schema Compare

Version 7 fixes the following issues:

- It fixes an issue with the SPARSE column. The NULL|NOT NULL constraints should be specified after the SPARSE attribute.
- It fixes an issue that occurred when the synchronization script contains extended characters, such as Chinese letters. Schema now uses UTF-8 encoding for the script and the log files.
- It fixes an issue with the DATA_COMPRESSION attribute on a filestream table. If the filestream table contains an embedded primary key, data compression must be specified either under the primary key or the table, but not both.