

IDEERA

SQL Comparison Toolset

Version 9.0



Table of Contents

Manage data and schema changes	1
Release notes	2
New features and fixed issues	3
Previous features and fixed issues	5
Known issues.....	18
Welcome to SQL Comparison Toolset	20
Getting started	21
Installing and Uninstalling the Toolset	22
Requirements.....	23
Activating a license	24
Supported platforms.....	25
Updates and Support	26
IDERA SQL Schema Compare	27
SQL Schema Compare Application window.....	28
SQL Schema Compare Application Settings	29
SQL Schema Compare Workspace	32
SQL Schema Compare Comparing Schemas.....	39
SQL Schema Compare Working with Filegroups.....	50
SQL Schema Compare Synchronizing Databases	56
SQL Schema Compare Comparison Sessions	63
SQL Schema Compare Tools and Utilities.....	64
IDERA SQL Schema Compare Command Line	67
Using SQL Schema Compare Command Line	68
SQL Schema Compare Command Line Wizard	69
SQL Schema Compare Command Line Source and Target Database	70
SQL Schema Compare Command Line Database from a Snapshot.....	72
SQL Schema Compare Command Line Creating a Snapshot.....	74
SQL Schema Compare Command Line Excluding Objects by Type	75
SQL Schema Compare Command Line Comparison Options.....	76
SQL Schema Compare Command Line Excluding Objects Using Entity Filters	78

SQL Schema Compare Command Line Excluding Objects by Name.....	80
SQL Schema Compare Command Line Filegroup Mappings.....	82
SQL Schema Compare Command Line Log Files and Options.....	83
SQL Schema Compare Command Line Return Codes	85
IDERA SQL Data Compare	86
SQL Data Compare Application window	87
SQL Data Compare Application Settings.....	88
SQL Data Compare Workspace	91
SQL Data Compare Comparing.....	96
SQL Data Compare Synchronizing Data.....	114
SQL Data Compare Comparison Sessions.....	119
IDERA SQL Data Compare Command Line	120
Using the SQL Data Compare Command Line	121
Starting with the Command Line Wizard	122
Specifying the Source and Target Databases	123
Pairing Tables and Views.....	124
Comparison Options	125
Scripting Options	126
Table Mapping Rules	127
Excluding Tables.....	128
Customizing Tables.....	130
View Mapping Rules.....	132
Excluding Views	133
Customizing Views	135
Log Files and Options.....	137
Return Codes.....	139
Comparison Toolset PDF	140

Manage data and schema changes

- Compare and synchronize data and schema objects
- Easy navigation of user interface improves efficiency
- Automate comparison and synchronization operations
- Customize schema compare and synchronization sessions
- Generate ready-to-use database synchronization scripts

Release notes

Idera SQL Comparison Toolset is a bundle of two products, Idera Schema Compare and Idera Data Compare. While Schema Compare allows you to compare database schemas and propagate schema changes from one environment to another, Data Compare lets you compare and synchronize tables across servers, databases, and versions when performing data migrations, copying databases, or auditing data.

To get a quick glimpse into the newest features, fixed issues, and known issues in this release of Idera SQL Comparison Toolset, review the following sections of the Release Notes:

- [New features and fixed issues](#)
- [Review issues fixed by this release](#)
- [Review previous features and fixed issues](#)
- [See known issues](#)

New features and fixed issues

Idera Comparison Toolset provides the following new features and fixed issues.

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: $90 < \text{char} < 97$. 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.

Previous features and fixed issues

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

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 are 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 constraints on these tables. Idera SQL Schema Compare supports the natively-compiled functions. Note 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 than 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 a 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-transactional 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 addition 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 “SERIALIZED” 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.

Known issues

Idera strives to ensure our products provide quality solutions for your SQL Server needs. The following known issues are described in this section. ***If you need further assistance with any issue***, please contact [Idera Support](#).

Known issues

Two different databases appear equal to some users

Idera SQL Comparison Toolset includes a number of options that can cause the following issues:

- Idera SQL Schema Compare index comparison options may “hide” differences between two different indexes and make them appear equal. Currently, Idera SQL Schema Compare supports seven index options available by clicking **Comparison Options > Index Options**. If an option is unchecked, the property affected by it is not compared.
- When check constraints have the same name on different tables, they may not be identified as different when using Idera SQL Schema Compare. For example, the check constraints would look as follows:

LOC_TX.g_1835_ck

LOC_CA.g_1835_ck

Note the following operation of Idera SQL Schema Compare:

- a. SQL Server supports two types of check constraints: column-constraints and table-constraints. A column constraint is applied to one column only while a table constraint affects multiple columns.
- b. Table-constraints have a table-scope, i.e. the constraints on the left database are compared with constraints of the same table on the right database.
- c. Column constraints have a table and column scope, so constraints on the left database are compared with the constraints of the same table, under the same column, on the right database.

Using the example, the constraints of the LOC_TX table on the left database are compared with those of the LOC_TX table on the right database. Constraints of LOC_TX and LOC_CA are not mixed together.

Idera SQL Schema Compare can ignore the name of a check constraint if the option **Ignore check constraint name** is checked. This option affects only column constraints and it is located under **Comparison Options > Constraint Options**.

Comparison grid appears to not work properly if a specific option is checked

To some users, it appears as if the **Select All / Clear All** feature in the comparison grid does not work properly. An object still can appear in the synchronization script even after being unchecked in the comparison grid. This behavior is by design and has to do with the dependencies in the database. To ensure that an object is excluded permanently, the user must uncheck the comparison option **Include dependent objects**.

Apparent error during comparison

Some users may see an error occurring during object pairing before the schema comparison. This issue usually results from a database created with case-sensitive collation and containing objects with the same name but different case, such as **Employee** and **employee**. Note that while this error is triggered, Idera SQL Schema Compare does continue to compare the databases. However, from the set of objects with the same name, only one of them is compared and the remainder is ignored.

Welcome to SQL Comparison Toolset

Idera SQL Comparison Toolset provides database administrators and developers with the tools they need to safely and efficiently manage SQL Server database changes.

This wiki contains useful information that will help you get started and take full advantage of the comparison and synchronization tools within the Toolset. The products covered in this wiki include:

- [IDERA SQL Schema Compare](#)
- [IDERA SQL Schema Compare Command Line](#)
- [IDERA SQL Data Compare](#)
- [IDERA SQL Data Compare Command Line](#)

Everything in this wiki, unless explicitly stated otherwise, applies to all editions of the Comparison Toolset.

Getting started

Idera SQL Comparison Toolset contains:

- **Schema Compare** for comparison the structure of SQL Server databases.
- **Schema Compare Command Line** for comparing the database structure via the command line.
- **Schema Compare Command Line Wizard** for managing schema compare command line configurations.
- **Data Compare** for comparing the content of SQL Server databases.
- **Data Compare Command Line** for comparing the database content via the command line.
- **Data Compare Command Line Wizard** for managing data compare command line configurations.

For two weeks from the installation date the products are fully functional and have no restrictions. After the trial period expires, you must acquire a license. You can purchase a license from our website at [Idera.com](https://www.idera.com).

Installing and Uninstalling the Toolset

i Generally, you can install new builds of the Idera SQL Comparison Toolset on top of a previously-installed version, without having to uninstall the older version first.

Installing SQL Comparison Toolset

When you download the Idera SQL Comparison Toolset from our web site, you receive an .msi setup file that you can run to install the toolset. The installation creates the Idera SQL Comparison Toolset group in the Start Menu, and places **Idera SQL schema compare** and **Idera SQL data compare** shortcuts on the desktop.

Uninstalling SQL Comparison Toolset

To uninstall the SQL Comparison Toolset

Use one of the following processes:

Process 1

1. Go to **Control Panel > Programs > Uninstall a Program**.
2. Select **Idera SQL Comparison Toolset**.
3. Click **Uninstall/Change**, and then follow the prompts to complete the uninstall.

Process 2

1. **If you still have the original .msi file**, double-click the file.
2. Click **Remove**, and then follow the prompts to complete the uninstall.

Requirements

Minimum requirements for running Idera SQL Comparison Toolset are:

- .NET Framework 4.6 or higher, available from the [Microsoft Developer Tools Download Center](#)
- at least 50MB of free space on the hard drive where the application will be installed (depends on the size of the databases you want to compare and synchronize)
- Windows XP SP3 or later / higher versions of Windows operating systems
- minimum of 512MB of memory; 1GB or higher recommended
- minimum resolution of 1024x768 suggested for the GUI

Activating a license

To activate a license for the Idera SQL Comparison Toolset, go to **Start > Idera SQL Comparison Toolset > License Activation**. In the dialog window that appears, enter the license number you received and then click **Activate License**. The license activate utility will connect to a web service running on our web servers that will validate the license you provided. Within a few seconds you should see a confirmation message indicating that the product was successfully upgraded.

The screenshot shows a dialog box titled "Idera SQL comparison toolset" with a subtitle "License Activation" and the instruction "Activate your license for Idera SQL comparison toolset." The dialog is divided into two main sections: "License Information" and "Request File".

License Information: This section contains two radio buttons. The first, "Activate license online", is selected. Below it are four text input fields: "New License", "Previous License", "Name", and "Email". The second radio button, "Load license from file", is unselected and has a file selection button next to its input field. An "Activate License" button is located at the bottom right of this section. A link "View collected information" is also present.

Request File: This section contains a heading "Request File" and a paragraph: "If you are experiencing difficulty activating the product, please follow these four steps:". Below this are four numbered steps: 1. Enter the License Number into the "New License" edit box. 2. Click on the "Generate Request File" button. 3. Email the file to licensing@idera.com. 4. Once you have received the license activation file, you may upgrade the product using the option "Load license from file". A "Generate Request File" button is located at the bottom right of this section.

A "Close" button is located at the bottom center of the dialog box.

Supported platforms

Idera SQL Comparison Toolset supports the following on-premise and versions of SQL Server:

- Microsoft Azure SQL Database
- SQL Server 2005, 2008, 2008 R2, 2012, 2014, 2016, 2017, 2019

Idera SQL Data Compare also supports SQL Server 2000.

Updates and Support

We update our products periodically but will only notify our customers via email when we publish an update that contains a critical fix or that is considered a major upgrade. to ensure that you are always running the latest version of our products, we recommend that you check our site [idera.com](https://www.idera.com) regularly for product updates.

IDERA SQL Schema Compare

Welcome to IDERA SQL Schema Compare, the tool of choice for thousands of DBAs and developers around the globe. Identifying schema differences between two databases and synchronizing those schemas is easy, fast, and safe with SQL Schema Compare.

Please direct any questions, comments and suggestions about the IDERA SQL Schema Compare to support@idera.com.

SQL Schema Compare Application window

IDERA SQL Schema Compare divides the main application window into three areas:

1. **The ribbon.** The context sensitive ribbon provides all the command buttons and icons you need to do your job quickly. The ribbon is organized in three distinct tabs each of which corresponding to the main steps of the database schema comparison and synchronization process and containing the relevant command buttons.
2. **Main panel.** The main panel contains a permanent tab called the [Workspace](#) which you cannot close, and a tab for each active comparison session.
3. **Output window.** The output window provides a running log of the tasks being performed in IDERA SQL Schema Compare.



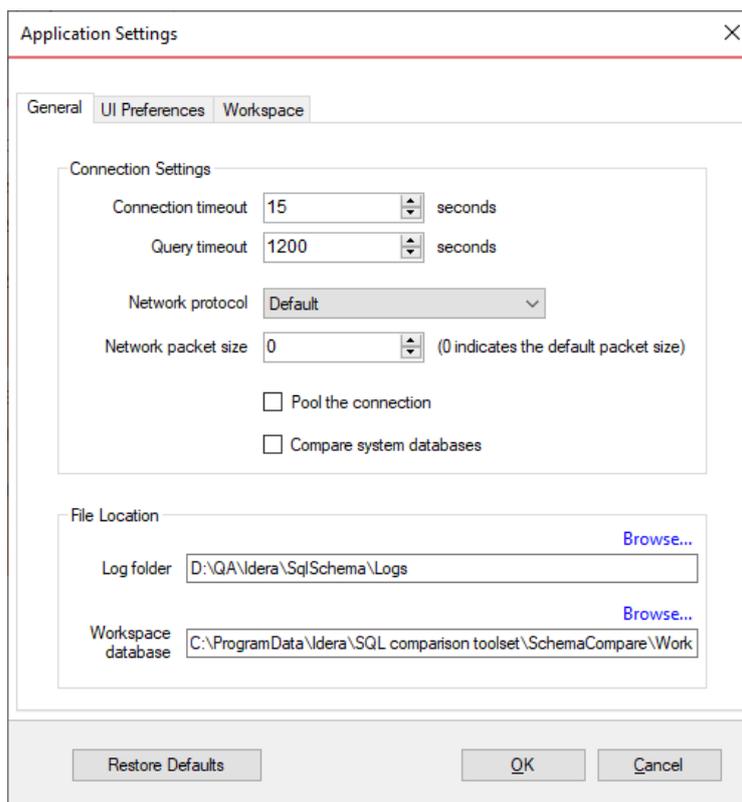
SQL Schema Compare Application Settings

Application settings allow you to tweak the behavior of IDERA SQL Schema Compare. It can be accessed from the drop-down menu off of the top-left corner product icon or from the link at the bottom-left corner on the Workspace tab.

Application level settings are grouped into the General, UI Preferences, and Workspace tabs.

General

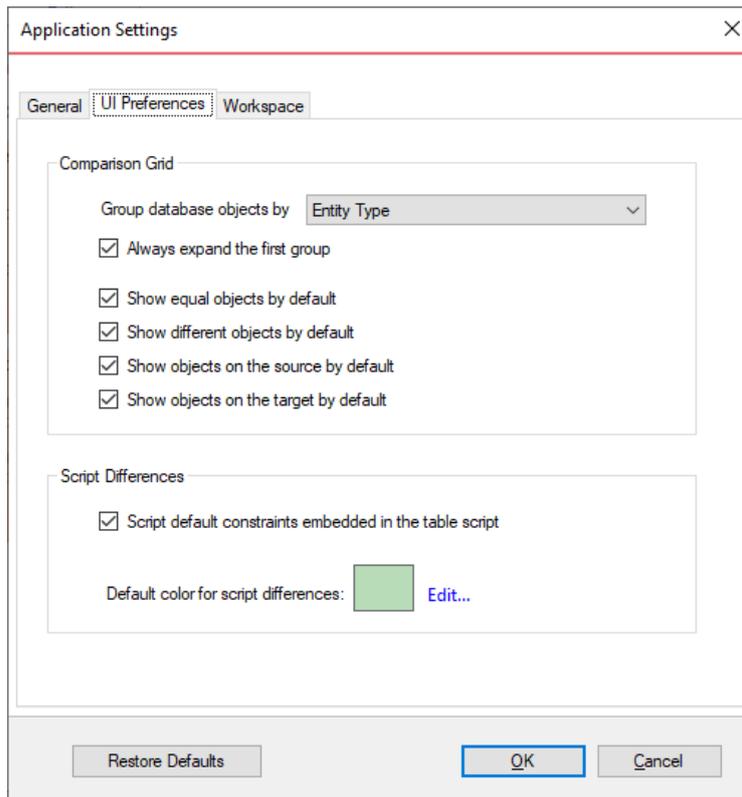
- **Connection Settings.** Generally, there is no need to change the connection settings, however, in certain scenarios you might need to adjust those values. From the application settings window you can set the **Connection Timeout**, **Query Timeout**, **Network Protocol**, and **Network Packet Size**.
- **Log folder.** Indicates the location where the IDERA SQL Schema Compare log files are saved.
- **Workspace database.** The workspace database file, which stores SQL Server connections, comparison settings, and more.



UI Preferences

- **Comparison Grid.** Allow you to adjust the behavior of the comparison results grid based on your preferences.

- **Script Differences.** Allow you to change the highlight color for the script differences as well as determine how the default constraints are scripted. The way the default constraints are scripted has no bearing on the comparison results or on the synchronization script, the only difference is on where you see those constraints scripted. By embedding the default constraints in the table script it is easier to see why two columns might be marked as different. If the default constraints are scripted separately then you might see two seemingly identical columns marked as different and you will need to scroll down below the table script to see what is different.

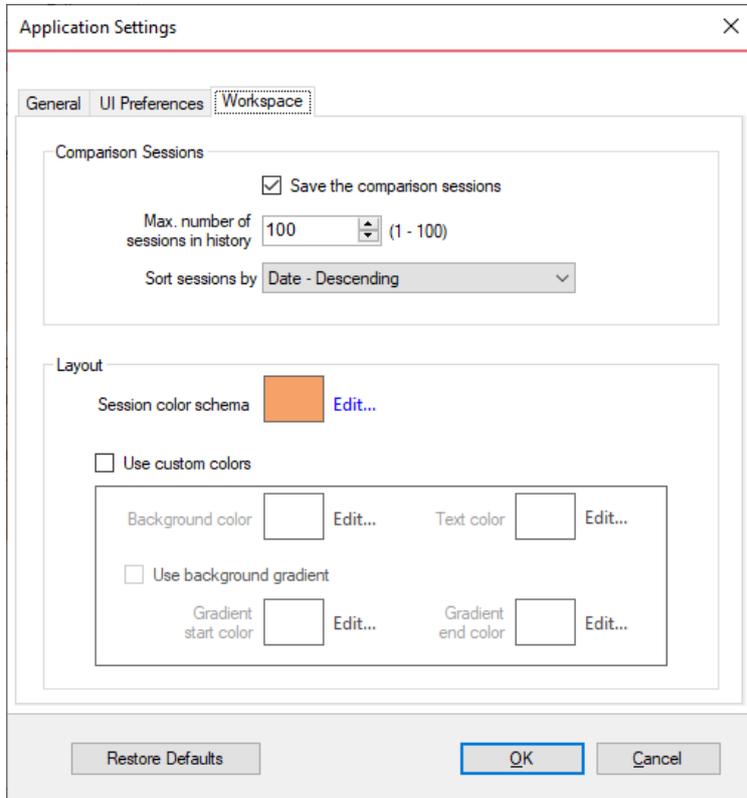


Workspace

Contains settings related to the workspace and comparison sessions.

- **Sessions**
 - **Save sessions.** By default is checked. It instructs the IDERA SQL Schema Compare to save comparison sessions for future use. A saved session contains all the necessary information so that you can rerun the comparison with one click. Saved sessions appear on the main panel of the Workspace tab.
 - **Max number of sessions in history.** By default, IDERA SQL Schema Compare keeps the last 25 sessions. Although there is no noticeable performance difference on application launch related to the number of sessions stored you may choose to set this parameter to a lower number. Allowable values are 1 to 25.

- **Sort sessions by.** Use this option to change the order in which the stored sessions appear in the Workspace tab.
- **Layout**
 - Allows you to custom session coloring. You can select from built-in color schemas or pick custom colors.



SQL Schema Compare Workspace

IDERA SQL Schema Compare Workspace is the storage and the interface for saving and managing databases, credentials, comparison sessions, and other UI options.

The Workspace window is divided into the following sections:

1. **Workspace actions.** Provides links that allow you to create a new workspace, add databases to the current workspace, compare schema, or change application settings. The **New Workspace** action resets the current workspace by erasing the comparison history and removing all registered databases.
2. **Databases/Snapshots.** Contains the list of databases and snapshots that have been registered in the workspace. On mouse over two action links appear for a registered server:
 - **Edit.** Allows you to change the connection settings for that server and/or add and remove databases.
 - **Remove.** Un-registers a server (deletes all the server information from the workspace).
When you move your cursor over a database name, the following action links related to that database appear:
 - **Take a Snapshot.** Allows you to take a snapshot of the schema of that database.
 - **Remove Database.** Removes the database from the workspace.
3. **Comparison History.** Contains the comparison sessions in the order specified in the [Application Settings](#). Schema Compare provides one-click comparison, which allows you to return a comparison just by clicking a saved session.
4. **Output.** Displays each workspace event as it occurs.



Workspace Storage

The workspace stores its settings in a database file, which by default is: **C:**

\ProgramData\Idera\SQL comparison

toolset\SchemaCompare\Workspace\workspace.db. If you prefer to use a different file on a different folder, do the followings:

- Copy the default database file to a folder of your choosing. Rename it if you wish to do so.
- Launch Schema Compare, open the [Application Settings](#), and then change the workspace database to the file you created.

i The workspace requires write permissions to its database file. If it can't write to its database, it will display a read-only message.

SQL Schema Compare Add Databases

Before comparing databases, you will need to add them to the workspace. The **Add Databases** functionality can be accessed from the ribbon (the **Home** tab) or from the **Add Databases** action link at the top left section of the Workspace tab.

On the Add Database dialog window, choose the server where you want the new database(s) to reside. You can click on the **Refresh** item of the SQL Server drop down to discover the SQL Server instances running on your network. If the application is not able to discover the SQL Server instance you are looking for because it is not exposing itself or it is not running on your local area network, or it is not listening on the default port, then you can type in [ServerName or IP Address]\[InstanceName] on the drop-down box.

i If the SQL Server is not listening on the default port you can also specify a custom port on the following format: [ServerName]\[Instance Name], [PortNumber]

Once you have chosen the server and the authentication method (including login name and password if you chose SQL Server Authentication), you can click **Read Databases** at the top of the Databases box at which point the application will connect to the server using the information you provided and populate the Databases box with the list of all user databases in that server. You can select one or more databases to add to the workspace.

Under the Advanced Settings tab, you can enter various connection settings, such as the timeouts, the connection protocol, packet size and a few more. By default schema compare uses the advanced settings specified in the [Application Settings](#).

The **Other connection settings** allows to specify additional connection properties as `name=value`, separated by semicolons: `name1=value1;name2=value2;...` To check whether these properties are allowed for a SQL Server connection, click **Check Settings**. For a full list of the connection properties supported by a SQL Server database, check the MSDN.

The following property names are already included in the SQL Server connection and should not be specified in the **Other connection settings**. Their synonyms must be excluded as well:

- Data Source
- Initial Catalog
- Integrated Security
- Persist Security Info
- User ID
- Password
- Pooling
- Connect Timeout
- Network Library
- Packet Size
- Application Name

The screenshot shows a dialog box titled "Add databases to the workspace". It has a close button (X) in the top right corner. The dialog is divided into several sections:

- SQL Server Instance:** A dropdown menu showing "DEVSRV003\SQL2019" and a "Clear All" button.
- Friendly Name:** A text input field containing "SQL 2019".
- Authentication Type:** A dropdown menu showing "Windows Authentication".
- Username:** An empty text input field.
- Password:** An empty text input field with a "Show Password" button to its right.
- Advanced Settings:** A section with a "Databases" tab and an "Advanced Settings" tab.
 - Connection Timeout:** A spinner set to "15" with the unit "seconds" and a "Restore Defaults" button.
 - Query Timeout:** A spinner set to "1200" with the unit "seconds".
 - Protocol:** A dropdown menu set to "Default".
 - Packet Size:** A spinner set to "0" with the text "(0 default)" to its right.
 - Pooled Connection:** A checkbox that is currently unchecked.
 - Check Settings:** A button located to the right of the "Other connection settings" text area.
 - Other connection settings:** A text area containing the text "ConnectRetryCount=2;ConnectRetryInterval=5". Above this text area is a small explanatory text: "Other connection settings, specified as name=value, separated by semicolons: name1=value1;name2=value2;..."

At the bottom of the dialog are "OK" and "Cancel" buttons.

SQL Schema Compare Add Snapshots

A schema snapshot is a compact file that contains all the schema information for a database. The schema information is stored in a proprietary format and can only be read by IDERA SQL Schema Compare. The schema snapshot does not contain data, only schema information.

Schema snapshots allow you to maintain a history of schema changes. You can take schema snapshots at different points in time and then you can compare those snapshots to each other or compare the snapshots to the live database to get a clear an accurate picture of the "evolution" of the database schema.

Take a snapshot

To take a schema snapshot, simply move the mouse over the name of the database on the left panel of the Workspace and you will see a "camera" icon appear on the side of the database name. Click on that icon to take the snapshot - you will be asked to choose the name of the snapshot and the location where you wish to save it.

Adding snapshots to the workspace

Before comparing snapshots to each other or to live databases you will need to add the snapshots you need to the workspace. The **Add Snapshots** functionality can be accessed from the ribbon (the Home tab).

On the Add Snapshots dialog window that appears, click **Browse...** to browse to the location where the snapshot(s) you wish to add to the workspace have been stored. You can select as many snapshots as you want from different locations. For each snapshot you select the application will display the file name, the database the snapshot was taken from, the date when the snapshot was taken, and SQL Server version.

Add database snapshots to the workspace ✕

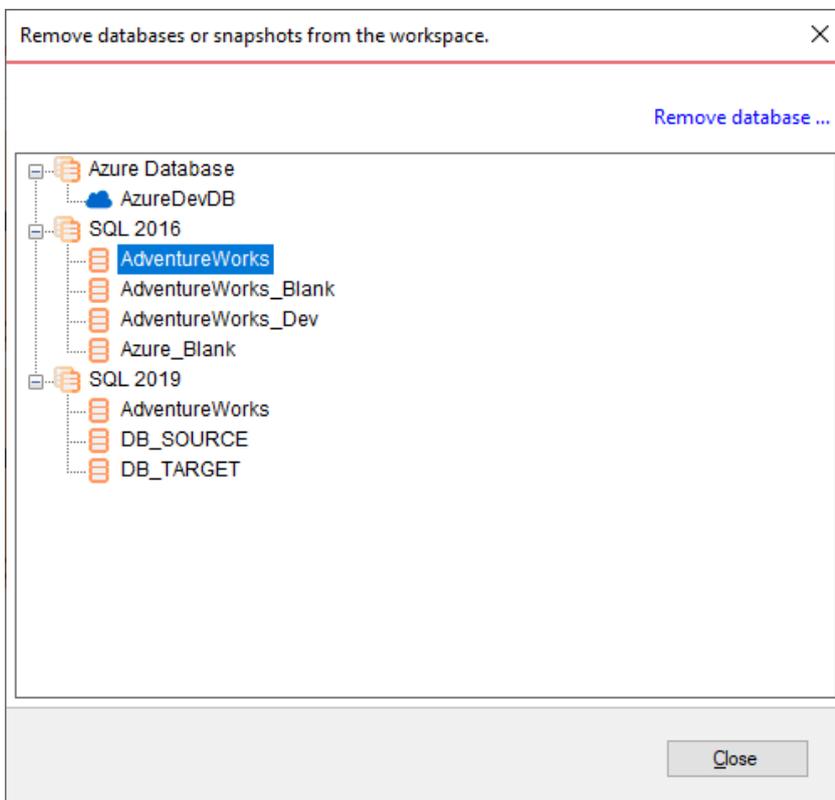
[Browse...](#) [Clear List ...](#)

SQL Server	Database	Date Created	Filename
DEVSRV002\SQL2016	AdventureWorks	Sep 04, 2019 14:20:55	AdventureWorks.snpx
DEVSRV002\SQL2016	AdventureWorks_Blank	Sep 04, 2019 14:21:04	AdventureWorks_Blank.snpx
DEVSRV002\SQL2016	AdventureWorks_Dev	Sep 04, 2019 14:21:12	AdventureWorks_Dev.snpx
DEVSRV003\SQL2019	DB_SOURCE	Sep 04, 2019 09:23:40	SQL2019.Schema_Source.snpx
DEVSRV003\SQL2019	DB_SOURCE	Sep 04, 2019 14:34:12	DB_SOURCE.snpx
DEVSRV003\SQL2019	DB_TARGET	Sep 04, 2019 09:24:05	SQL2019.Schema_Target.snpx
DEVSRV003\SQL2019	DB_TARGET	Sep 04, 2019 14:34:22	DB_TARGET.snpx

SQL Schema Compare Remove Databases

There are two ways to remove a database or a schema snapshot from the workspace:

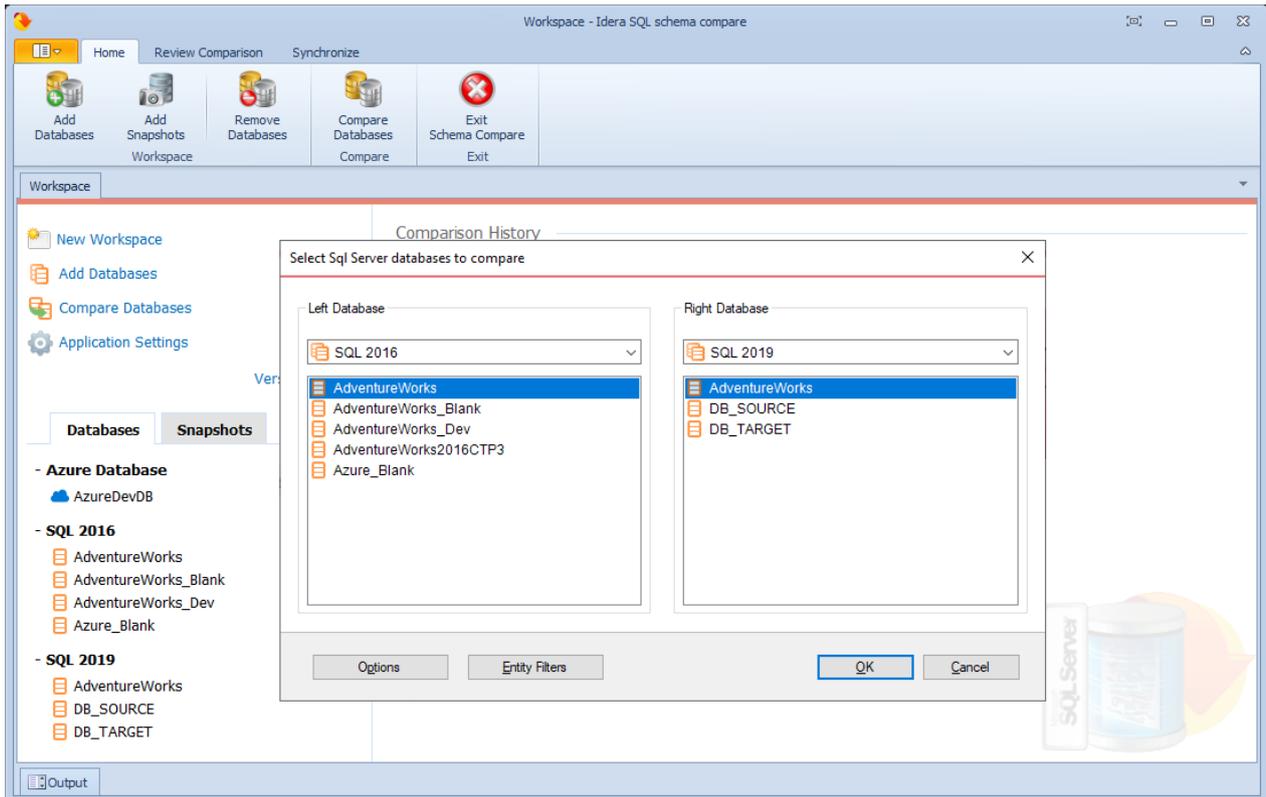
- Move the mouse over the name of database or the schema snapshot you wish to remove under the Registered Databases on the left panel of the Workspace. The database/snapshot name will be highlighted and a **Remove** icon/link will become visible. Simply click on that icon and confirm the removal when prompted. Note, that in the same way you can remove a registered server and all its databases from the workspace.
- Click **Remove Databases** on the Home tab of the ribbon. A remove databases dialog window appears showing all the registered servers/databases - select the server or database you wish to remove and click on the **Remove Selected SQL Server / Database...** link at the top right corner and then confirm the removal when prompted.



SQL Schema Compare Comparing Schemas

To use SQL Schema Compare to start a comparison between two databases, click **Compare Databases** on the Home tab of the ribbon. The SQL Server Database Selection dialog window appears and allows you to choose the server(s) where the databases you wish to compare reside as well as the databases themselves.

From this window, you can also access the Comparison Options that will be applied during the comparison.



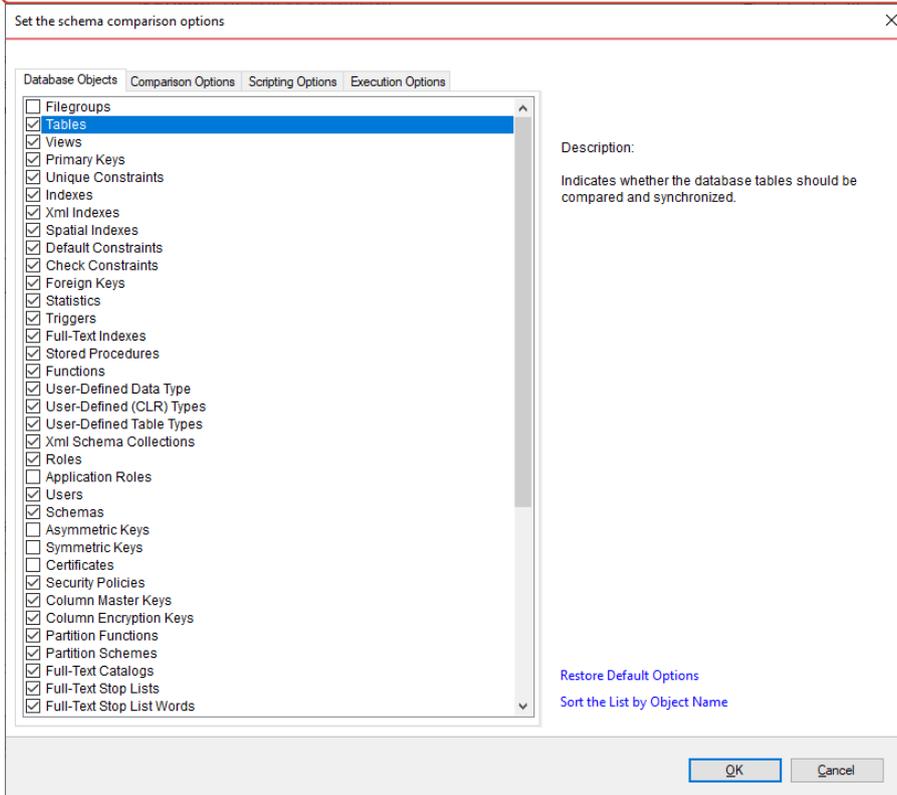
SQL Schema Compare Object Types

IDERA SQL Schema Compare supports over 45 different types of database objects. By default, all but a handful of object types will be included in the comparison and synchronization.

In the majority of the cases, you should not need to change the default settings. There are scenarios in which you might wish to exclude certain categories that are included by default or include those that are excluded by default.

To exclude or include certain object types, click **Comparison Options**. The first tab on the Comparison Options window that appears is the Database Objects tab which provides a brief description for each object type and allows you to select or clear the object types you want.

⚠ Because of the dependencies between database objects, excluding all objects of a given type may result on a synchronization script that will fail to execute due to invalid dependencies. For more information about the handling of dependencies, see [SQL Schema Compare Comparison Options](#).



SQL Schema Compare Options

IDERA SQL Schema Compare exposes over 60 different options that provide the user with highly granular control of the behavior of the comparison and synchronization engine.

The defaults have been carefully selected to match the most common scenarios so that in the majority of the cases you don't even need to look at those options at all, however, every SQL Server implementation is different and there is no "one size fits all" scenario.

For clarity and easy access, the options have been organized on three different tabs:

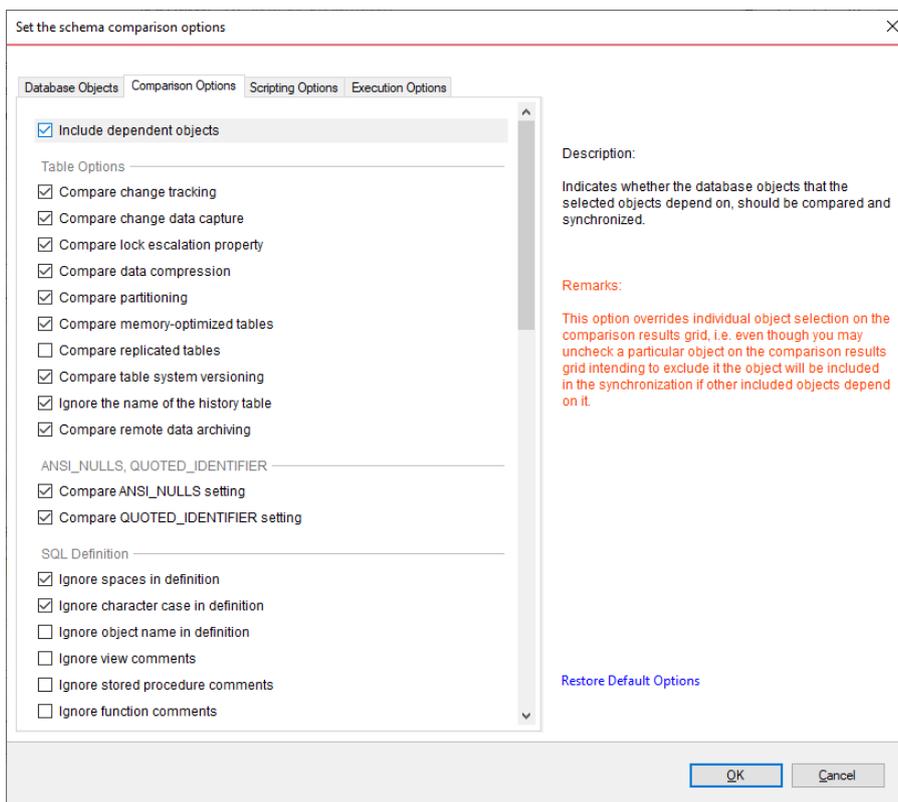
- [Comparison Options](#)
- [Scripting Options](#)
- [Execution Options](#)

SQL Schema Compare Comparison Options

When are two database objects equal? The answer is: it depends! It depends on what you want to compare those objects on - for example: is "Equal" equal to "equal"? The comparison options allow you to have complete control over the comparison engine which considers those options before deciding to mark a pair of objects as equal or different.

So, when two objects are found to be different by the comparison engine while you believe they should be equal, or they are found to be equal while you believe they should be different don't panic - investigate the differences and carefully review the comparison options - more likely than not one of those options holds the answer to your surprise.

The comparison options tab provides a description and important remarks for each option while also allowing you to turn those options ON or OFF. To view the description and remarks for a given option simply move the mouse over that option as shown on the image below.



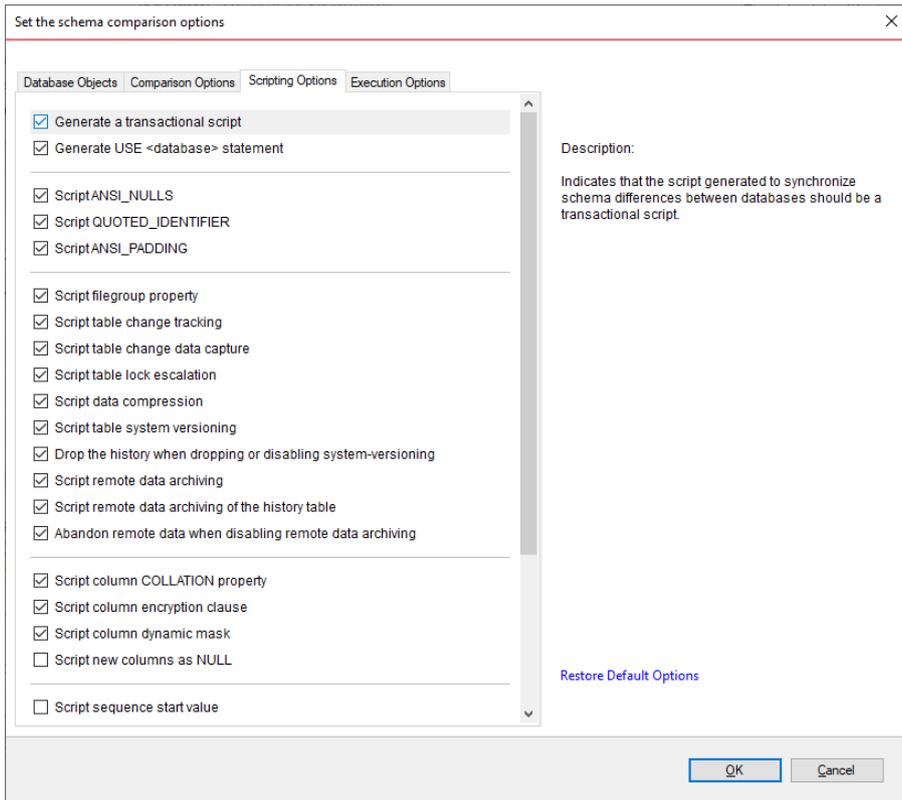
SQL Schema Compare Scripting Options

Scripting options allow you to control the way objects are scripted in the database synchronization script. You can choose, for example, not to script the table change tracking, script the start and end value of a sequence, omit the filegroup clause of database objects and many more.

i If you are not sure what options would be best for your scenario, click **Restore Defaults**, and then use the default options.

Most of the scripting options are self-explanatory, and for those that are not, the description provided in the Comparison Options window is sufficient. However, there are two options that merit special attention:

- **Script New Columns As NULL.** When the synchronization process requires adding a NOT NULL column on the target database, but no default constraint is specified for that column, SQL Server rejects the new column and the synchronization fails. In such scenarios the best course of action would be to add a default constraint on the source column. If such an action is not possible, you can add the new column to the target database as a NULL ALLOWED, and then, after the synchronization, connect to the target database via SSMS and change that particular column from NULL ALLOWED to NOT NULL.
- **Script New Constraints WITH NOCHECK.** When the synchronization requires adding new constraints, such as a foreign key or a check constraint, on a table, the existing data may violate those constraints, in which case SQL Server rejects the action and the synchronization script fails. To overcome this obstacle, you can choose to add those constraints the WITH NOCHECK clause first, and then verify if the existing data violates the constraints. When this option is ON, Schema Compare generates the following two synchronization scripts which are executed one after the other:
 - **Main synchronization script.** Creates the new constraints with NOCHECK so that no data violation occurs and databases are synchronized successfully.
 - **Constraints enabling script.** Enables the constraints that were created WITH NOCHECK and runs after the main script, on a separate transaction. If the existing data violates these constraints, then the script fails, but the failure does not affect the main database synchronization. In a case of failure, you must address the constraint violations on a case-by-case basis.



SQL Schema Compare Execution Options

IDERA SQL Schema Compare exposes a wide variety of options to let you tweak the way the comparison engine works. From ignoring white spaces to performing automated file group translation you have control over it all. The default options have been carefully selected to match the most common scenarios however, every SQL Server implementation is different and there is no "one size fits all" scenario.

When an object is found to be different by the comparison engine while you believe it should be equal, or the other way around, don't panic - investigate the differences and carefully review those options - more likely than not one of those options holds the answer to your surprise.

Rule of thumb: whenever you are not sure what options would be better for your scenario, click **Restore Default Options**.

The screenshot shows a dialog box titled "Set the schema comparison options" with a close button (X) in the top right corner. The dialog has four tabs: "Database Objects", "Comparison Options", "Scripting Options", and "Execution Options". The "Execution Options" tab is active. Inside the dialog, there are three checkboxes: "Execute a transactional script" (checked), "Continue script execution on error" (unchecked), and "Log script execution" (unchecked). Below the "Execute a transactional script" checkbox is a text box labeled "Transaction size (in MB)" with the value "0" and a spin button, followed by the text "(up to 2048 MB)". To the right of these options is a "Description:" section with the text: "Indicates that the synchronization script executed by Schema Compare should be transactional. If the max size is reached, the transaction that is active commits and a new one starts." Below the description is a "Remarks:" section with red text: "Use this option carefully. Turning off the transactional option would make it impossible to reverse schema changes if an error occurs during the database synchronization." and "0 indicates an unlimited transaction." At the bottom of the dialog, there is a "Restore Default Options" link and two buttons: "OK" and "Cancel".

SQL Schema Compare Entity Filters

Entity filters allow you to exclude database objects based on some predefined criteria. You could exclude, for example, tables whose name starts with TEST_, views ending with "_TEMP", procedures by name, function using regular expressions and more.

There are many ways to launch the entity filters:

- The **Entity Filters** button, in the Database Selection form, before starting the comparison.
- In the comparison form, the **Entity Filter** link located at the Ribbon
- In the comparison form, the **Entity Filters** from the context menu.

Schema Compare creates filters for specific object types, such as tables, views, procedures. A filter is set of criteria, combined with the OR/AND operator. If you wish to exclude, for example, all tables whose name starts with DEV_ or TEST_, you could create the following filter (notice the OR operator):

Table filter: name starting with "DEV_" OR name starting with "TEST_"

If you wish to exclude all views, except V_EMPLOYEES and V_JOBS, the filter would be as follows (notice the AND operator):

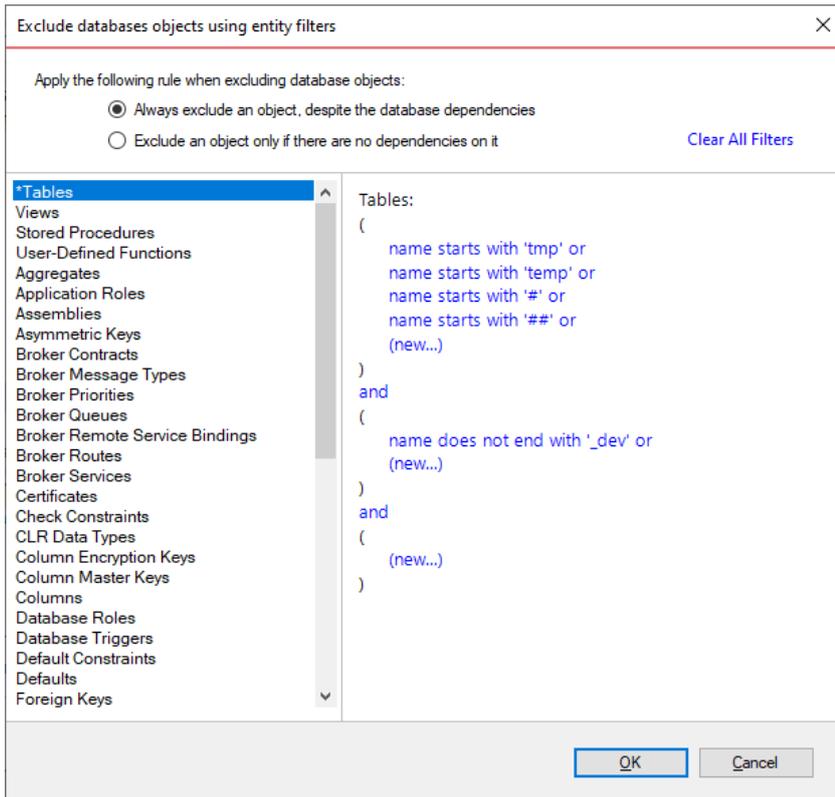
View filter: name != "V_EMPLOYEES" AND name != V_JOBS

You create an entity filter as follows:

1. Launch the **Entity Filters** from the Ribbon link, the **Comparison Form** context menu, or via the **Entity Filters** button in the database selection form.
2. Select the object type, for which the filter is intended, such as the **Tables** or **Views**.
3. Click on the **(new...)** link, and then create the filter criteria. Add as many criteria as you need.

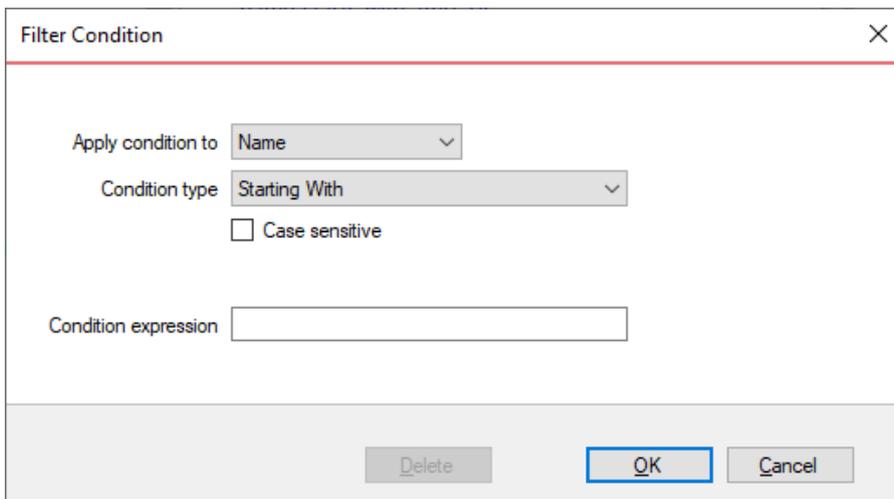
To change the criteria operator from **OR** to **AND** or vice versa, simply click on one operator.

 For a more complex filter, that combines criteria with both OR and AND operators, create multiple groups. Place the OR-criteria in one group and the AND-criteria in another.



Filter condition

Condition is the filter criteria that an object name or an object schema is checked against. If the object meets the filter criteria, it is excluded from the comparison. A condition can be applied on the name or on the schema of an object.



SQL Schema Compare Comparison Results

Upon completion of the schema compare operation, the results of the comparison are displayed on a new comparison session tab. The Comparison Results window is divided into three main sections:

1. **Comparison summary.** The main section displays the list of all objects that were compared, grouped by object type, indicating for each object the result of the comparison (equal, different, exists only on one of the databases). By default, all the objects are displayed including the ones that are identical, however, a handy context menu allows you to filter objects based on object type or/and based on the result of the comparison.
2. **Differences.** The bottom section of the results window contains three tabs.
 - The Differences tab displays, side by side, the scripts of the object pair that has been selected in the top section of the results grid in both of the databases being compared. The differences are highlighted using a simple color coding scheme.
 - The Left Database Script contains the t-sql script that you would need to execute on the left database to make the selected object on the left side the same as the corresponding object on the right side.
 - The Right Database Script contains the t-sql script that you would need to execute on the right database to make the selected object on the right side the same as the corresponding object on the left side.
3. **Action.** The right panel provides quick access to action buttons/links that allow you to generate synchronization scripts, view/change comparison options, refresh comparison results etc.

Comparison Result

Synchronize	Entity Type	Schema	Name	Status
Table object(s): 70, Equals: 65, Different: 5, Source Only (left): 0, Target Only (right): 0				
	Table	dbo	AWBuildVersion	=
	Table	dbo	ErrorLog	=
	Table	HumanResources	Department	=
<input checked="" type="checkbox"/>	Table	HumanResources	Employee	↔
	Table	HumanResources	EmployeeAddress	=
	Table	HumanResources	EmployeeDepartmentHistory	=
	Table	HumanResources	EmployeePayHistory	=
	Table	HumanResources	JobCandidate	=
	Table	HumanResources	Shift	=
<input checked="" type="checkbox"/>	Table	Person	Address	↔
	Table	Person	AddressType	=
<input checked="" type="checkbox"/>	Table	Person	Contact	↔

Schema Differences

Left Database Script	Right Database Script
<pre> 1 -- -Main Script - Different 2 CREATE TABLE [HumanResources].[Employee] 3 (4 [EmployeeID] [int] IDENTITY(1,1), 5 [NationalIDNumber] [nvarchar](15) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 6 [ContactID] [int] NOT NULL, 7 [LoginID] [nvarchar](256) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 8 [ManagerID] [int] NULL, 9 [Title] [nvarchar](50) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 10 [BirthDate] [datetime] NOT NULL, 11 [MaritalStatus] [nchar](1) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 12 [Gender] [nchar](1) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 13 [HireDate] [datetime] NOT NULL </pre>	<pre> 1 -- -Main Script - Different 2 CREATE TABLE [HumanResources].[Employee] 3 (4 [EmployeeID] [int] IDENTITY(1,1), 5 [NationalIDNumber] [nvarchar](15) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 6 [ContactID] [int] NOT NULL, 7 [LoginID] [nvarchar](256) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 8 [ManagerID] [int] NULL, 9 [Title] [nvarchar](55) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 10 [BirthDate] [date] NOT NULL, 11 [MaritalStatus] [nchar](1) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 12 [Gender] [nchar](1) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL, 13 [HireDate] [datetime] NOT NULL </pre>

SQL Schema Compare Working with Filegroups

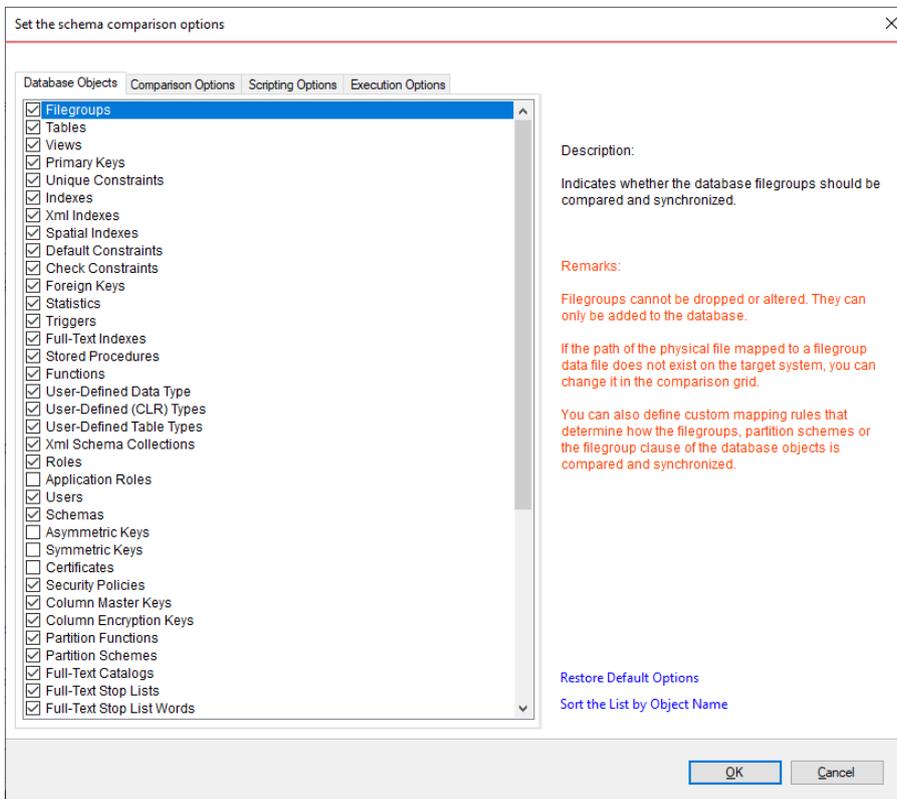
The following topics describe the filegroup options in detail:

- [Comparing and Synchronizing Filegroups and Data Files](#)
- [Working with Filegroup Mapping Rules](#)

SQL Schema Compare Comparing Filegroups

Idera SQL Schema Compare allows you to compare and synchronize database filegroups including the filegroup data files. Database log files are not supported.

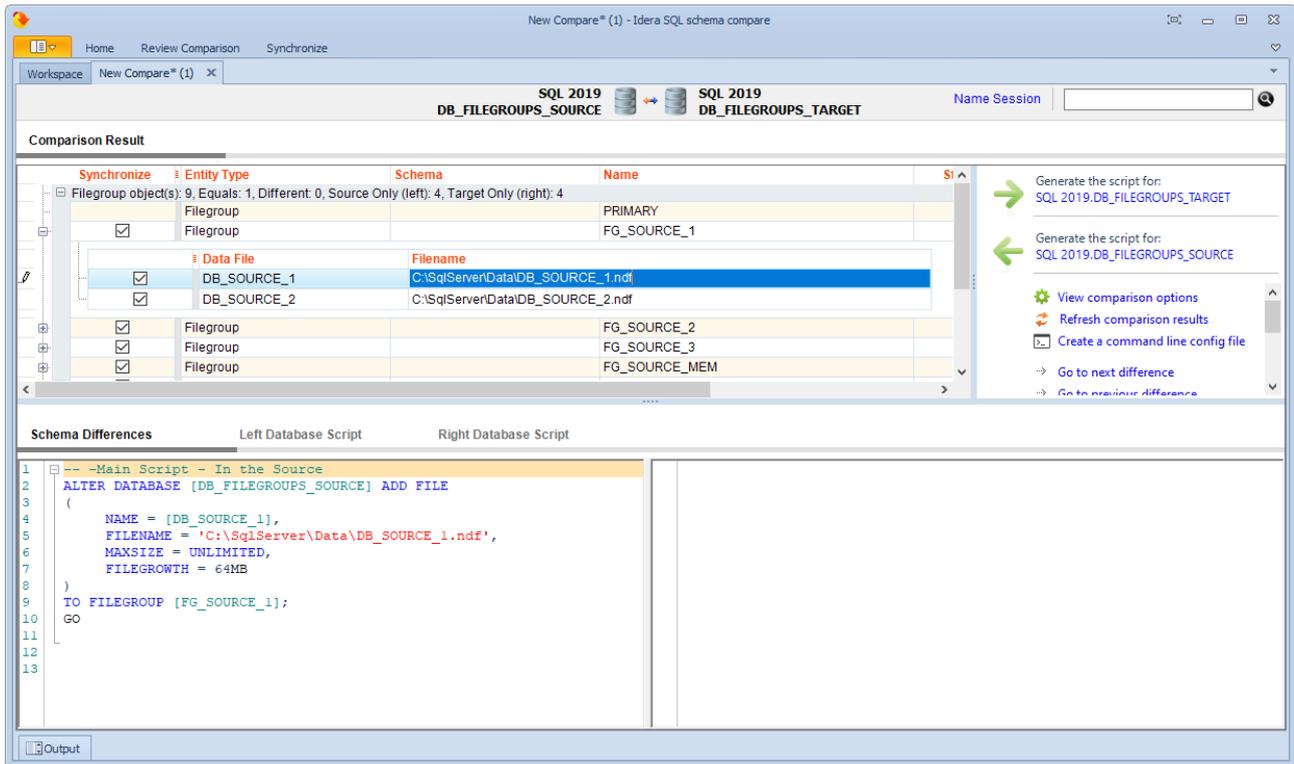
Comparing filegroups is similar to other database objects. You start by selecting the **Filegroups** option in the Comparison Options Database Objects window:



When the filegroup option is selected, a filegroup category appears in the comparison grid. The plus sign adjacent to the filegroup allows you to view the data files associated with the filegroup.

When you choose to synchronize the database filegroups, it is important to remember that the path of the physical file mapped to the data files must exist on the target system. If this is not the case, you can change the path and filename of the data file in the comparison grid as follows:

- Expand the filegroup that you wish to change
- In the data file section of the comparison grid, enter the new filename in the **Filename** column



You can also exclude filegroups or specific data files, similar to other database objects. Excluded items will not be synchronized.

The following restrictions apply to filegroup synchronization:

- Filegroups can only be added to the database. They cannot be altered or removed.
- Data files are only added to the new filegroups. Data files on existing filegroups are not modified.
- SQL Server does not support transactional execution of DDL statements for filegroups or data files, which means that changes to the filegroup structure cannot be rolled back.

SQL Schema Compare Filegroup Mappings

When the source and the target database have a different filegroup structure, Idera SQL schema compare allows you to map the filegroups with each other. Filegroup mapping can be accessed from the main ribbon or the comparison grid context menu.

About filegroup mapping

Filegroup mapping provides a method to map the source and the target filegroups with each other, so that filegroup differences do not require database synchronization. Consider, for example, what happens when you compare the following databases:

Database: DB_1

Contains the filegroup: FG_1

Contains the Employees table created on FG_1 filegroup

Database: DB_2

Contains the filegroup: FG_2

Contains the Employees table created on FG_2 filegroup

The default comparison finds a schema difference in the Employees table triggered by the filegroup clause. The script generated for the DB_2 database attempts to create the table on FG_1 filegroup, while the script generated for DB_1 attempts to create the table on FG_2 filegroup.

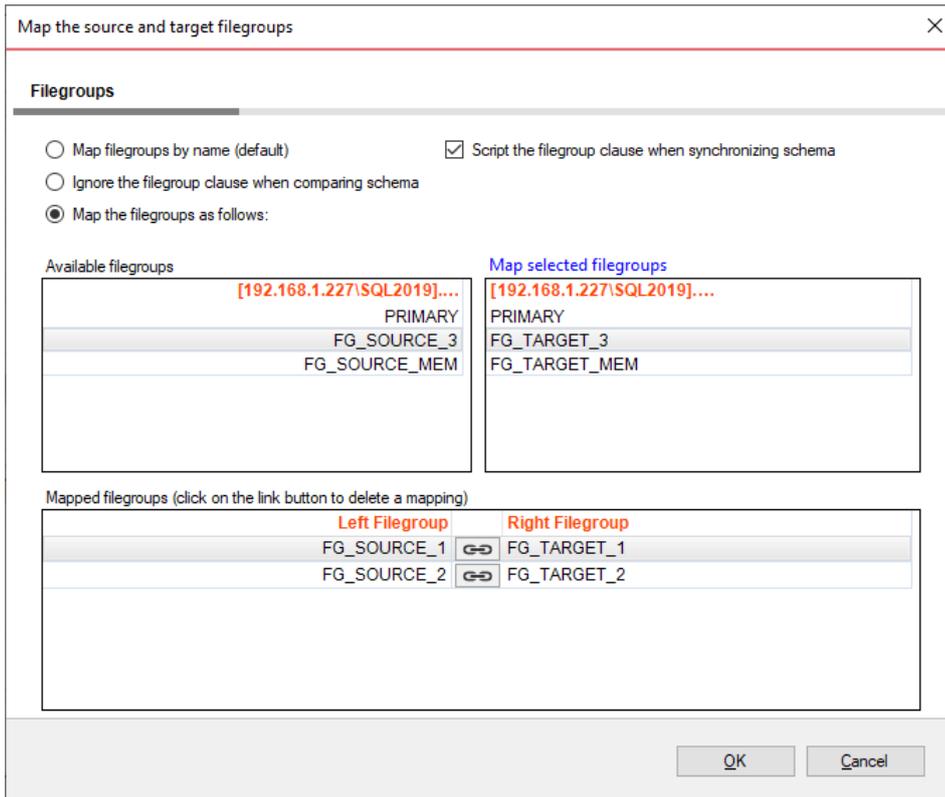
While this could work in most of the cases, there are scenarios in which you might want to keep the filegroup unchanged when transferring objects between the DB_1 and the DB_2 databases. You can achieve this by creating a mapping for the filegroups FG_1 and FG_2. When a mapping exist, the comparison and scripting of the Employees table changes as follows:

- FG_1 and FG_2 are considered equivalent, which means that no schema difference is triggered by the filegroup clause of the Employees table.
- If other schema differences require the Employees table to be synchronized, it will be created on the "mapped" filegroup, not the original one.

Working with filegroup mappings

IDERA SQL Schema Compare provides a few mapping rules that you can choose:

- Mapping filegroups by name: it is the default mapping option.
- Ignoring filegroups when comparing schema: indicates that the filegroup clause is ignored and does not trigger a schema difference between objects.
- Custom mappings: allows you to create custom mappings between database filegroups.



You can create a filegroup mapping as follows:

- Select a filegroup from the list of filegroups on the left database.
- Select a filegroup from the list of filegroups on the right database.
- Click **Map Selected Filegroup**. A mapping row will appear in the **Mapped Filegroup** grid.

To delete a mapping, select the mapping row in the **Mapped Filegroup** grid, and then click **Link**. The members of the mapped filegroup will appear on the **Available Filegroup** grids.

i If one of the databases contains filestream filegroups, a second tab appears in the mapping form that allows you to map filestream filegroups in the same way you map the regular filegroups.

Scripting filegroups

The mapping form provides an additional option that affects the scripting of the filegroup clause for various database objects. When unchecked, database objects that support the filegroup clause are created in the DEFAULT filegroup or, in some cases depending on the object, the filegroup option is ignored.

i The filegroup script option does not affect the database partition schemes.

Objects affected by filegroup mappings

Database objects affected by the filegroup mapping option include:

- **Tables.** Affects the filegroup, text filegroup, filestream filegroup and the filegroup option of the change data capture
- **Primary Keys.** Affects the filegroup and the filestream filegroup clause
- **Unique Constraints.** Affects the filegroup and the filestream filegroup clause
- **Relational Indexes.** Affects the filegroup and the filestream filegroup clause
- **Spatial Indexes.** Affects the filegroup clause
- **Full-Text Indexes.** Affects the filegroup clause
- **Service Broker Queues.** Affects the filegroup clause
- **Partition Schemes**

Best practices and restrictions

Mapping the filegroups, when not done properly, could produce unexpected results. These simple rules and restrictions can help you achieve the intended results:

- Filegroup mappings are one-to-one.
- Even though it is not required, we recommend that you map all filegroups. The filegroups that are not mapped explicitly, will be mapped by name (the default rule).
- Filestream filegroups are mapped separately from the regular filegroups. You will notice that if one of the databases that you are comparing contains filestream filegroups, the mapping form includes an additional tab for filestream filegroups. The mapping rules are the same as the ones for regular filegroups.
- The filegroup scripting option, when unchecked, effects database objects differently. Some objects, such as tables, indexes, are scripted with the DEFAULT filegroup clause. Other objects, such as change data capture, ignore the filegroup option entirely. The database filegroups and partition schemes are not effected by the scripting option.

SQL Schema Compare Synchronizing Databases

Propagating database schema changes from one environment to another (often from development to production) safely and efficiently is the most critical functionality that IDERA SQL Schema Compare provides to software developers and database administrators.

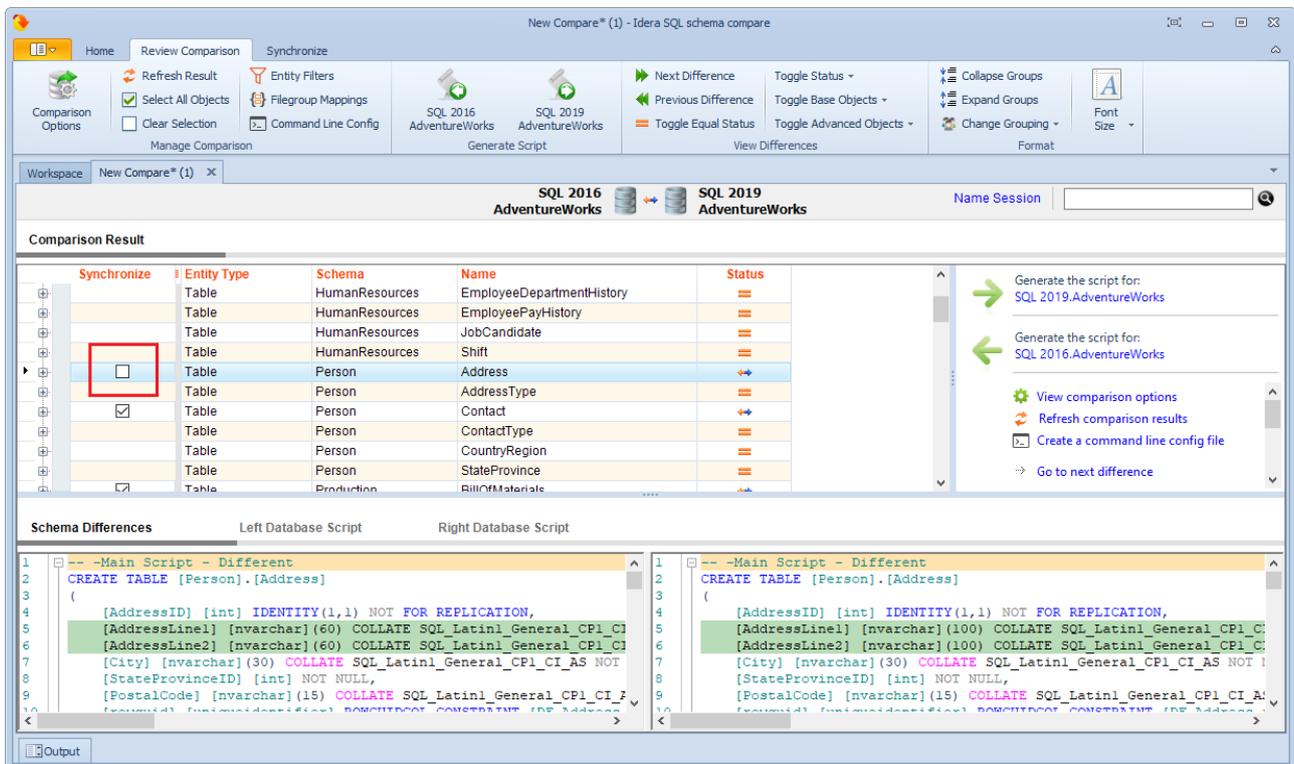
After the comparison of the schemas of two databases has been completed you can select the objects you want to include in the synchronization and generate the schema synchronization script. The generated script will be displayed on a new tab where you will be able to review and execute it.

SQL Schema Compare Selecting Database Objects

By default, all the objects that were found to be different during the comparison of the two databases are marked for inclusion in the synchronization script. However, you can choose to exclude or include certain objects from the comparison by checking or clearing the **Synchronize** checkbox next to each object. Selecting or clearing a parent object also affects the dependent objects. For example, if the checkbox for a table is cleared, then all of its primary keys, foreign keys, indexes etc. are also cleared.

Cleared but included

You may be surprised to see included in the synchronization script an object that you explicitly excluded. Why are those objects included? To answer this you need to look under the comparison / synchronization options and see if the option to **Include Dependent Objects** is **ON**. When that option is **ON**, an included object will cause all the objects on which it depends to also be included in the synchronization script. For example, if you have a table T1 that contains foreign keys coming from tables T2 and T3 then, even though you may have cleared tables T2 and T3 if T1 is checked then both T2 and T3 will be included. When the **Include Dependent Objects** option is **OFF**, then your object selection is respected, however, in this case you should be aware that the resulting synchronization script might not execute successfully.



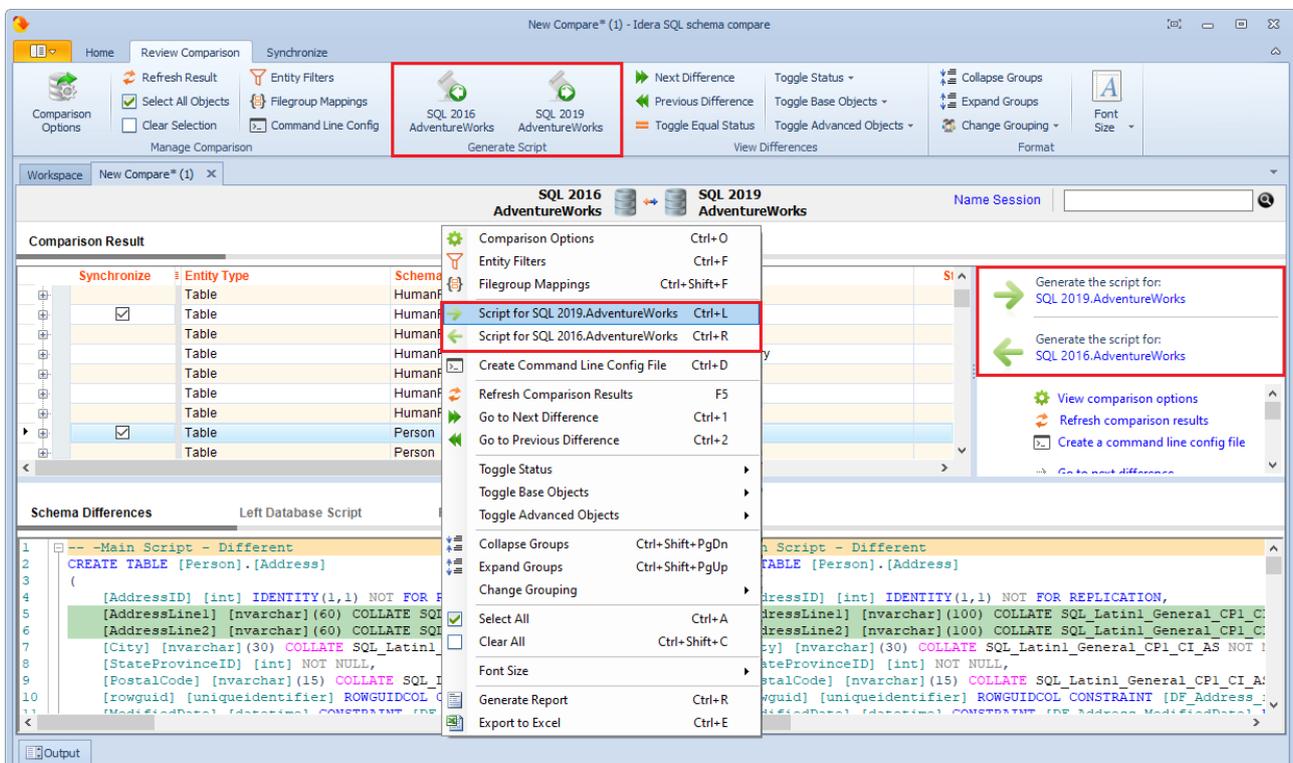
SQL Schema Compare Generating Script

Once the schema comparison operation is finished you are now ready to start the synchronization operation.

By default, all objects that are found to have differences are automatically marked to be included in the synchronization. You can however easily change which objects should or should not be included in the script by checking or clearing the Synchronization checkbox (see [SQL Schema Compare Selecting Objects](#)).

Once you have made your selections you can generate the synchronization script to either change the database on the right of the comparison to make it the same as that on the left or the other way around. There are three ways to start the script generation:

- **Ribbon.** In the Review Compare tab of the ribbon there are two big buttons in the middle under the grouping **Generate Script**. Each button is labeled with the name of the server plus the name of the database for which the script will be generated.
- **Action Links** on the right panel. (listed under the label **Generate the script for**) Clicking on the [server] . [database] link generates the synchronization script that will make that database the same as the other one.
- **Context menu.** When you right click on the comparison results grid a context menu will pop up. The context menu allows you to access the comparison options and trigger the generation of the synchronization script in addition of other context relevant actions.



SQL Schema Compare Working with the Script

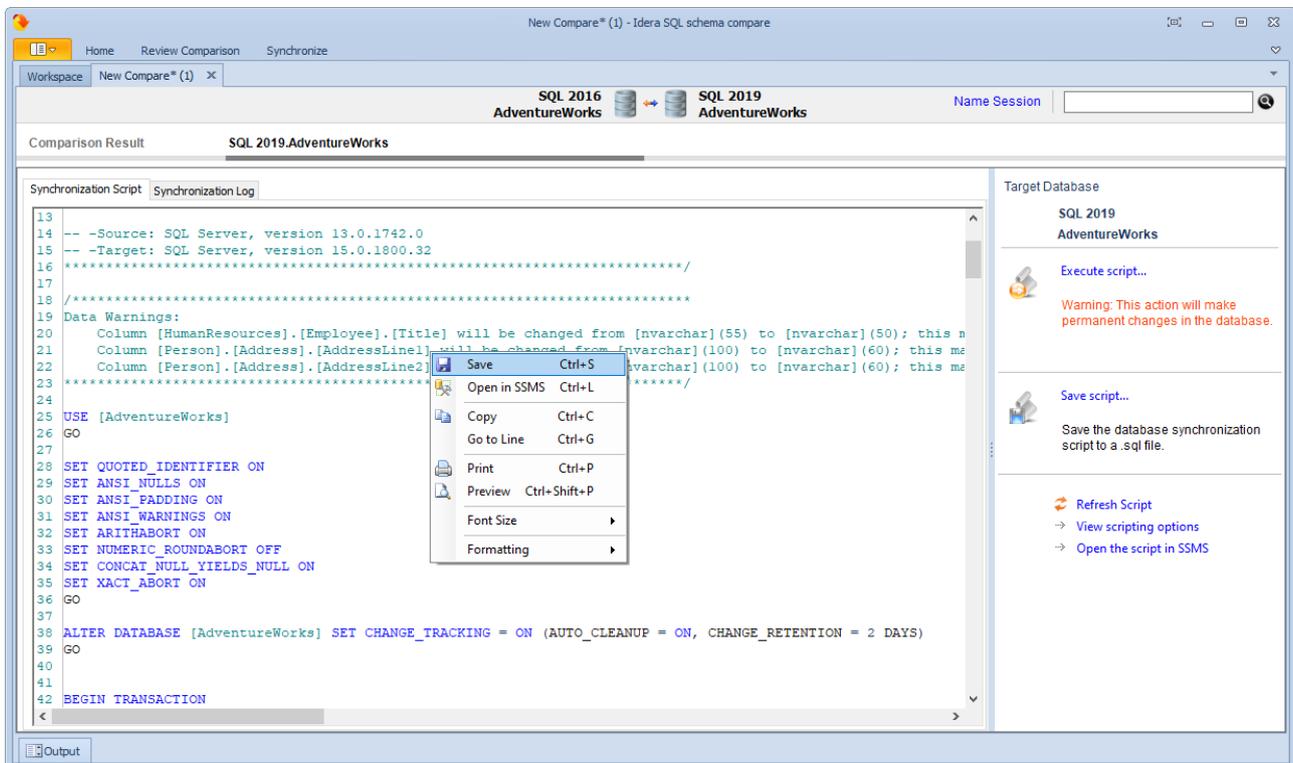
The data synchronization script will be displayed on a new window within the comparison sessions tab. The new window is named with the [server].[database] for which the script was generated and it contains two sub-tabs. The Synchronization Script tab contains the actual script whereas the Synchronization Log tab contains a summary of the actions that will be performed on the target database.

Warnings section

It is important that you pay special attention to the warnings displayed at the top section of the synchronization script window. When the synchronization requires dropping objects such as columns or tables that may contain data or in situations when the synchronization has to be performed through an intermediary table IDERA SQL Schema Compare will display a warning indicating the potential risk that such action carries.

Both the ribbon and context menu allow you to open the script in SSMS, save the script, copy, print, or jump to a certain line in the script.

Note that if you go back to the comparison results tab and redo the comparison the corresponding synchronization script may be out of sync as the script is not automatically re-generated. To refresh the script, click **generate the script...** on the Comparison Results tab.



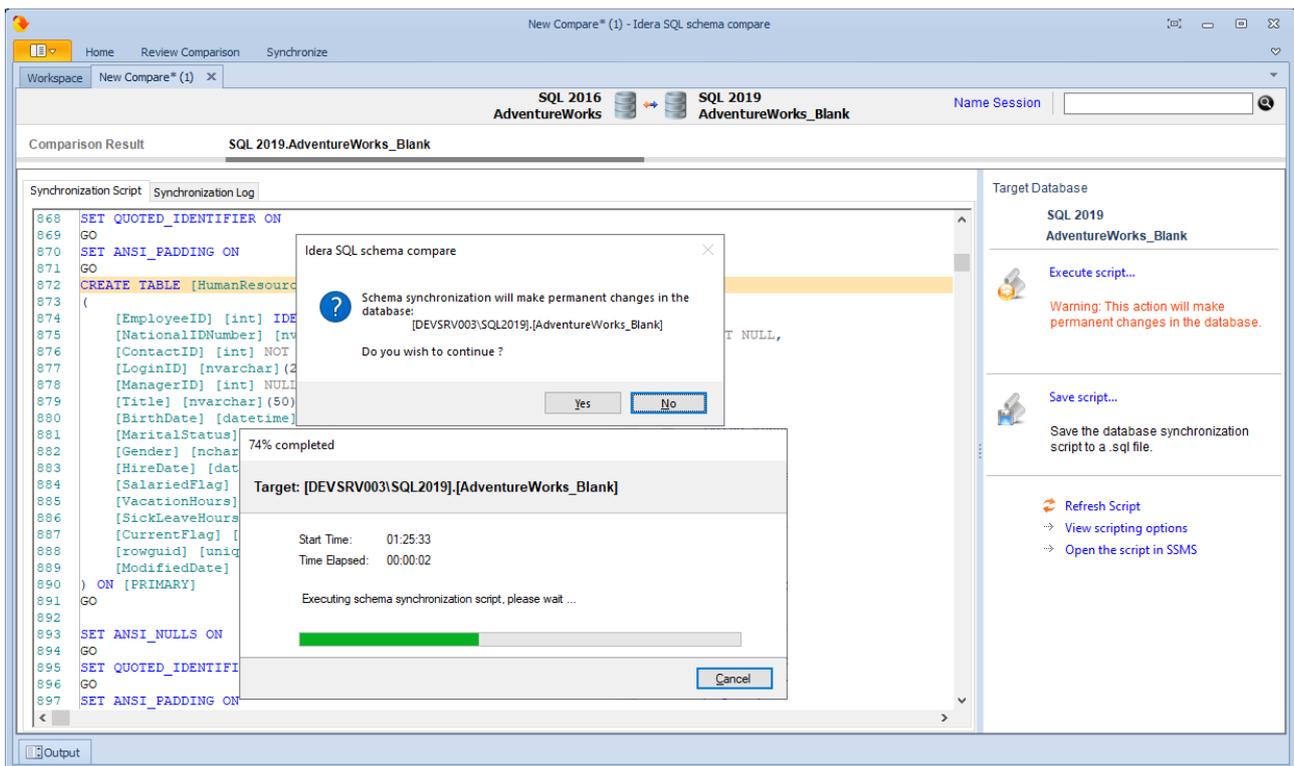
SQL Schema Compare Executing the Script

After reviewing the synchronization script and making sure that the script will do what you intend it to do, you can execute it.

⚠ This action will make permanent changes to the content of the target database. It is common sense and we strongly recommend that before executing the script you:

- Make a full backup of the target database. Keep in mind that if columns or tables that contain data will be dropped by the synchronization script the only way to reverse the changes will be to restore the data from a full backup of the database.
- Save the script you are about to execute so that you know exactly what was done to the database and when.

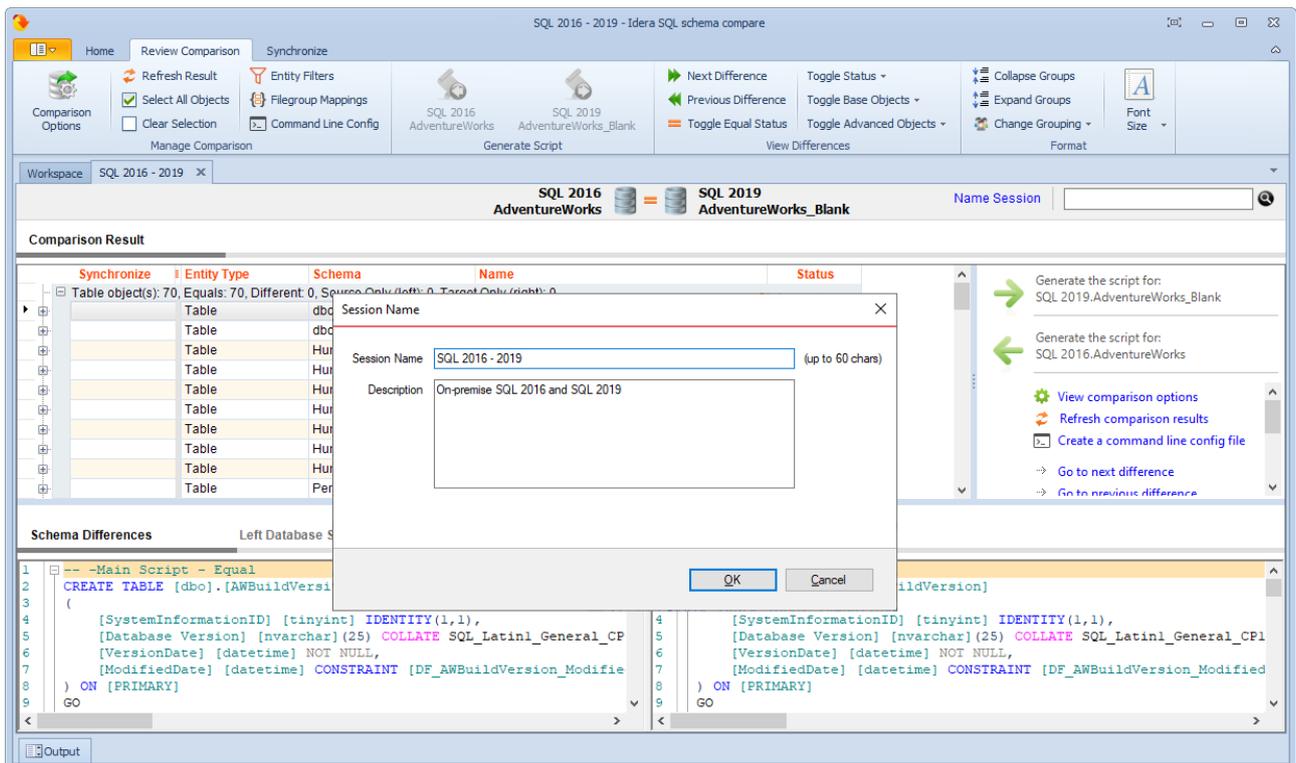
You can execute the script directly from IDERA SQL Schema Compare SSMS and execute it from there. Please pay attention to the Action section on the header of the synchronization script to make sure that the script will be executed against the correct database.



SQL Schema Compare Comparison Sessions

Depending on the scenario, the first time you compare two particular databases you might have to spend some time to configure the comparison and synchronization just the way you want it. IDERA SQL Schema Compare automatically saves all your configuration choices on a "comparison session" so that next time you might need to compare and synchronize those same databases you can do that with one click without having to go through that configuration effort again.

You can also name the comparison sessions to make it easier to identify them. Up to 25 sessions are saved and displayed on the Workspace tab.



SQL Schema Compare Tools and Utilities

IDERA SQL Schema Compare provides the following tool to help you work with database snapshots. Click on the following link to view details about the tool:

- [IDERA Snapshot Utility](#)

SQL Schema Compare IDERA Snapshot Utility

IDERA Snapshot Utility is a tool that allows you to create database snapshots via the command line. A snapshot is a file that contains all the schema information for a database. The schema information is stored in a proprietary format and can only be read by the IDERA SQL Schema Compare user interface or command line.

i A snapshot file stores only the structure of the database, with no user-data of any kind. It can be used to recreate the tables and other objects in a database, but the tables will be empty.

The Snapshot Utility accepts parameters in the format `/name:value`. There are no spaces between the name and the value and the value is separated from the name with a colon `<:>`. If the value itself contains spaces, it should be surrounded in double quotes.

Idera Snapshot Utility supports the following arguments:

- `/s:<server>` Specifies the SQL Server instance.
- `/d:<database>` Specifies the database name.
- `/t:` Indicates that the connection to the database should be a trusted connection. If not specified, the connection by default is trusted.
- `/u:<username>` Specifies the username when the database uses the SQL Server authentication.
- `/p:<password>` Specifies the password when the database uses the SQL Server authentication.
- `/sp:<filename>` Saves the snapshot to the specified filename. If the filename is not set, the snapshot file will be named as `database.snpx`.
- `/cs:<connection_string>` Specifies the connection string to the database. Use this option when you need to set additional settings that are not supported directly such as the connection timeout.
- `/f:<setting_file>` Creates the snapshot using the settings in the specified file. When the `/f` parameter is specified, all other parameters are ignored. In addition of being easier to manage, using a setting file allows you also to create snapshots for multiple databases. In this case use one line per database.

Samples:

The following command creates a snapshot for the database AdventureWorks on the local SQL Server using a trusted connection:

```
IderaSnapshot /s:(local) /d:AdventureWorks /sp:"c:\AdventureWorks.snpx"
```

The following command creates the snapshot by specifying a connection string:

```
IderaSnapshot /cs:"Persist Security Info=False;User ID=<username>;Password=<password>;Initial Catalog=AdventureWorks;Data Source=(local)" /sp:"c:\AdventureWorks.snpx"
```

The following command uses the setting file snapshot.txt to create the snapshot for two databases:

```
IderaSnapshot /f:"c:\snapshot.txt"
```

The setting file snapshot.txt contains the following lines:

```
/s:(local) /d:AdventureWorks_DEV /sp:"c:\AdventureWorks_DEV.snpx"
```

```
/s:(local) /d:AdventureWorks_PROD /sp:"c:\AdventureWorks_PROD.snpx"
```

IDERA SQL Schema Compare Command Line

Idera SQL Schema Compare command line is a set of tools that allow you to compare SQL Server databases via the command line.

With the command line you can:

- Perform automated schema comparison and synchronization;
- Compare live SQL Server databases or database snapshots;
- Compare and synchronize databases as part of your setup and deployment solution;
- Compare databases and execute the synchronization script in batch files;
- Schedule the database comparison.

The command line includes:

- A wizard for generating the command line xml config file.
- The command line utility, that uses the xml config file to compare and synchronize SQL Server databases.

We included command line samples for the most common scenarios, in order to help you get started. The sample xml config files are located in the toolset installation folder, usually under **\Program Files (x86)\Idera\SQL comparison toolset v<n>\Command Line Samples\Schema Compare**, where **v<n>** is the toolset version number.

Using SQL Schema Compare Command Line

IDERA SQL Schema compare command line expects and reads the comparison settings from an xml config file. The config file contains the SQL Server instance, its credentials, the SQL Server database, the comparison options, excluded objects, filters, the output files and many more. You can generate the config file using the command line wizard.

Running the command line, using the config file config.xml, is as simple as follows:

```
IderaSchemaCmd <path>\config.xml
```

You can also use the command line to validate a config file, without comparing the SQL Server databases. This can be useful if you create the config file manually.

Validation is performed via the /v parameter:

```
IderaSchemaCmd <path>\config.xml /v
```

To view the command line usage, use the help parameter: /? /h or /help

```
IderaSchemaCmd /?
```

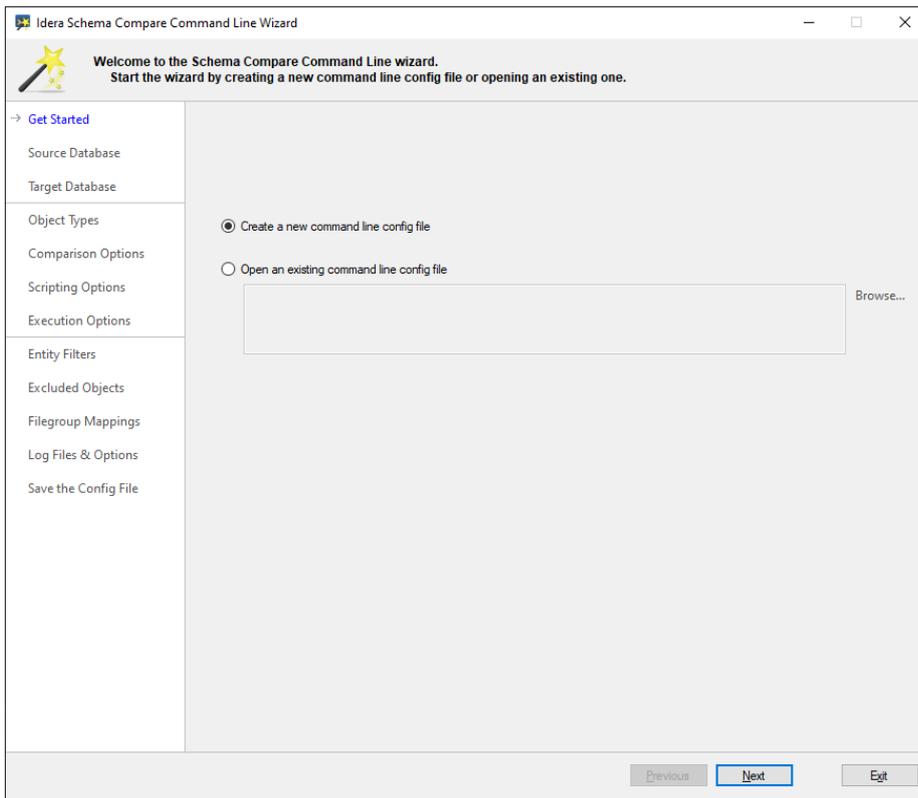
 Due to the complexity of the config file, we strongly recommend that you use the command line wizard to generate the xml config files.

SQL Schema Compare Command Line Wizard

The command line wizard, included with Idera SQL Comparison Toolset, allows you to create a new xml config file or modify an existing one. The wizard hides the complexity and generates a consistent, syntactically correct and well-optimized config file, especially for complicated comparison scenarios.

To launch the command line wizard, go to the **Start\Idera SQL comparison toolset** and click **Schema Compare Command Line Wizard**.

You can start by creating a new config file or opening an existing one.



i When you open an existing config file, the wizard will perform a thorough validation of it. If the xml file is correct, you will see a green check mark to the left of the filename. If the xml file is incorrect, depending on the severity of the problems, the wizard could generate warnings or reject the file entirely.

SQL Schema Compare Command Line Source and Target Database

The source and target database steps allow you to specify the SQL Server databases that will be compared. For more details on the database connection settings, see [SQL Schema Compare Add Databases](#).

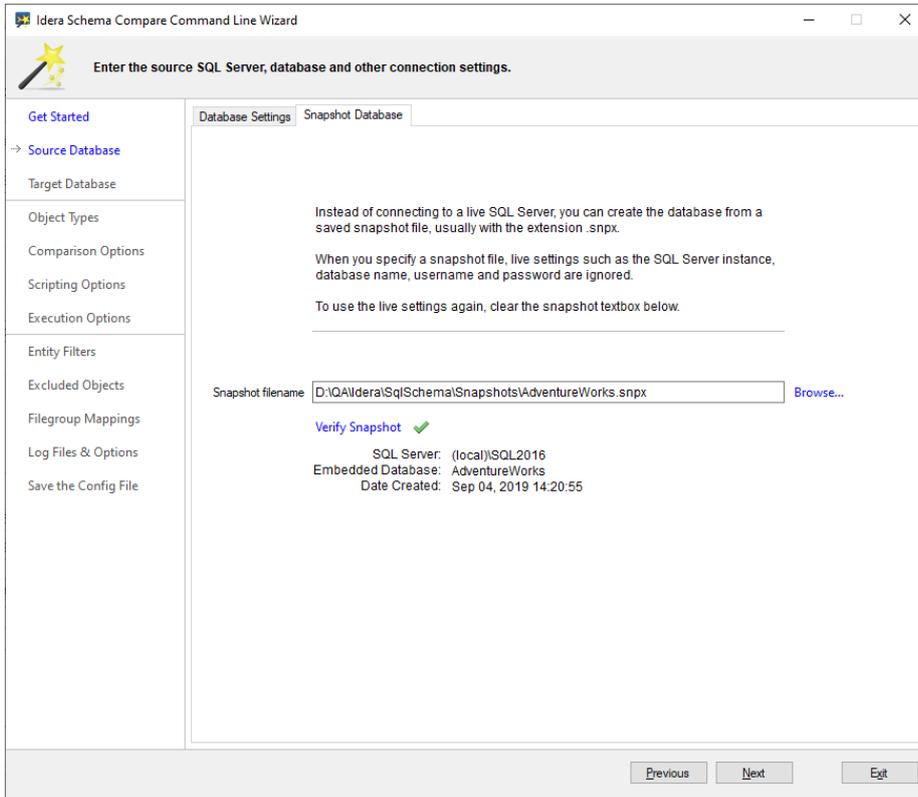
i Even though we refer to the SQL Server database as the "source" or "target", the command line can switch them, so the source can become the target and vice versa. More details on setting the target database are included in the Log Files and Options step.

Below is the xml fragment that contains the SQL Server databases:

```
<LeftDatabase>
  <SqlServer>DEVSRV002\SQL2016</SqlServer>
  <DatabaseName>AdventureWorks</DatabaseName>
  <TrustedConnection>>true</TrustedConnection>
</LeftDatabase>
<RightDatabase>
  <SqlServer>DEVSRV003\SQL2019</SqlServer>
  <DatabaseName>AdventureWorks</DatabaseName>
  <TrustedConnection>>true</TrustedConnection>
</RightDatabase>
```


SQL Schema Compare Command Line Database from a Snapshot

Instead of a live SQL Server database, you can specify a snapshot file. The database snapshot is a proprietary file, created by the schema compare, that contains all the information necessary to recreate the structure of a database.



i A snapshot file can be specified for the source or the target database. However, the database that you intend to synchronize, has to be a live database. The synchronization script cannot be executed in a database created from a snapshot file.

Below is the xml fragment that creates the database from a snapshot file:

```
<LeftDatabase>
  <LoadFromSnapshotFile>D:
  \QA\Idera\SqlSchema\Snapshots\AdventureWorks.snpx</
  LoadFromSnapshotFile>
</LeftDatabase>
<RightDatabase>
  <SqlServer>DEVSRV003\SQL2019</SqlServer>
  <DatabaseName>AdventureWorks</DatabaseName>
  <TrustedConnection>>true</TrustedConnection>
</RightDatabase>
```


SQL Schema Compare Command Line Creating a Snapshot

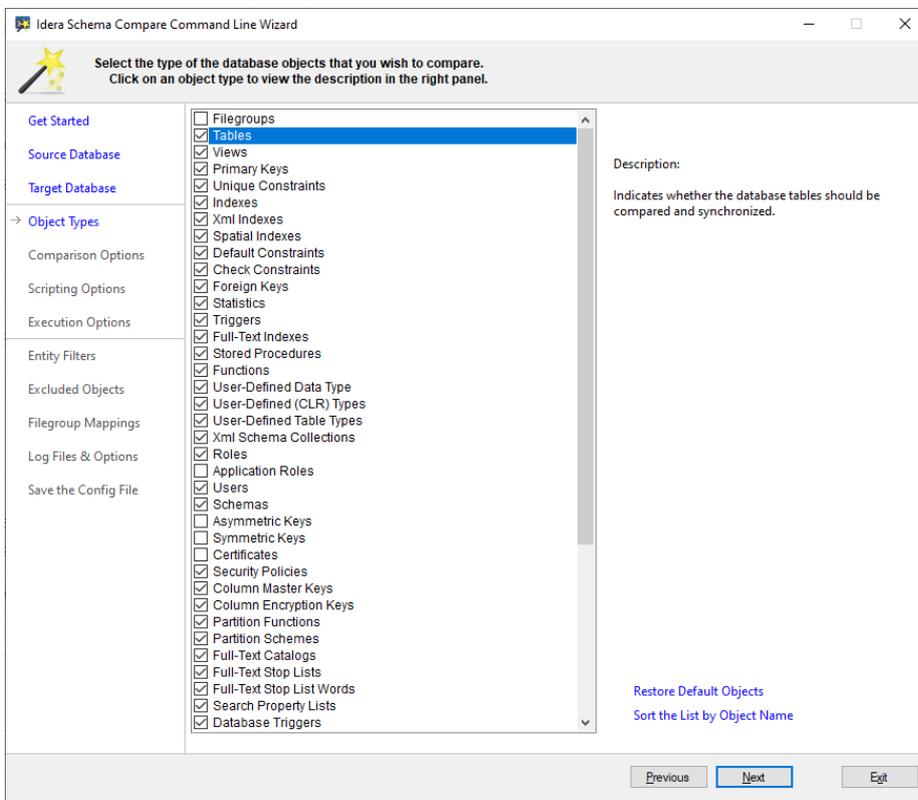
You can choose to save the database structure to a snapshot file before comparing and synchronizing the database schema. To do this, check **Take a snapshot of the database before synchronization**, and then specify a snapshot file.

Below is the xml fragment generated when you choose to save the database to a snapshot file.

```
<RightDatabase>
  <SqlServer>DEVSRV003\SQL2019</SqlServer>
  <DatabaseName>AdventureWorks</DatabaseName>
  <TrustedConnection>true</TrustedConnection>
  <SaveToSnapshotFile>D:
  \QA\Idera\SqlSchema\Snapshots\AdventureWorks.snpx</SaveToSnapshotFile>
</RightDatabase>
```

SQL Schema Compare Command Line Excluding Objects by Type

The Object Types step allows you to include or exclude SQL Server objects by type.



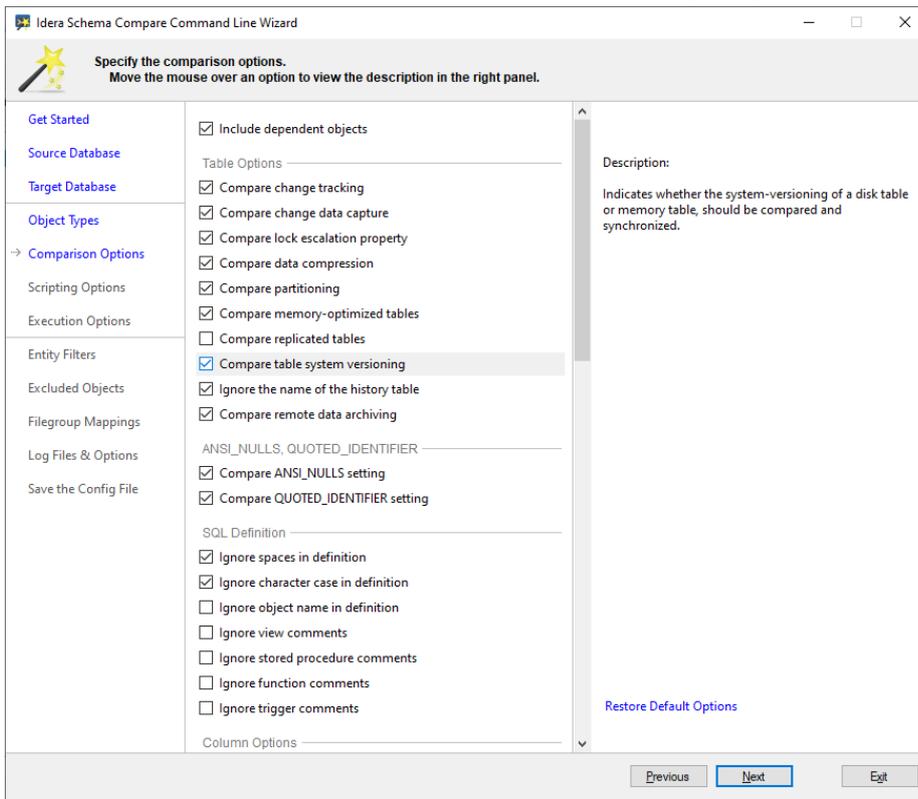
Below is an example of the xml fragment generated when some object types are excluded from the comparison:

```
<ObjectTypeSelection>
  <ObjectType Type="UserDefinedDataType" Include="false" />
  <ObjectType Type="ClrDataType" Include="false" />
  <ObjectType Type="UserDefinedTableType" Include="false" />
  <ObjectType Type="XmlSchemaCollection" Include="false" />
  <ObjectType Type="DatabaseRole" Include="false" />
  <ObjectType Type="User" Include="false" />
  <ObjectType Type="Schema" Include="false" />
  <ObjectType Type="PartitionFunction" Include="false" />
  <ObjectType Type="PartitionScheme" Include="false" />
  <ObjectType Type="SecurityPolicy" Include="false" />
  <ObjectType Type="ColumnMasterKey" Include="false" />
  <ObjectType Type="ColumnEncryptionKey" Include="false" />
</ObjectTypeSelection>
```

SQL Schema Compare Command Line Comparison Options

The command line wizard organizes the comparison options into three groups:

- **Comparison Options.** contains the options that effect the comparison of the SQL Server databases.
- **Scripting Options.** contains the options that are used to generate the synchronization script.
- **Execution Options.** contains the options that effect the execution of the synchronization script.



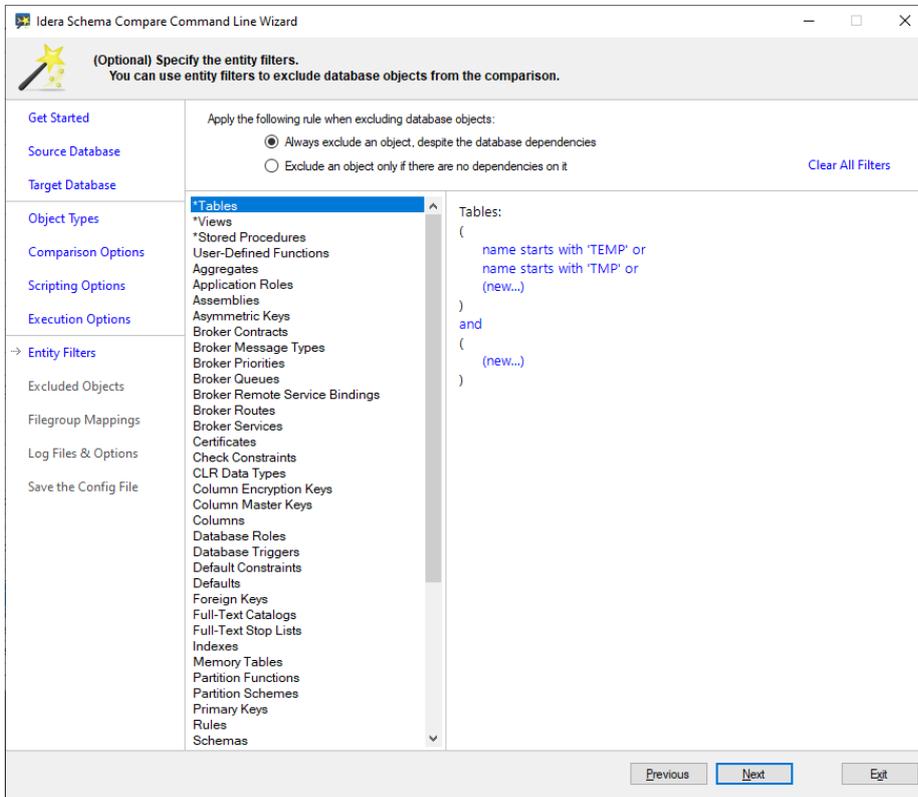
The following is an example of the xml fragment generated when some comparison options have been changed:

```
<SelectedComparisonOptions>
  <ComparisonOption>IgnoreNameInDefinition</ComparisonOption>
  <ComparisonOption>IgnoreViewComments</ComparisonOption>
  <ComparisonOption>IgnoreProcedureComments</ComparisonOption>
  <ComparisonOption>IgnoreFunctionComments</ComparisonOption>
  <ComparisonOption>ScriptSequenceStartValue</ComparisonOption>
  <ComparisonOption>ScriptSequenceMinValue</ComparisonOption>
  <ComparisonOption>ScriptSequenceMaxValue</ComparisonOption>
  <ComparisonOption>LogScriptExecution</ComparisonOption>
</SelectedComparisonOptions>
```

```
<ExcludedComparisonOptions>  
  <ComparisonOption>CompareDataCompression</ComparisonOption>  
  <ComparisonOption>CompareTableChangeTracking</ComparisonOption>  
  <ComparisonOption>CompareTableChangeDataCapture</ComparisonOption>  
  <ComparisonOption>CompareTableLockEscalation</ComparisonOption>  
  <ComparisonOption>ScriptAnsiNullsSetting</ComparisonOption>  
  <ComparisonOption>ScriptQuotedIdentifierSetting</ComparisonOption>  
  <ComparisonOption>ScriptAnsiPaddingSetting</ComparisonOption>  
</ExcludedComparisonOptions>
```

SQL Schema Compare Command Line Excluding Objects Using Entity Filters

The Entity Filters step allows you to exclude database objects using filters. For more information about filters, see [SQL Schema Compare Entity Filters](#).



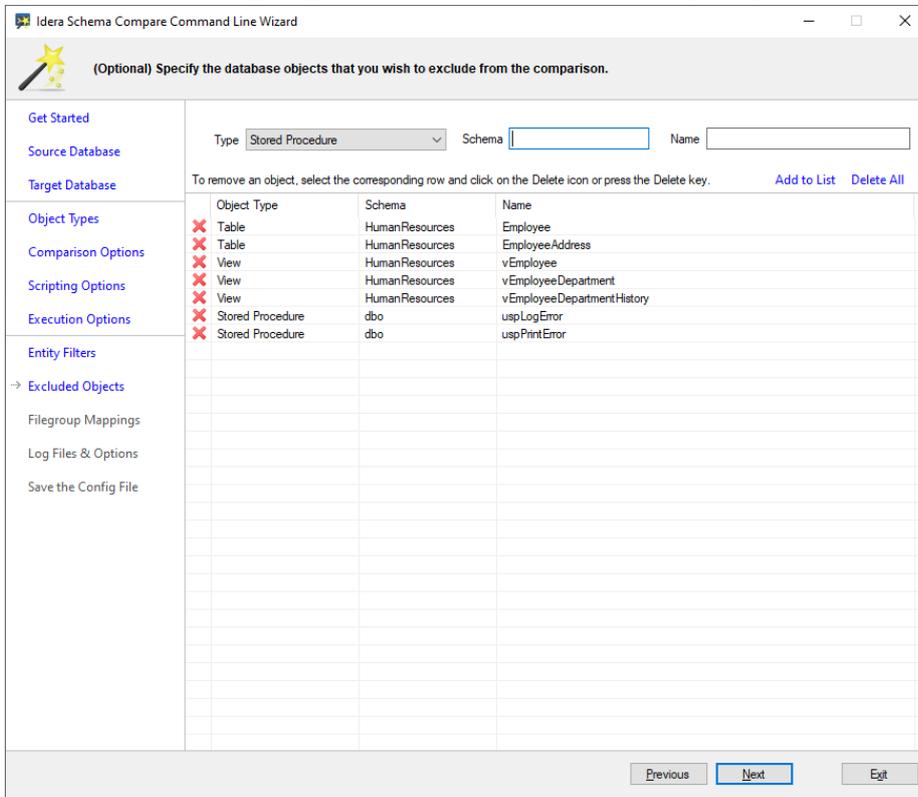
The following xml fragment contains a set of filters for the database tables, views and procedures:

```
<EntityFilters>
  <Filter EntityType="Table">
    <Group ConditionOperator="Or">
      <Condition>
        <ConditionType>StartingWith</ConditionType>
        <Expression>TEMP</Expression>
      </Condition>
      <Condition>
        <ConditionType>StartingWith</ConditionType>
        <Expression>TMP</Expression>
      </Condition>
    </Group>
  </Filter>
  <Filter EntityType="View">
    <Group ConditionOperator="Or">
      <Condition>
```

```
    <ConditionType>StartingWith</ConditionType>
    <Expression>V_TEMP</Expression>
  </Condition>
  <Condition>
    <ConditionType>StartingWith</ConditionType>
    <Expression>V_TEMP</Expression>
  </Condition>
</Group>
</Filter>
<Filter EntityType="StoredProcedure">
  <Condition>
    <ConditionType>Containing</ConditionType>
    <Expression>_DEV_</Expression>
  </Condition>
</Filter>
</EntityFilters>
```

SQL Schema Compare Command Line Excluding Objects by Name

The Excluded Objects step allows you to exclude various database objects by name.



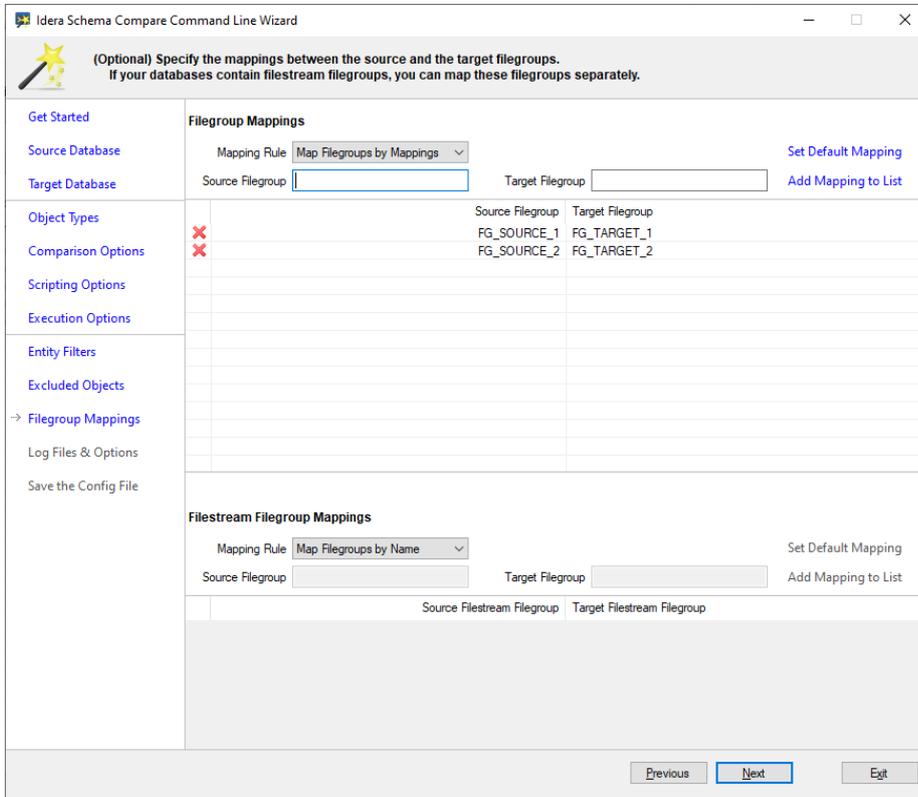
The following fragment is the xml generated when a few objects have been excluded from the comparison by name:

```
<ObjectSelection>
  <Object Type="Table" Schema="HumanResources" Name="Employee"
Include="false" />
  <Object Type="Table" Schema="HumanResources" Name="EmployeeAddress"
Include="false" />
  <Object Type="Table" Schema="Sales" Name="Customer"
Include="false" />
  <Object Type="Table" Schema="Sales" Name="CustomerAddress"
Include="false" />
  <Object Type="Table" Schema="Sales" Name="SalesOrderDetail"
Include="false" />
  <Object Type="View" Schema="HumanResources" Name="vEmployee"
Include="false" />
  <Object Type="View" Schema="HumanResources"
Name="vEmployeeDepartment" Include="false" />
  <Object Type="StoredProcedure" Schema="dbo" Name="uspLogError"
Include="false" />
```

```
<Object Type="StoredProcedure" Schema="dbo" Name="uspPrintError"  
Include="false" />  
</ObjectSelection>
```

SQL Schema Compare Command Line Filegroup Mappings

Filegroup mappings allow you to compare databases that have different filegroup structure. For more details about filegroup mappings, see [SQL Schema Compare Filegroup Mappings](#).



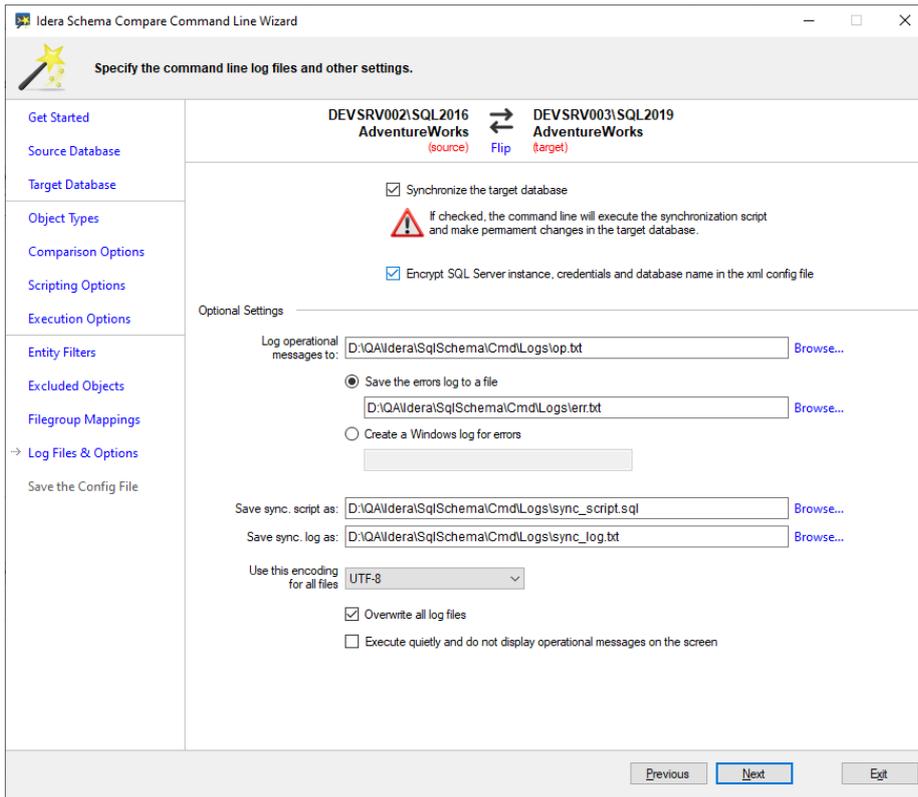
Below is the xml fragment generated for this filegroup mapping scenario:

```
<FileGroupSettings>
  <ComparisonType>ByMappings</ComparisonType>
  <Mappings>FG_SOURCE_1:FG_TARGET_1;FG_SOURCE_2:FG_TARGET_2</Mappings>
</FileGroupSettings>
```

SQL Schema Compare Command Line Log Files and Options

The Log Files and Options contains the command line log files and other settings:

- The top section of the page contains the source and the target schema. You can flip them by clicking on the Flip link. Target database is the database on which the synchronization will be executed.
- **Synchronize the target database.** Indicates whether the synchronization script should be executed in the target database. If unchecked, the script is generated, optionally saved to a file, but not executed.
- **Encrypt SQL Server instance, credentials and database name.** Encrypts these settings in the xml config file.
- **Log operational messages to.** Saves the operational logs to the specified file.
- **Save the error log to a file.** Saves the error to the specified file.
- **Create a Windows error log.** Creates an Windows Event Log for the comparison errors instead of a log file.
- **Save sync script to.** The file where the synchronization script should be saved.
- **Save sync log to.** The file where the synchronization log should be saved.
- **Use this encoding for all files.** The encoding that should be used for all output files.
- **Override all log files.** Whether the log files should be overridden or appended to.
- **Execute quietly and hide operational messages.** If checked, does not show any operational messages on the console.



The following xml fragment contains the command line settings:

```
<CommandLineSettings>
  <LogFile>D:\QA\Idera\SqlSchema\Cmd\Logs\op.txt</LogFile>
  <ErrorLogName>D:\QA\Idera\SqlSchema\Cmd\Logs\err.txt</ErrorLogName>
  <SchemaScriptFile>D:\QA\Idera\SqlSchema\Cmd\Logs\sync_script.sql</
SchemaScriptFile>
  <SchemaLogFile>D:\QA\Idera\SqlSchema\Cmd\Logs\sync_log.txt</
SchemaLogFile>
  <FileEncoding>UTF8</FileEncoding>
  <Overwrite>true</Overwrite>
  <TargetDatabase>RightDatabase</TargetDatabase>
  <!--**** the synchronization script will be executed against the
target database ****-->
  <Synchronize>true</Synchronize>
  <EncryptConnectionSettings>>false</EncryptConnectionSettings>
</CommandLineSettings>
```

SQL Schema Compare Command Line Return Codes

The command line returns 0 when it finishes successfully. A non-zero code indicates an error.

IDERA SQL Data Compare

Welcome to Idera SQL data compare, the tool of choice for thousands of SQL Server DBAs and developers.

Data compare is a tool that allows you to compare and synchronize data between SQL Server databases. It supports on-premise SQL Server databases from 2000 to 2019, and the Azure SQL Database.

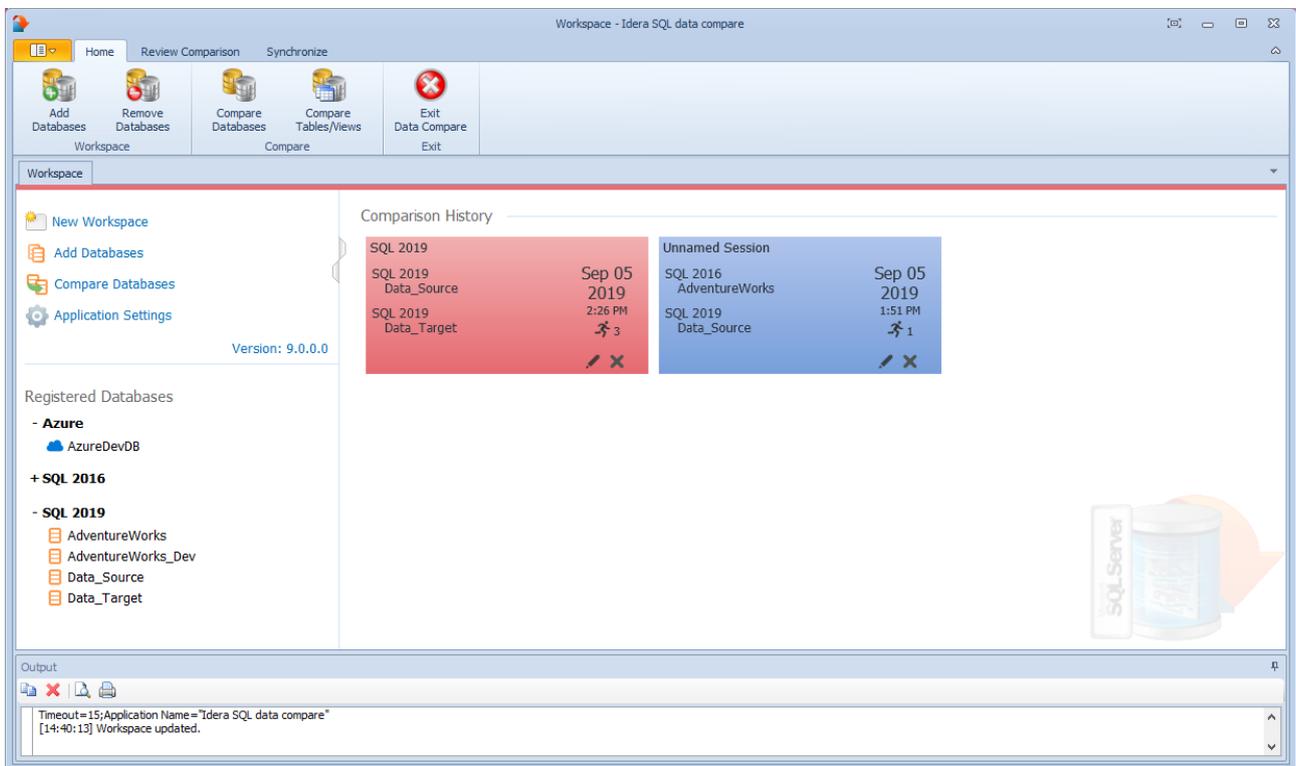
We are indebted to you, users of our tools, for all the comments and suggestions you have given us over time – we truly appreciate your help and attribute much of our success to you! Please continue to help us in our quest to make Idera SQL data compare even better by sending us unfiltered feedback on the current version and suggestions for the future versions at support@idera.com.

Everything in this document, unless it is explicitly stated otherwise, applies to all editions of Idera SQL data compare.

SQL Data Compare Application window

IDERA SQL Data Compare divides the main application window into the following areas:

1. **The ribbon.** The context sensitive ribbon provides all the command buttons and icons you need to do your job quickly. The ribbon is organized in three distinct tabs each of which corresponding to the main steps of the database comparison and synchronization process and "housing" the relevant command buttons.
2. **Main panel.** The main panel contains a permanent tab called the [Workspace](#) and a tab for each active comparison session.
3. **Output window.** The output window provides a running log of the tasks being performed in IDERA SQL Data Compare.

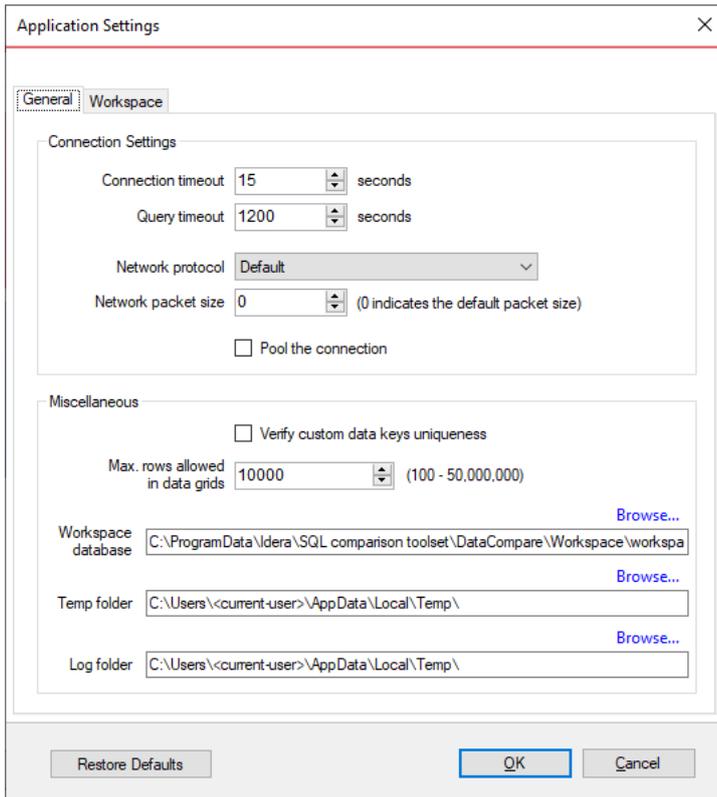


SQL Data Compare Application Settings

Application settings allow you to tweak the behavior of the IDERA SQL Data Compare and can be accessed from the drop down menu off of the top left corner product icon or from the Application Settings link at the top left corner on the Workspace tab.

Under the General tab, you will find:

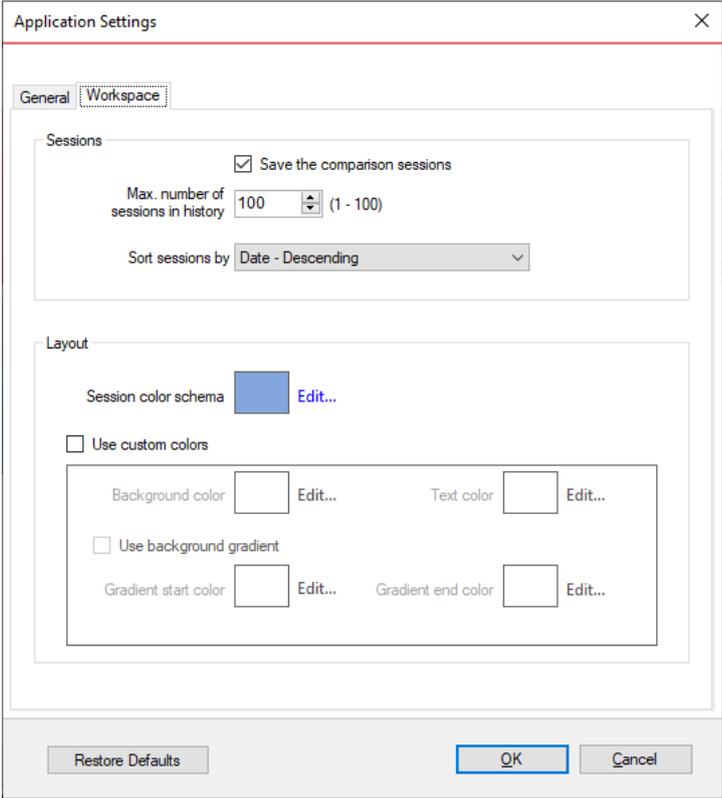
- **Connection Settings** If you are reading this you are already familiar with the database connection settings. Under normal circumstances you do not need to change any of these settings.
- **Verify custom data keys uniqueness** Before comparing the data IDERA SQL Data Compare will pair the rows based on a unique key that is automatically determined. However, you can choose to pair the rows based on some custom key that you define either because the tables in question do not have a predefined unique key or because you wish to use a different key. In this case there is an option to validate the chosen key. By default the validation simply checks for type compatibility. If you wish to ensure that the chosen key is unique you can check this option. Please note however that depending on the size of those tables the validation for uniqueness may take from a few seconds to minutes to complete.
- **Max rows allowed in the data grids** Indicates the maximum number of rows that are displayed in the various data grids.
- **Temp folder** Indicates the temp folder that Data Compare uses for row and script serialization. Note that if you change the temp folder, you need to restart Data Compare for the new setting to take effect. The default folder is **C:\Users\\AppData\Local\Temp**.
- **Log folder** the folder where the log files, such as the error log or the execution log are stored. The default folder is **C:\Users\\AppData\Local\Temp**.



Under the Workspace tab you will find:

- **Sessions** Contains settings related to comparison sessions:
 - **Save sessions** By default it is checked. It instructs the IDERA SQL Data Compare to store comparison sessions for future use. A stored session contains all the necessary information to allow you to repeat the comparison with one click. Stored sessions will appear on the main panel of the Workspace tab.
 - **Max number of sessions in history** By default IDERA SQL Data Compare will store the last 100 comparison sessions. Although there is no noticeable performance difference on application launch related to the number of sessions stored you may choose to set this parameter to a lower number. Allowable values are 1 to 100.
 - **Sort sessions by** Change the order that stored sessions appear on the workspace tab.
- **Layout** Allows you to change the color schema of a session. You can choose from built-in colors or pick custom ones.

To revert all settings back to their default values, click **Restore Defaults**.

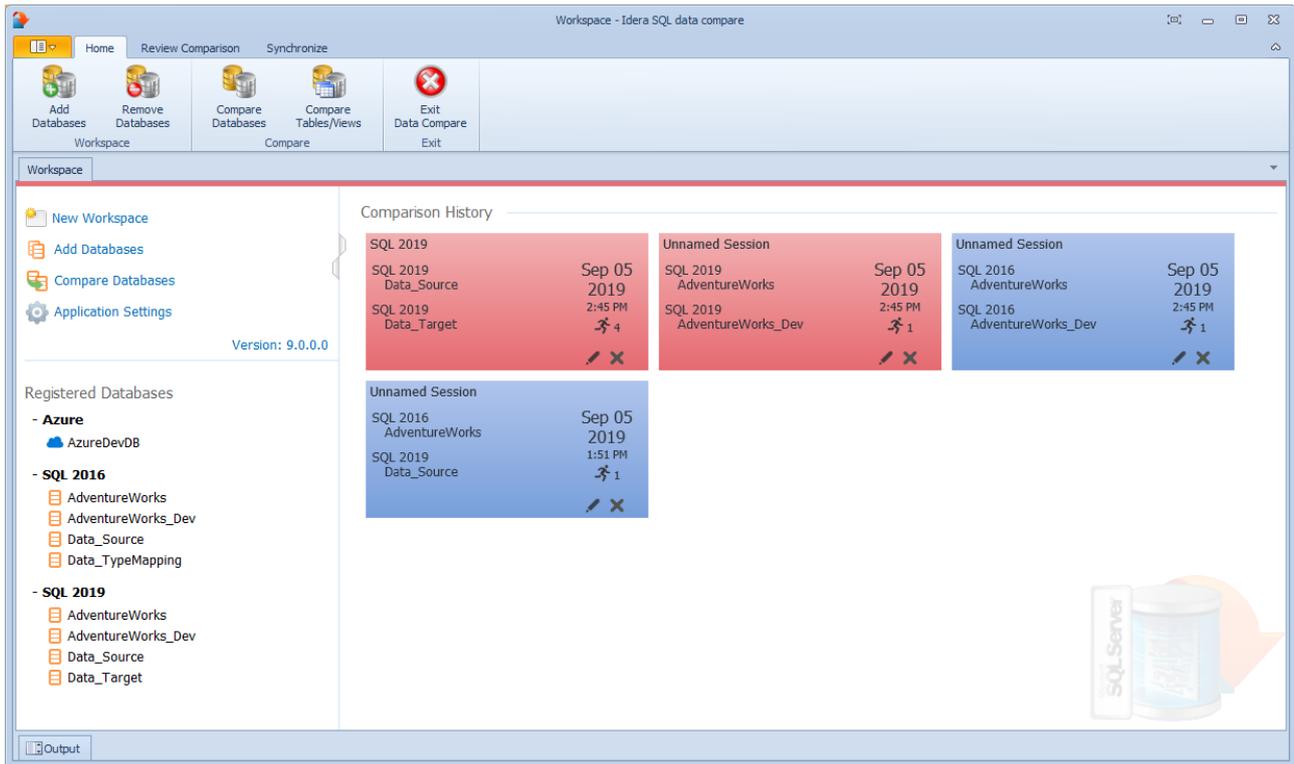


SQL Data Compare Workspace

IDERA SQL Data Compare workspace is the storage and the interface for saving and managing SQL Server databases, credentials, comparison sessions and other UI options.

The Workspace window is divided into the following sections:

- **Workspace actions.** This section provides links that allow you to create a new workspace, add databases to the current workspace, compare database and view application settings. The **New Workspace** action resets the current workspace by erasing the comparison history and the registered databases.
- **Registered Databases.** Shows the list of servers/databases that have been registered. On mouse over two action links appear for a registered server: **Edit** allows you to change the connection settings for that server and/or add and remove databases; and **Remove** un-registers a server (deletes all the server information from the workspace).
- **Comparison History.** Lists up to 100 comparison sessions from the most recent one to the oldest one. To re-open a comparison session, simply click it. The two links at the bottom-right of each session allow you to edit a session or delete it. For each session, data compare displays the following information:
 - The left and the right databases included in the session. These are the databases that are compared when you launch the session.
 - The date and time when the session was created
 - The session execution count, which indicates the number of times to session has been opened. A new sort order, under the application settings, allows you to arrange the sessions in the workspace by execution count, so that the ones that you use the most are always at the top. You can also change the execution count by editing the session.



Workspace Storage

The workspace stores its settings in a database file, which by default is C:\Program Data\Idera\SQL Comparison Toolset\Data Compare\Workspace\workspace.db. If you prefer to use a different file on a different folder, do the following:

- Copy the default database file to a folder of your choosing. You may rename the folder.
- Launch SQL Data Compare, open the Application Settings, and then change the workspace database to the file you created.

i The workspace requires write permissions to its database file. If it cannot write to its database, it displays a read-only message.

SQL Data Compare Add Databases

Before comparing SQL Server databases, you must add them to the workspace. Access **Add Databases** from the ribbon (the Home tab) or from the **Add Databases** action link located in the top-left section of the Workspace.

On the Add Database window, enter the SQL Server instance, a friendly name you wish to use in the UI, and the SQL Server credentials. Click **Refresh** in the SQL Server drop down to discover the SQL Server instances running on your network. If the application is unable to discover SQL Server instances, type in the SQL Server instance in the drop down box.

i If the SQL Server is not listening on the default port 1433, you can specify the port number as: **[ServerName]\[Instance Name],PortNumber**

Once you have chosen the SQL Server and the credentials, click **Read Databases**. Data compare will connect to the specified SQL Server and bring in the databases from that server. You can add one or more databases to the workspace by checking them in the database listbox.

Under the Advanced Settings tab, you can enter various connection settings, such as the timeouts, the connection protocol, packet size and a few more. By default data compare uses the advanced settings specified in the Application Settings.

The Other connection settings edit box allows you to specify additional connection properties as **name=value**, separated by semicolons: **name1=value1;name2=value2;...** To check whether these properties are allowed for

a SQL Server connection, click **Check Settings**. For a full list of the connection properties supported for by a SQL Server database, check the MSDN.

The following property names are already included in the SQL Server connection and should not be specified in the **Other connection settings** area. Their synonyms must be excluded as well:

- Data Source
- Initial Catalog
- Integrated Security
- Persist Security Info
- User ID
- Password
- Pooling
- Connect Timeout
- Network Library
- Packet Size
- Application Name

Add SQL Server databases to the workspace

SQL Server Instance: DEVSRV003\SQL2019 [Clear All](#)

Friendly Name: SQL 2019

Authentication Type: Windows Authentication

Username:

Password: [Show Password](#)

Databases | **Advanced Settings**

Connection Timeout: 15 seconds [Restore Defaults](#)

Query Timeout: 1200 seconds

Protocol: Default

Packet Size: 0 (0 default)

Pooled Connection

[Check Settings](#)

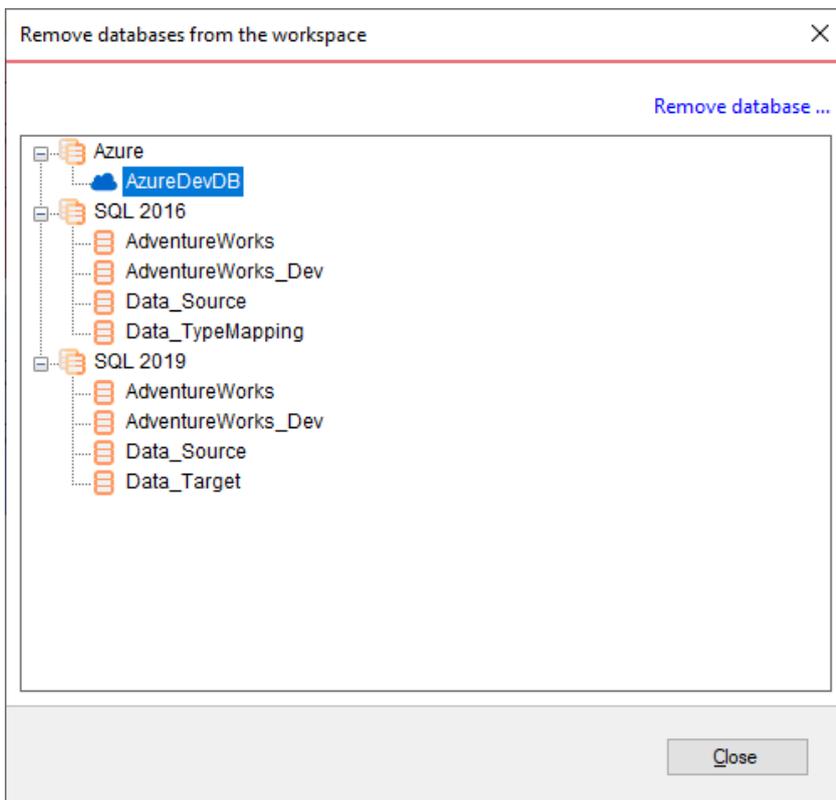
Other connection settings, specified as name=value, separated by semicolons: name1=value1;name2=value2;...

ConnectRetryCount=2;ConnectRetryInterval=5

SQL Data Compare Remove Databases

Using IDERA SQL Data Compare, there are two ways to remove a database from the workspace:

- Move the mouse over the name of database you want to remove under the Registered Databases on the left panel of the Workspace. The database name is highlighted and a **Remove** link is available.
- Click **Remove Databases** on the Home tab of the IDERA SQL Data Compare ribbon. A remove databases dialog window appears showing all the registered servers/databases. Select the database you want to remove, and then click **Remove Selected Database...** at the top right corner.



SQL Data Compare Comparing

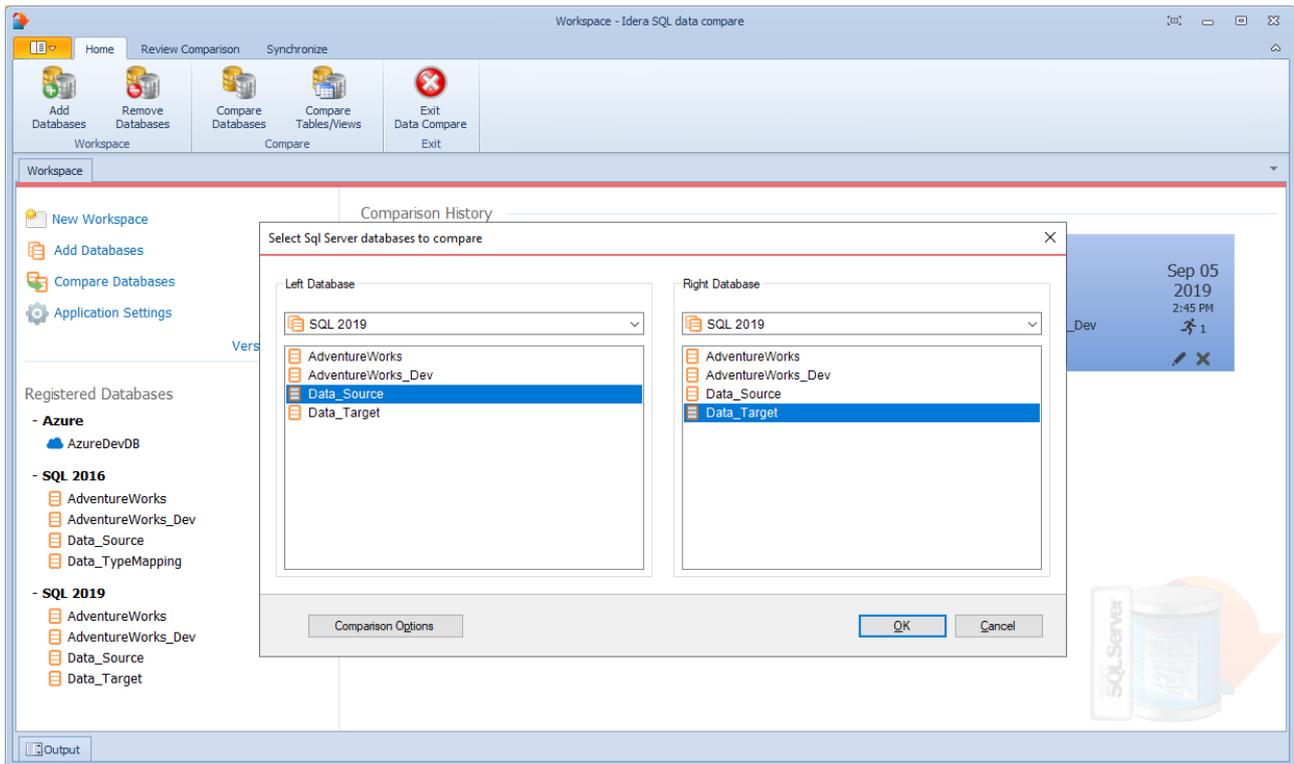
Comparing the data in two databases is part of the core functionality of IDERA SQL Data Compare.

The databases you wish to compare (or tables within them) can be virtually anywhere – as long as you have access to them you can compare and synchronize them. Furthermore, the databases do not have to be of the same version or flavor – you can compare between any “flavors” of the SQL Server: Express, Developer, Standard, Enterprise.

SQL Data Compare Comparing Databases

To start a comparison between two databases click on the IDERA SQL Data Compare **Compare Databases** button on the Home tab of the ribbon. The SQL Server Database Selection dialog window that appears allows you to chose the server(s) where the databases you wish to compare reside as well as the databases themselves.

From this window, you can also access the [Comparison Options](#) that are applied during the comparison.



SQL Data Compare Comparing Select Tables/Views

When you choose to compare two databases, tables in those databases are mapped with each other by name, i.e. an Employee table on the source database is mapped to the same Employee table on the target database. Mapping rules could alter the name-mapping, but not my much.

If you wish to compare a few specific tables (or views), that do not share the same name, click **Compare Tables/Views** on the SQL Data Compare ribbon.

i In addition to tables, data compare has the ability to compare views as well, if the comparison option **Compare and Synchronize Views** is checked. Synchronizing views however may not always succeed.

The following considerations apply to views:

- Not all views are updatable.
- If a view has indexes, data compare will select one in the same order as the tables indexes; otherwise you must define a view custom index.
- Since SQL Server does not support SET IDENTITY_INSERT ON | OFF on views, the insert statements might fail if one of the view's underlying tables contains an identity column and the identity column is included in the view columns.
- Data compare cannot synchronize views that contain large binary fields such as varbinary(max) and image, or views with large text field such as varchar(max), nvarchar(max), text and ntext.

Once you make your selections, click **Read Database Objects**. This populates two list boxes with the database tables and optionally the database views.

To map two objects, select the first object on the left list box, then select the second object on the right list box and click the "left-to-right" arrow button. The mapping appears in the Mapped Objects list box. To remove a mapping, selected the mapped objects and click on the "right-to-left" arrow button.

Select the database tables to compare.

Left Database

Server:

Database:

Show objects that start with:

Right Database

Server:

Database:

Show objects that start with:

Read Database Objects

🔍

Type	Schema	Name
	Human...	Department
	Human...	Employee
	Human...	EmployeeAddress
	Person	Address
	Person	Contact
	Person	CountryRegion
	Product...	Product
	Product...	ProductDescription
	Product...	ProductModel
	Product...	ProductReview
	Sales	Customer
	Sales	CustomerAddress
	Sales	SalesPerson
	Sales	SalesTerritory

Mapped Objects

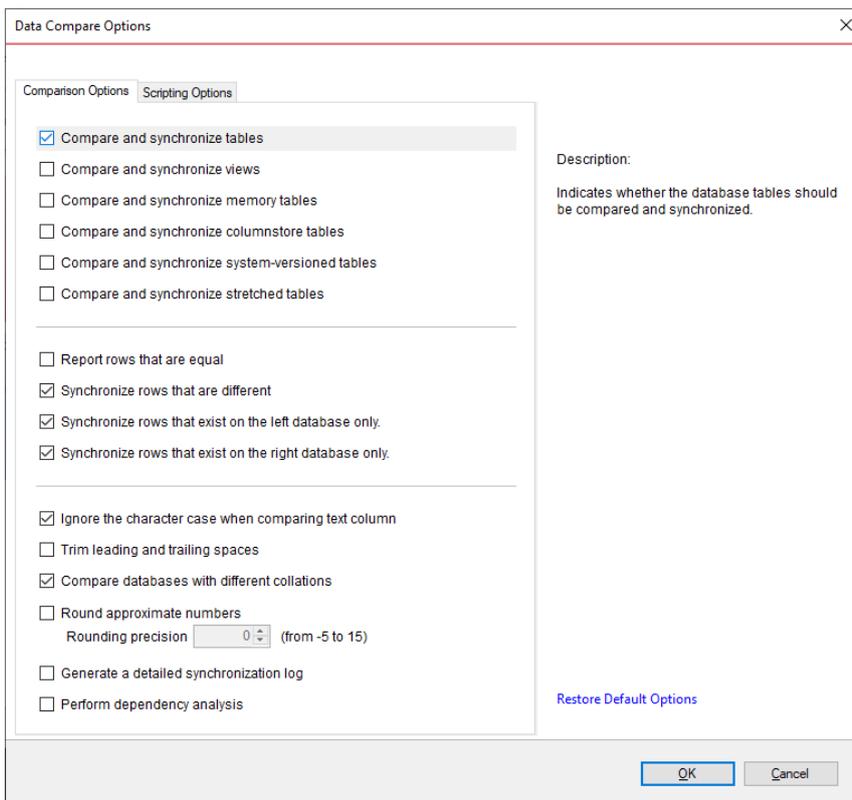
Type	Left Object	Right Object
	[HumanResources].[Employee]	[HumanResources].[Employee]
	[HumanResources].[EmployeeAddress]	[HumanResources].[EmployeeAddress]
	[Person].[Address]	[Person].[Address]
	[Person].[Contact]	[Person].[Contact]
	[Person].[CountryRegion]	[Person].[CountryRegion]

Table: Production.Product

SQL Data Compare Comparison Options

IDERA SQL Data Compare exposes a series of options that allow you to customize the way the comparison is performed. On mouse over a description of the highlighted option is displayed on the right panel of the options window.

You can access the Data Compare options from the Compare Databases and Compare Tables dialog windows before the object mapping operation. After the objects (tables and indexed views) have been mapped you can access the Comparison Options from the Review Comparison tab of the ribbon as well as from the action links on the right panel of the Mapped Objects tab.



SQL Data Compare Scripting Options

Scripting Options are listed on the second tab of the IDERA SQL Data Compare Options window. Those options affect the way the synchronization script is generated but do not affect the comparison operation.

- **Transaction size.** Determines the max. size of the SQL Server transaction, under which the synchronization script runs. In some cases, the synchronization script can get extremely large, with millions of insert/update/delete statements. Wrapping such a large script in a single transaction may not be possible, due to the transaction size limitations, or acceptable, for performance and other reasons. Setting a max. transaction size is a way around these restrictions. If the transaction has a max. size, data compare commits the current transaction when the max. size is reached and starts a new one.

To have a transaction with no upper limit, set the max transaction size to 0.

i Please use this option with caution. If you set a max. transaction limit and the synchronization process is canceled, only the current, open transaction is rolled back. Changes committed prior to that transaction are permanent and cannot be reversed. We strongly recommend, if possible, to leave the transaction unlimited, since it ensures that all changes in the database are either committed or rolled back.

- **Batch size.** Determines the size of data that is transmitted to SQL Server in one execution. A transaction normally contains many batches, but the batch itself is a single unit of work that cannot be divided any further.

The screenshot shows the 'Data Compare Options' dialog box with the 'Scripting Options' tab selected. The 'Generate and execute a transactional script' checkbox is checked, with a 'Transaction size (in MB)' spinner set to 256 (up to 2048 MB). The 'Execution batch size (in KB)' spinner is set to 16 (1 - 64). Other options include 'Log script execution' (unchecked), 'Disable DML triggers' (checked), 'Disable system-versioning of the temporal tables' (unchecked), 'Drop/Create' (dropdown), 'Action on foreign keys' (dropdown), and 'Local and Remote' (dropdown). A 'Description' section explains the transactional script option, and a 'Remarks' section in red text warns against turning off the transactional option. 'OK' and 'Cancel' buttons are at the bottom.

SQL Data Compare Preparing the Comparison

Before comparing the content of two databases, IDERA SQL Data Compare will analyze the schemas of both databases (or the selected tables). If you manually map the tables between the two databases, SQL Data Compare will use that mapping as is and proceed with analyzing the pairs of tables. If you compare whole database it will first perform an automatic mapping of the tables based on their names (owner name + table name) - every table on one database is paired with a table (when possible) on the other database. Then, it proceeds to analyze the pairs of tables to map individual columns, determine the available comparison keys and choose a key that will be used to pair rows during the data compare.

During this preparation phase you will be able to customize the object mapping, include and exclude certain tables and certain columns within those tables, choose comparison keys and define data filters to compare only subsets of rows from certain tables.

SQL Data Compare Mapped Objects

The results of the schema analyses and the automatic mapping operation are displayed on the IDERA SQL Data Compare Mapped Objects tab. For each pair of mapped objects the following information is displayed:

- **Object Type.** The object can be a table or an indexed view. The type is indicated through the small icon in the second column of the grid.
- **Compare.** A checkbox indicating whether the pair should be included in the comparison. By default, all mapped pairs will be included in the comparison. You can quickly select / deselect all objects using the **Select All Objects** and **ClearSelection** commands in the ribbon.
- **Owner and object name.** For both sides. In a default mapping the Left Owner and Left Name would be identical to respectively Right Owner and Right Name.
- **Mapping Status.** A blank value indicates that the objects have been fully mapped, that is, there was a complete match of columns. A **Partial** value indicates that some columns were not matched and will not be included in the comparison.
- **Left/Right Keys.** Those are the unique keys that have been selected by SQL Data Compare to be used in the mapping of the rows during the data comparison. If those two fields are empty for a given pair that means that SQL Data Compare was not able to identify a unique key that could be used. If you do not choose a custom key for this pair then those objects will not be included in the comparison.

i **Changing the comparison options.** Please note that changing the comparison options after the mapping operation has completed will trigger a refresh of the Mapped Objects tab. For example if you uncheck the **Compare and synchronize tables** option the table pairs will be removed from the mapped objects grid.

The screenshot displays the IDERA SQL Comparison Toolset interface. The main window is titled "New Compare* (1) - Idera SQL data compare". The interface includes a ribbon with tabs for "Home", "Review Comparison", and "Synchronize". The "Home" tab is active, showing options like "Compare Data", "Refresh Result", "Comparison Options", and "Command Line Config". The "Review Comparison" tab shows "Next Difference", "Previous Difference", and "Toggle Equal Objects". The "Synchronize" tab shows "Select All Objects", "Clear Selection", "Toggle Tables", "Toggle Views", "Toggle Invalid Objects", and "Font" settings.

The workspace area shows two SQL 2016 databases: "AdventureWorks" and "AdventureWorks_Dev". Below this, the "Mapped Objects" table is displayed. The table has columns for "Compare", "Left Owner", "Left Name", "Mapping", "Right Name", "Right Owner", and "Left Key". The "Compare" column contains checkboxes, and the "Left Key" column contains primary key names. The table lists various database objects such as tables, views, and stored procedures, along with their respective owners and primary keys.

On the right side of the interface, there are three panels: "Compare Data", "Comparison Options", and "Mapping Rules". The "Compare Data" panel contains a play button and the text "Perform data comparison. The result will be displayed in the Comparison Result tab." The "Comparison Options" panel contains a gear icon and the text "View or change data compare options. Note that if options are changed, the 'Comparison Result' tab will be refreshed." The "Mapping Rules" panel contains a link icon and the text "Mapping rules determine how database objects and data types are paired. The default rules are based on the name of the object."

SQL Data Compare Mapping Rules

In most cases the default mapping mechanism that pairs the objects based on the schema name and the object name is what is needed. However, there are often scenarios in which the same table may be owned by different schemas in different databases, or the tables on the development server for example may have the names prefixed with dev_ whereas the same tables in the production server do not have that prefix. In such cases you may want to define certain mapping rules that would allow you to pair objects that would otherwise not be paired together by IDERA SQL Data Compare.

The Mapping Rules can be accessed from the ribbon or from the action links on the right panel of the comparison tab.

There are three types of rules you can set:

- **Schema Mapping Rules.** By default SQL Data Compare performs an exact match of the schema names, however you have two additional options:
 - **Ignore schema name.** In this case table [dbo].[T1] on database 1 will be paired with table [user1].[T1] on database 2, in other words the schema name is irrelevant.
 - **Manually map the schema names.** This allows you to choose which schema name from database 1 to map to which schema name from database 2. If you map schema [S1] from DB1 to schema [S2] from DB2 then [S1].[T1] from DB1 will be paired with [S2].[T1] from DB2. To un-map a pair of schemas double click on the icon in the middle of the schema pairs on the Mapped Schemas box; that will break the mapping and move the pair of schemas to the **Unmapped Schemas** area. To map two schemas click on the schema on the left to select it; click on the schema you want to map it to on the right box and then click on the **Map Selected Schemas** link.
- **Name Mapping Rules.** By default SQL Data Compare performs an exact match on the object names, however, you can choose to ignore certain prefix and/or postfix. For example you can indicate that on the left database the dev_ prefix and the _old postfix and on the right database the qa_ prefix and the _new postfix should be ignored. In such case table [dev_T1_old] from the left database will be paired with [qa_T1_new].
- **Data Type Mapping Rules.** By default data types are mapped by compatibility. If column C1 of table T1 on DB1 is of the type varchar but the same column of table T1 on DB2 is of type nvarchar then those two columns will be paired together since those two data types are compatible with each other. However, you have the option to enforce an exact type match for your comparison. When you choose the **Map data types by name** then two columns will only be paired if they are of the same exact type.

Create the rules to use when database objects are mapped ✕

Tables

Schema Mapping Rules **Name Mapping Rules** Data Type Mapping Rules

Match the exact name

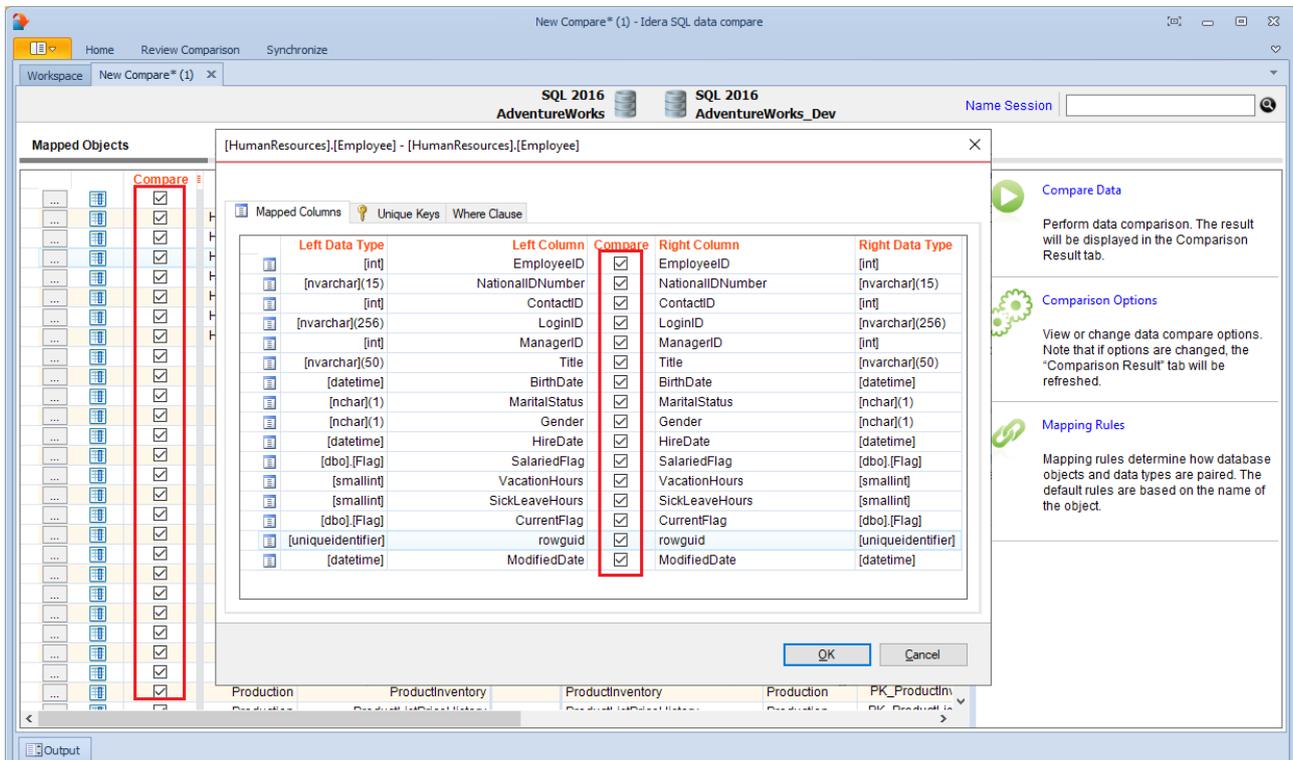
Ignore the name prefix, such as the tmp in the tmp_Employee
Source table prefix Target table prefix

Ignore the name postfix, such as the tmp in the Employee_tmp
Source table postfix Target table postfix

SQL Data Compare Exclude Objects

The first step in customizing your comparison after the objects are paired is to choose which objects you wish to include in the comparison and for each object pair which columns you wish to exclude. The options available in IDERA SQL Data Compare:

- **Exclude objects.** By default all objects (tables and indexed views) are marked to be included in the comparison. You can decide which pairs you want to exclude by un-checking (clearing) the **Compare** checkbox for each object pair. To de-select or select all object pairs in one click you can use the **Select all objects** and **Clear selection** buttons in the ribbon.
- **Exclude columns.** For each object pair you can click on the details button (the first column of the Mapped Objects grid) to further customize the comparison. By default all the mapped columns are marked to be included in the comparison. However, you may choose to exclude certain columns from the comparison (example: image or text type columns that would take a long time to compare). To exclude columns simply uncheck the **Compare** checkbox for that column pair. Important: note that the column(s) that are part of the key that has been selected to be used as the comparison key will always be included in the comparison regardless of whether you have checked or un-checked them in the Mapped Columns tab.



SQL Data Compare Comparison Keys

The IDERA SQL Data Compare Unique Keys tab is divided in two main sections. The top section contains two boxes with the list of unique keys that are potential candidates to be used as the comparison key for each table in the pair. Initially the boxes contain only the predefined unique keys that SQL Data Compare found, but you can add other unique keys that can be a combination of any and all the columns on the given table. The bottom section contains a box with the pair of unique keys that have been selected as the key to be used for the data comparison operation. SQL Data Compare picks the comparison key in the following order (whichever is found first):

1. User-defined keys. User-defined keys (or custom keys) takes precedence over the built-in keys and indexes.
2. Primary Key
3. Unique Constraint
4. Unique Index. Indexes that are not unique are not considered.

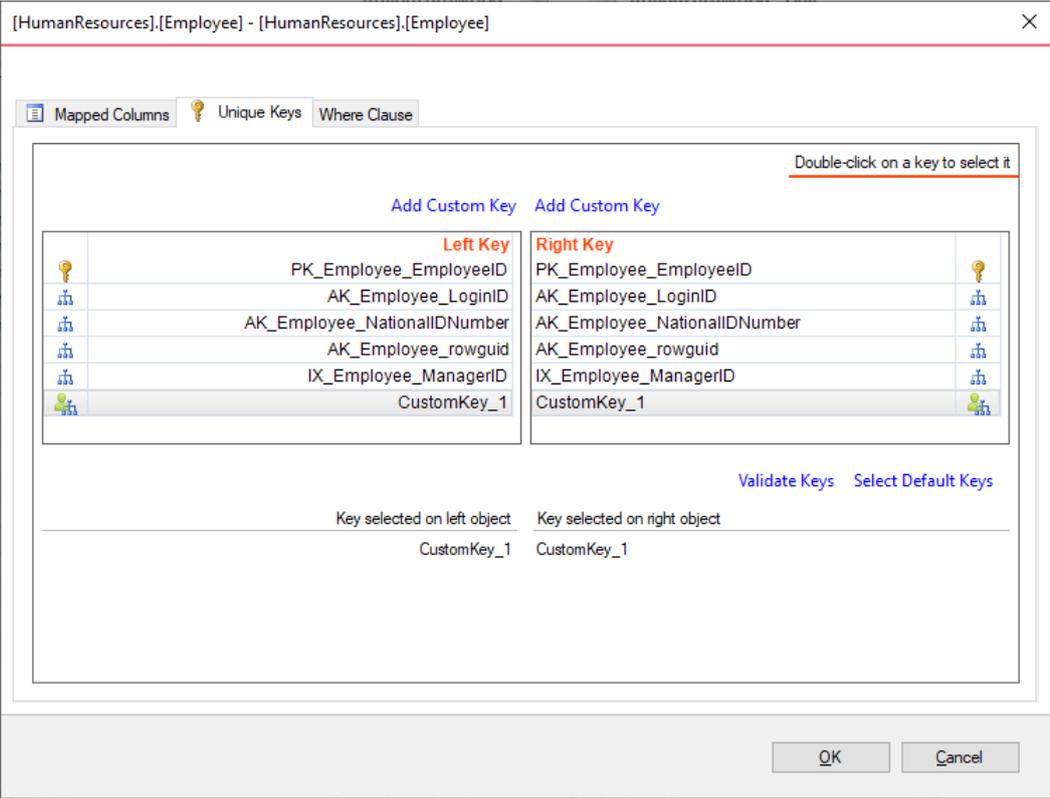
When the selected keys are not those that were picked by SQL Data Compare you should validate the keys by clicking on the **Validate** button. If the selected keys are not valid you will not be able to compare the those tables. By default the keys are only validated for type compatibility. If you wish to validate your custom defined keys for uniqueness then you should check that option in the Application Settings.

User-defined key validation is done as follows:

1. Columns participating in the the user-defined key in both objects in the pair must have the same names
2. Columns that participate in the key must have the same data type. Only the name of the type is checked. The length and other type properties are not considered.
3. The uniqueness of the key is checked only if the **Check custom key uniqueness** option in the application settings under Miscellanies section is checked.

Selecting a key for the comparison. Double-clicking on a unique key on the top section will replace the selected key with the new one.

 User-defined keys are not supported on Memory Tables.



SQL Data Compare Where Clause

For large tables the comparison operation may take time since all the rows from both sides will have to be read and compared. So, if you know for example that out of 10 million rows that 9 million of them are equal it would be very inefficient to compare those equal rows. Furthermore, regardless of how many rows may be different you might need to do a quick comparison of just a subset of rows, for example, only rows that were modified today.

IDERA SQL Data Compare allows you to set where clauses for each object being compared so that you can read and compare only the rows you need to. When defining a where clause by default the same filter will be applied to both tables in the pair, however, if you wish to define different filters for each of the tables you can uncheck the **Use the same where clause** checkbox. We suggest you validate the data filters to ensure they have been defined correctly.

i Do not write the "where" word on the clause box. Simply write the condition the way you would write it in T-SQL (see the screen shot below for an example).

[HumanResources].[Employee] - [HumanResources].[Employee]

Mapped Columns Unique Keys Where Clause

Enter the where clause for each object in the corresponding text box. [Validate where clause](#)

[AdventureWorks].[HumanResources].[Employee] **Status: Valid**

EmployeeID > 100

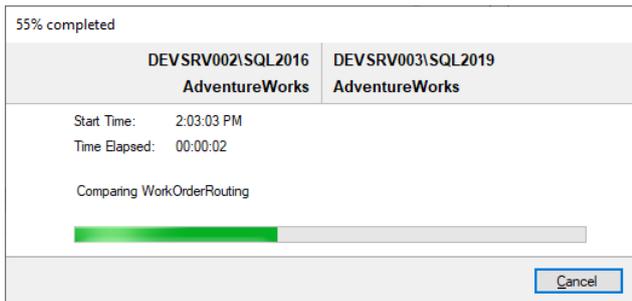
Use the same where clauses for both objects

[AdventureWorks_Dev].[HumanResources].[Employee] **Status: Valid**

OK Cancel

SQL Data Compare Comparing Data

After completing the preparation phase of the comparison you are ready to compare the data in all those mapped objects (tables and views). To start the comparison either click on the **Compare Data** button on the ribbon (under the Review Comparison tab) or click on the **Compare Data** on the right panel of the Mapped Objects window. A progress window displays the start time, elapsed time and the names of the objects being compared. If you cancel the comparison operation IDERA SQL Data Compare will display partial comparison results.



On completion of the data compare operation the results are displayed on a new window called Comparison Result within that comparison session's tab. The Comparison Result window is divided into multiple sections:

- **Object grid.** Displays the tables that were compared, indicating the total number of rows, the number of rows that were equal, number of rows that were found only on either the source or the target table and number of rows that are different. Tables that have data differences are displayed in bold.

i If the comparison is interrupted, data compare displays the comparison result up to the point of cancellation and marks the objects that were not compared with a question mark. If an object is marked as "Invalid" in the comparison grid, it either didn't have a data key or the keys were invalid.

- **Data grid.** When you click on a pair of tables that are different, depending on the data differences, up to three tabs will appear on the bottom section of the Comparison Results window: "Left Rows"; "Right Rows"; and "Different Rows". For the different rows the paired columns from both tables are shown next two each other so that you can easily see the differences. You can "zoom in" on a particular row to take a closer look - right click on the row you wish to inspect and click on "View Row Details".

i Notes on details grid:

- Columns in the light-blue color represent data key columns. The "key" label appears in the column header as well.
- Grid cells in orange color indicate data differences.
- By default, equal rows are not displayed in the data grid unless the comparison option **Report rows that are equal** is checked.

- **Synchronization script links.** These links generate the synchronization script for either database.

- **Action links.** Performs various data operation.
- **Legend.** Provides a list of symbols used in the object grid.

The screenshot displays the IDERA SQL Comparison Toolset interface. The main window shows a comparison result table with columns for Status, Sync, Left Owner, Left Name, Right Name, Right Owner, Total Rows, and Equal Rows. The table lists various database objects like Department, Employee, Address, Contact, Product, and Sales. To the right of the table, there are action links such as 'Generate the script for SQL 2019.Data_Target' and 'Generate the script for SQL 2019.Data_Source'. Below the table, there is a legend section with icons for Table, View, Missing data key, Excluded object, and Object did not compare.

Below the main comparison result, there is a detailed view of differences (labeled '2') showing a table with columns for row number, jffix, EmailAddress, EmailPromotion, Phone, PasswordHash, and Password. The table shows three rows of differences between the source and target data.

Row	Left EmailAddress	Right EmailAddress	Left EmailPromotion	Right EmailPromotion	Left Phone	Right Phone	Left PasswordHash	Right PasswordHash	Left Password
1	catherine0@adventur...	catherine0@yahoo.c...	1	1	747-555-0171	747-555-0171	19712A42FC40F1...	19712A42FC40F1...	rpyd5Tw=
2	kim2@adventure-wor...	kim2@yahoo.com	0	0	334-555-0137	227-645-0137	CB CD65769648C...	CB CD65769648C...	rrgbG/U=
3	humberto0@adve...	humberto0@adve...	2	2	599-555-0127	599-275-0127	69CF91B5628FA4...	69CF91B5628FA4...	F5qyyxs=

SQL Data Compare Synchronizing Data

Identifying the data differences between two databases helps you as a DBA/Developer in many ways but often your job does not end there - you need to migrate those changes from one database to another. IDERA SQL Data Compare allows you to do just that - after comparing the contents of two databases you can use it to generate the synchronization script - a T-SQL script that will update the target database (or just the selected tables/rows) to make it the same as the source database.

The data synchronization operation consists of three key actions:

- **Update.** All rows that exist in both sides will be updated on the target database to match the source.
- **Insert.** All rows that exist only on the source database but not on the target will be inserted on the target database.
- **Delete.** All the rows that exist only on the target database but not on the source will be deleted from the target database.

In case you do not wish to delete the rows that only exist on the target database you can change the comparison options so that those rows are not included in the synchronization.

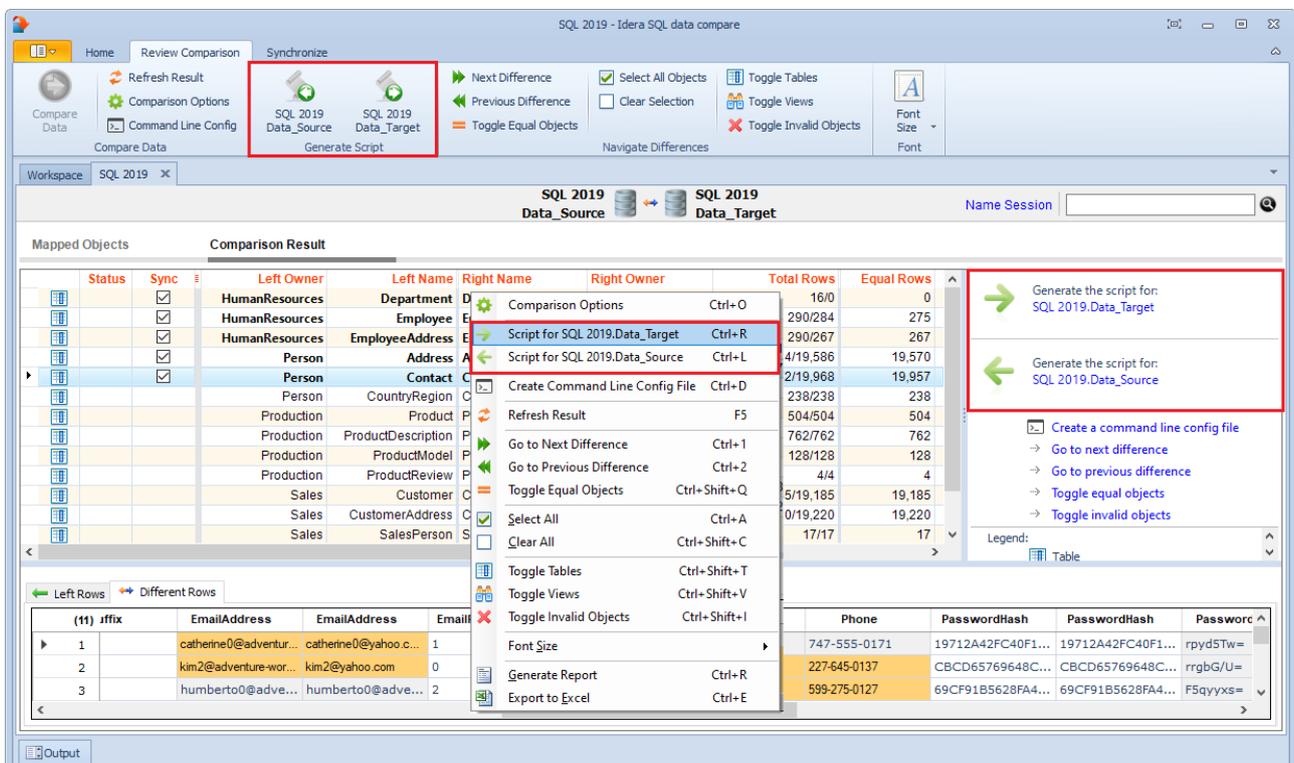
SQL Data Compare Generating the Script

Once the data comparison operation is finished you are now ready to start the data synchronization operation.

By default, all tables that are found to have differences and all the different rows in those tables are automatically marked to be included in the synchronization. You can however easily change which tables, and which rows in those tables, should or should not be included in the script by checking or un-checking the **Sync** checkbox.

Once you have made your selections you can generate the synchronization script to either change the database on the right of the comparison to make it the same as that on the left or the other way around. There are three ways to start the script generation:

- **Ribbon.** In the Review Comparison tab of the ribbon there are two big buttons in the middle under the grouping **Generate Differences Script**. Each button is labeled with the name of the server + name of the database for which the script will be generated.
- **Action Links on the right panel.** Are listed under the label **Generate the script for:**. Clicking on the [server].[database] link generates the synchronization script that will make that database the same as the other one.
- **Context menu.** When you right click on the comparison results grid a context menu will pop up. The context menu allows you to access the comparison options and trigger the generation of the synchronization script in addition of other context relevant actions.



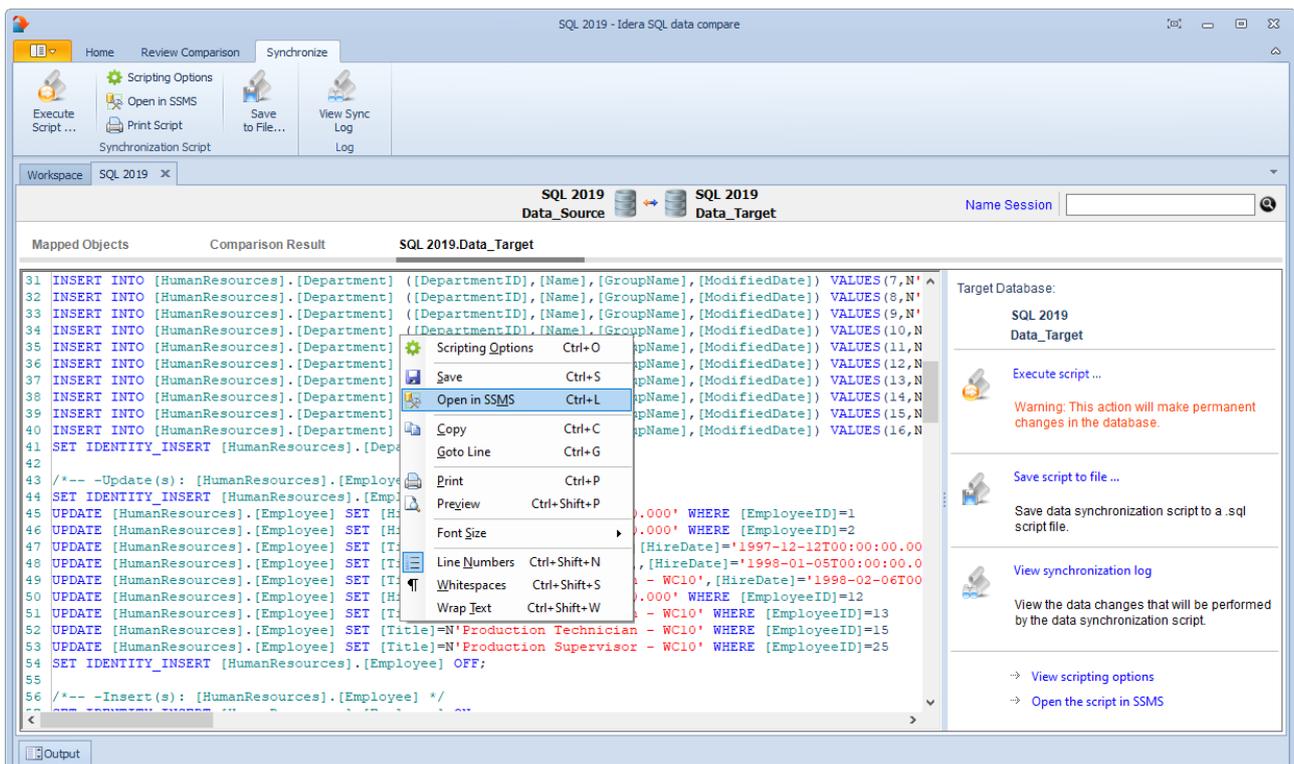
SQL Data Compare Working with the Script

The data synchronization script will be displayed on a new window, within the comparison session's tab. A context menu allows you to save the script, copy it, print it, add line numbers, jump to a certain line in the script, etc.

You can also open the script in SQL Server Management Studio (SSMS) assuming you have the SSMS installed on your machine. Simply right click anywhere on the script and click **Open SSMS** or click on the **Open in SSMS** button in the ribbon.

In addition to the main script, data compare generates also a synchronization log, which can be accessed from the ribbon or from the action links on the right panel. The synchronization log contains a summary of the actions that will be performed by the synchronization script on the target database.

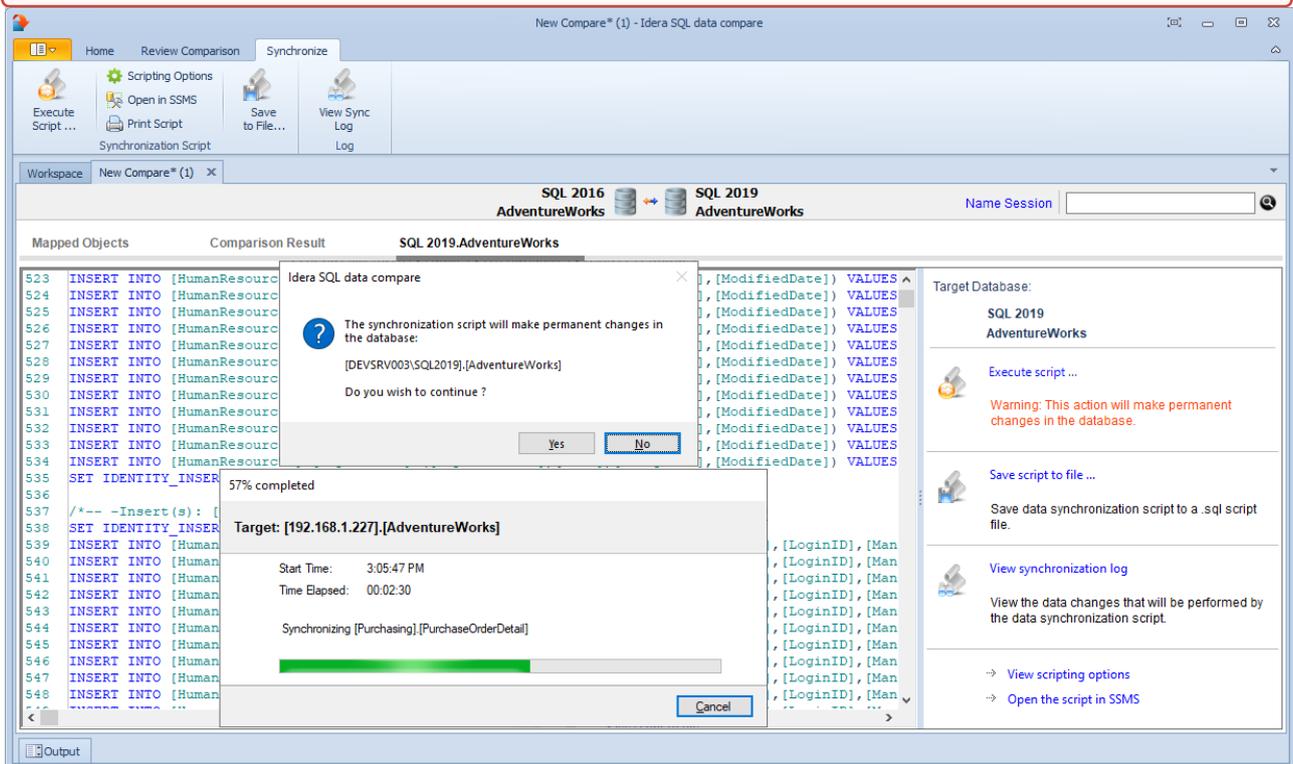
 If the synchronization script is too big, data compare will display only a portion of it. If you wish to view the whole script, you can save it to a file and open it with an external text editor.



SQL Data Compare Executing the Script

Once you have reviewed and you are comfortable with the synchronization script you can execute it on the target database.

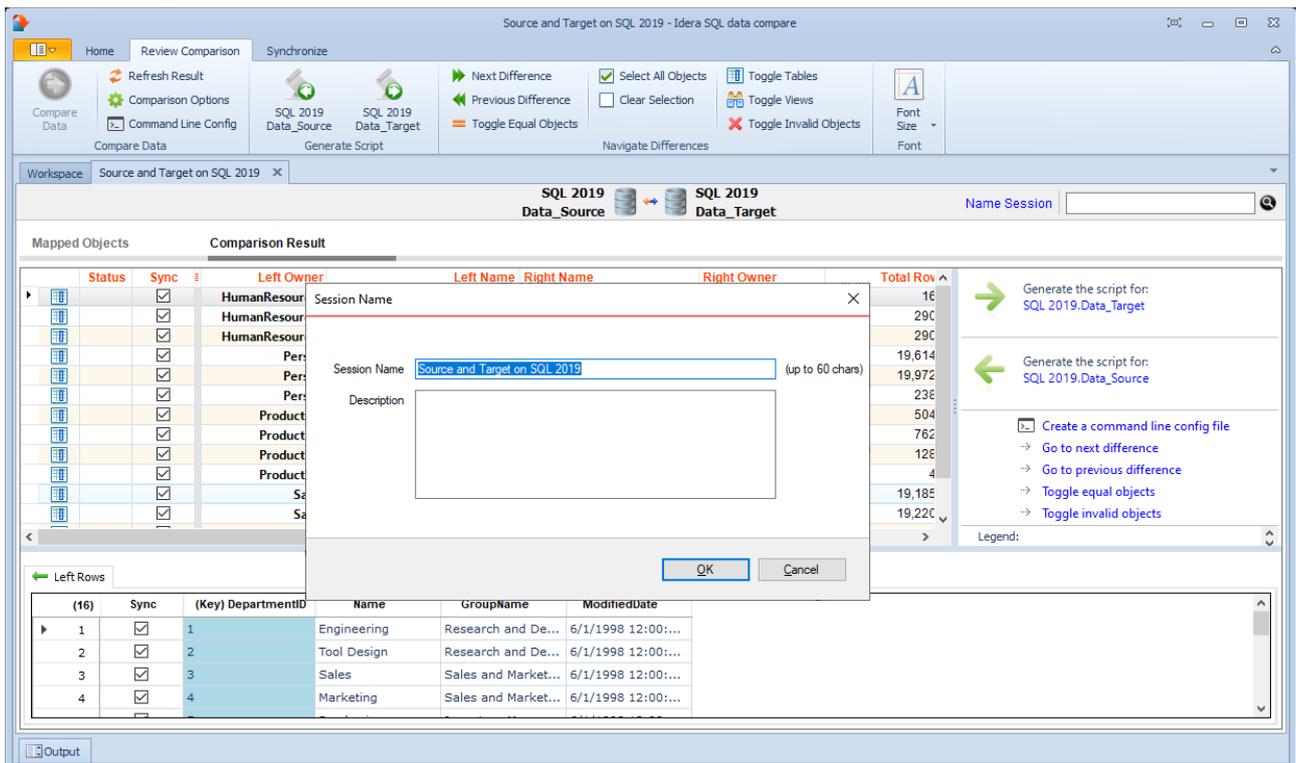
- ⚠ This action will make permanent changes to the content of the target database. It is common sense and we strongly recommend that before executing the script you:
- Make a full backup of the target database;
 - Save the script you are about to execute so that you know exactly what was done to the database and when.



SQL Data Compare Comparison Sessions

The preparation phase of the data compare can be a time consuming process if you need to exclude certain objects, do custom mapping, define custom comparison keys and set data filters. IDERA SQL Data Compare automatically saves all your preparation work so that you can repeat the same comparison in the future with a single click without having to go through that process again.

You can also name the comparison sessions to make it easier to identify them. Up to 25 sessions are saved and displayed on the Workspace tab.



IDERA SQL Data Compare Command Line

IDERA SQL Data Compare command line provides for comparing and synchronizing the content of databases via the command line. With the command line you can:

- Perform automated data comparison and synchronization;
- Compare and synchronize database content as part of your setup and deployment solution;
- Compare database content and execute the data synchronization script in batch files;
- Schedule the comparison.

The command line includes:

- A wizard for generating the command line xml config file
- the command line utility that uses the xml config file to compare and synchronize the data between SQL Server databases.

The command line comes with samples for most typical data comparison scenarios. The sample configuration files are located under **\Program Files (x86)\Idera\SQL Comparison Toolset v<n>\Command Line Samples\Data Compare**, where **n** is the Toolset version.

Using the SQL Data Compare Command Line

Data compare command line expects and reads the comparison settings from an xml config file. The config file contains the SQL Server instance, its credentials, the SQL Server database, the comparison options, excluded objects, mapping rules, the output files and many more. You can generate the config file using the command line wizard.

Running the command line, using the config file config.xml, is as simple as follows:

```
IderaDataCmd <path>\config.xml
```

You can also use the command line to validate a config file, without comparing the SQL Server databases. This can be useful if you create the config file manually. Validation is performed via the /v parameter:

```
IderaDataCmd <path>\config.xml /v
```

To view the command line usage, use the help parameter: /? /h or /help

```
IderaDataCmd /?
```

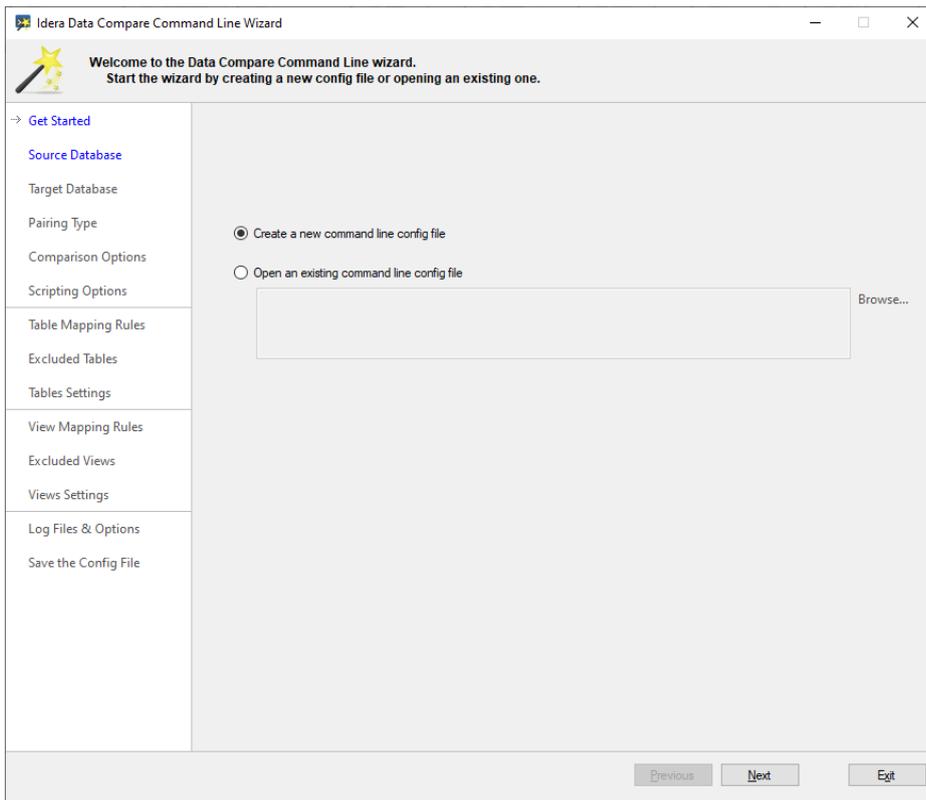
 Due to the complexity of the config file, we strongly recommend that you use the command line wizard to generate the xml config files.

Starting with the Command Line Wizard

The command line wizard, included with Idera SQL Comparison Toolset, allows you to create a new xml config file or modify an existing one. The wizard hides the complexity and generates a consistent, syntactically correct and well-optimized config file, especially for complicated data comparison scenarios.

To launch the command line wizard, go to **Start > Idera SQL Comparison Toolset > Data Compare Command Line Wizard**.

You can start by creating a new config file or opening an existing one.



i When you open an existing config file, the wizard performs a thorough validation of that file. If the xml file is correct, you will see a green check mark to the left of the filename. If the xml file is incorrect, depending on the severity of the problem, the wizard could generate warnings or reject the file entirely.

Specifying the Source and Target Databases

The source and target database steps allow you to specify the SQL Server databases that you want to compare. For more detail about the database connection settings, see [SQL Data Compare Add Databases](#).

i Even though we refer to the SQL Server database as the "source" or "target", the command line can switch them, so the source can become the target and vice versa. More details on setting the target database are included in the Log Files and Options step.

Below is the xml fragment that contains the SQL Server databases:

```
<LeftDatabase>
  <SqlServer>DEVSRV003\SQL2019</SqlServer>
  <DatabaseName>AdventureWorks</DatabaseName>
  <TrustedConnection>>true</TrustedConnection>
</LeftDatabase>
<RightDatabase>
  <SqlServer>DEVSRV002\SQL2016</SqlServer>
  <DatabaseName>AdventureWorks</DatabaseName>
  <TrustedConnection>>true</TrustedConnection>
</RightDatabase>
```

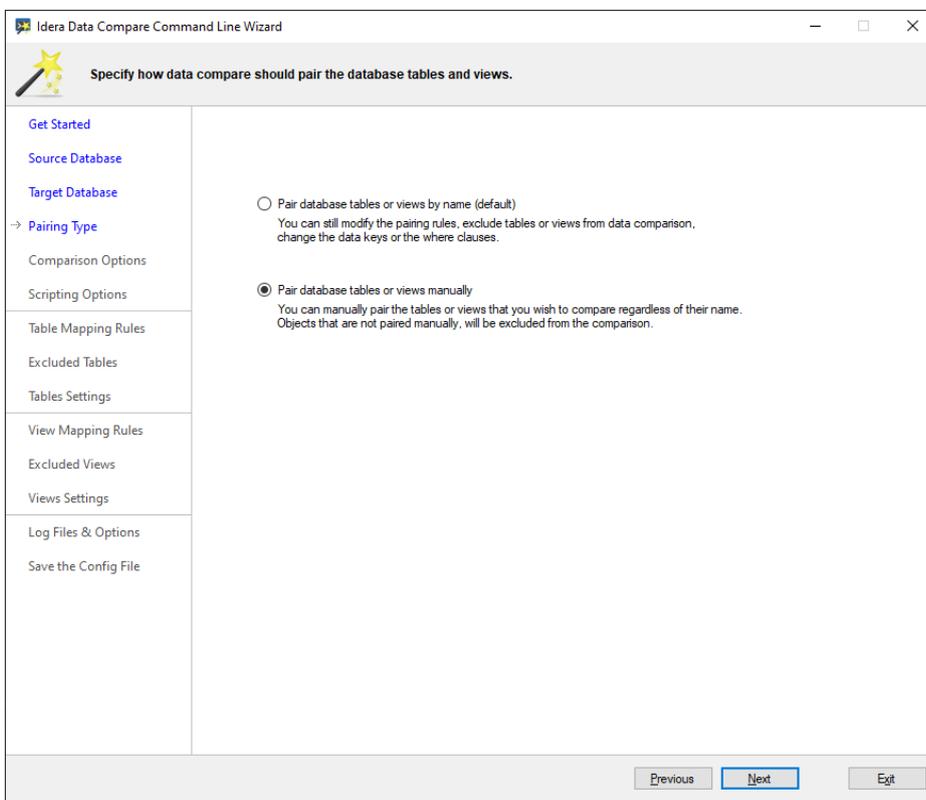
Pairing Tables and Views

The Pairing Type step of the wizard allows you to specify the pairing method that should be used to map the database tables with each other.

Before data can be compared, tables (and views if needed) must be paired with one another, that is an Employee table on the source database must have the same Employee table on the target database to compare with. This can be done in two ways:

- **Default pairing.** Tables are paired together by name.
- **Manual pairing.** Tables must be paired manually. This method is suitable when you wish to compare a handful of tables that have different names.

The default pairing is the recommended method. In the data compare UI, the default pairing method is used when you compare two databases. The manual pairing is used when you compare specific tables.

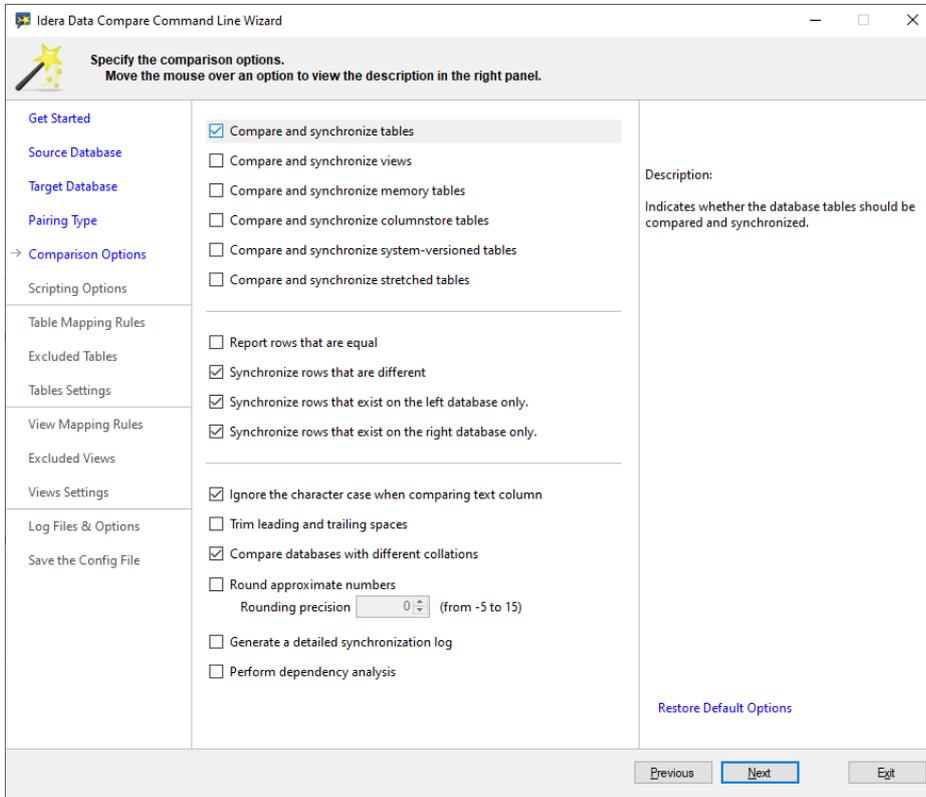


The manual pairing sets the following option in the xml config file:

```
<CommandLineOptions>
  <MappingType>Custom</MappingType>
</CommandLineOptions>
```

Comparison Options

The Comparison Options step of the wizard allows you to change the comparison options. The Command Line wizard saves only those options that are different from the default.



The following fragment is an example of the xml generated when some comparison options are changed:

```
<DataCompareOptions>
  <CompareMemoryTables>true</CompareMemoryTables>
  <CompareSystemVersionedTables>true</CompareSystemVersionedTables>
  <CompareStretchedTables>true</CompareStretchedTables>
  <GenerateDetailedLog>true</GenerateDetailedLog>
  <RoundApproximateNumber>true</RoundApproximateNumber>
  <RoundPrecision>5</RoundPrecision>
</DataCompareOptions>
```

Scripting Options

The Scripting Options step of the wizard allows you to change the scripting options. The Command Line wizard saves only those options that are different from the default.

The following fragment is an example of the xml generated when some scripting options are changed:

```
<DataCompareOptions>
  <DisableSystemVersioning>true</DisableSystemVersioning>
  <StretchedTableQueryScope>LocalOnly</StretchedTableQueryScope>
  <CompareViews>true</CompareViews>
  <LogScriptExecution>true</LogScriptExecution>
  <ScriptExecutionLogFile>D:\QA\Idera\SqlData\Cmd\Logs\exec.txt</
ScriptExecutionLogFile>
</DataCompareOptions>
```

Table Mapping Rules

The Table Mapping Rules step allows you to define how data compare should perform the name-matching of the database tables. By default, tables with the same name, on both database, are paired with one another and compared. You can use mapping rules if your databases have tables that do not have the exact name, but are different in a consistent way. If there are tables in the development environment, whose name starts with DEV_, but in production the same tables start with PROD_, then you can use the following mapping rule to map them. Without the mapping rule, these tables will be excluded.

The following fragment is an example of the xml generated for this mapping rule:

```
<TablesMappingRules>
  <NameMapping MappingRule="IgnorePrefix" IgnoreLeftName="DEV_"
  IgnoreRightName="PROD_">
</TablesMappingRules>
```

Excluding Tables

The Excluded Tables step allows you to exclude database tables by name.

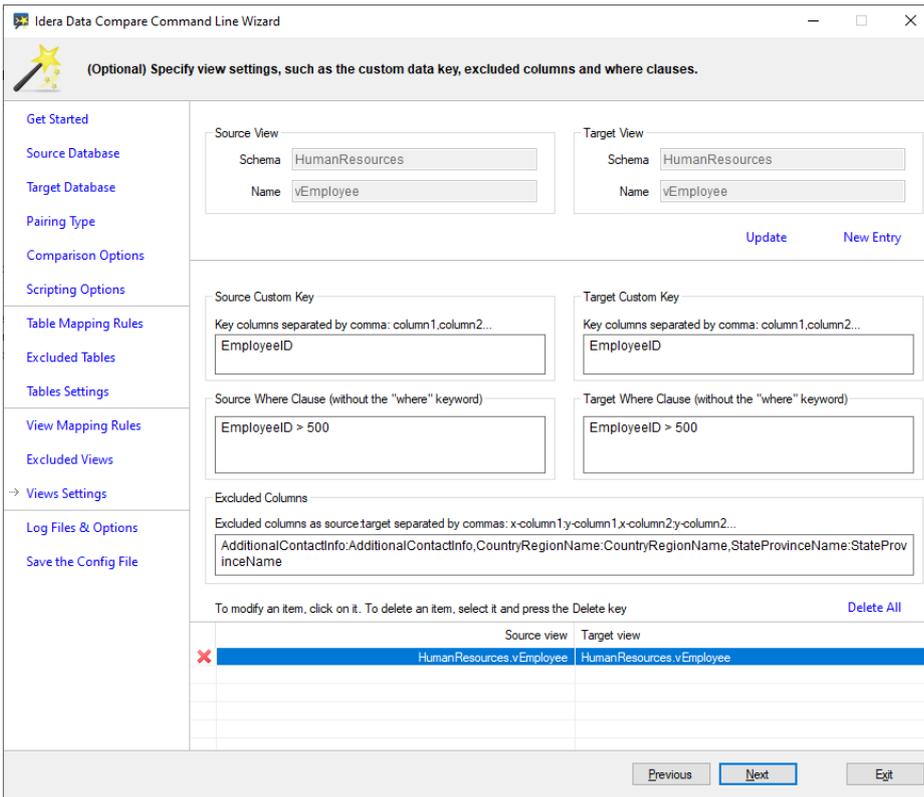
The following fragment is the xml generated when some tables are excluded by name:

```
<TablePairs>
  <!--**** excluded tables ****-->
  <Pair Action="Exclude">
    <LeftObject Schema="HumanResources" Name="EmployeeAddress" />
    <RightObject Schema="HumanResources" Name="EmployeeAddress" />
  </Pair>
  <Pair Action="Exclude">
    <LeftObject Schema="Sales" Name="SalesOrderDetail" />
    <RightObject Schema="Sales" Name="SalesOrderDetail" />
  </Pair>
  <Pair Action="Exclude">
    <LeftObject Schema="Sales" Name="SalesOrderHeader" />
    <RightObject Schema="Sales" Name="SalesOrderHeader" />
  </Pair>
  <Pair Action="Exclude">
    <LeftObject Schema="Sales" Name="SalesReason" />
    <RightObject Schema="Sales" Name="SalesReason" />
  </Pair>
</TablePairs>
```


Customizing Tables

The Table Settings step allows you to customize various table settings. You can change all or some of the following settings:

- The key that should be used to compare the data between tables. This is useful when a table has no primary key or other unique indexes in SQL Server.
- A where clause that should be used to filter the table data.
- Columns that should be excluded from the comparison.



The following fragment is the xml generated when some tables settings are changed:

```
<TablePairs>
  <!--**** ***** ****-->
  <!--**** custom tables ****-->
  <!--**** HumanResources.Employee ****-->
  <Pair Action="Include">
    <LeftObject Schema="HumanResources" Name="Employee">
      <Key Name="410c760f-c11a-47b6-a400-924eb70f7f2a">
        <Column>EmployeeID</Column>
      </Key>
    </LeftObject>
    <RightObject Schema="HumanResources" Name="Employee">
      <Key Name="605242e3-ccfa-449e-ad12-5b9c59df0780">
        <Column>EmployeeID</Column>
      </Key>
    </RightObject>
  </Pair>
</TablePairs>
```

```
</RightObject>
<Columns Action="Exclude">
  <ColumnPair>
    <LeftColumn>rowguid</LeftColumn>
    <RightColumn>rowguid</RightColumn>
  </ColumnPair>
  <ColumnPair>
    <LeftColumn>ModifiedDate</LeftColumn>
    <RightColumn>ModifiedDate</RightColumn>
  </ColumnPair>
</Columns>
</Pair>
<!--**** *****-->
</TablePairs>
```

View Mapping Rules

The View Mapping Rules step allows you to define how data compare should perform the name-matching of the database views. By default, views with the exact name, on both schemas, are paired with one another and compared. You can use mapping rules if your schemas have views that do not have the exact name, but are different in a consistent way. If there are views in the development environment, whose name starts with V_DEV_, but in production the same views start with V_PROD_, then you can use the following mapping rule to map them. Without the mapping rule, these views will be excluded.

i The wizard steps related to views are disabled by default. To enable them, check the comparison option Compare and synchronize views.

The screenshot shows the 'View Mapping Rules' step in the Idera Data Compare Command Line Wizard. The window title is 'Idera Data Compare Command Line Wizard'. The main heading is '(Optional) Define the mapping rules between database views.' The left sidebar contains a list of steps: Get Started, Source Database, Target Database, Pairing Type, Comparison Options, Scripting Options, Table Mapping Rules, Excluded Tables, Tables Settings, View Mapping Rules (highlighted with a right-pointing arrow), Excluded Views, Views Settings, Log Files & Options, and Save the Config File. The main area is titled 'View Mapping Rules' and has a 'Restore Defaults' link. It contains three tabs: 'Schema Mapping Rules', 'Name Mapping Rules' (selected), and 'Data Type Mapping Rules'. Under 'Name Mapping Rules', there are three radio button options:

- Match the exact name
- Ignore the name prefix, such as the tmp in the tmp_Employee. This option has two text input fields: 'Source view prefix' with the value 'V_DEV' and 'Target view prefix' with the value 'V_PROD'.
- Ignore the name postfix, such as the tmp in the Employee_tmp. This option has two empty text input fields: 'Source view postfix' and 'Target view postfix'.

 At the bottom of the window are three buttons: 'Previous', 'Next', and 'Exit'.

The following fragment is the xml generated for this mapping rule:

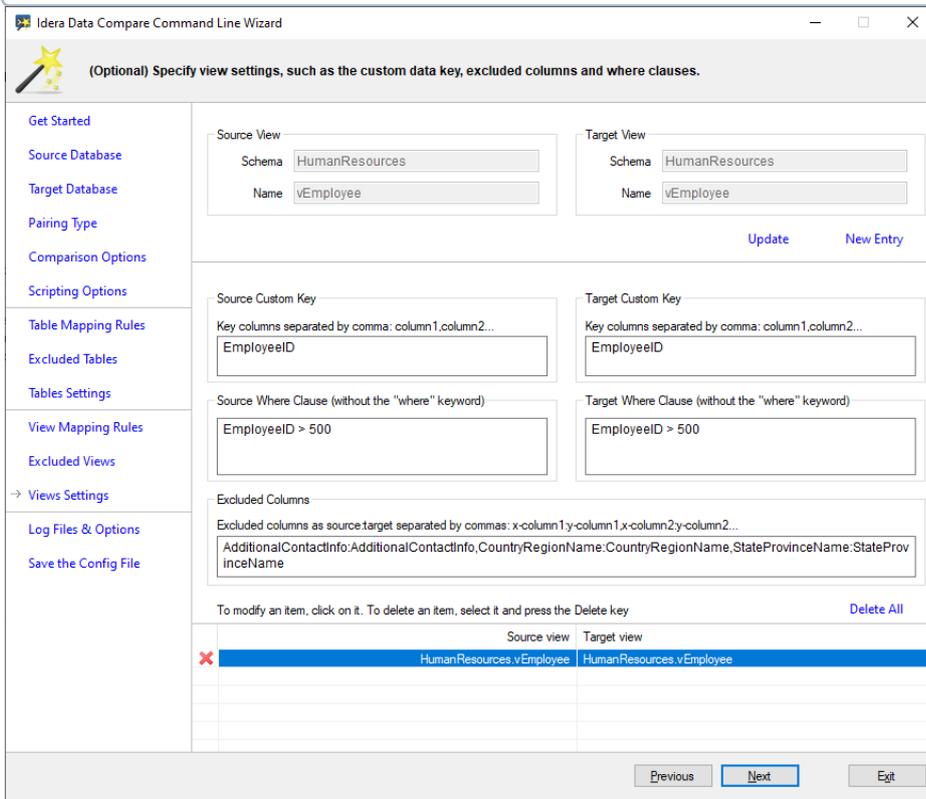
```
<ViewsMappingRules>
  <NameMapping MappingRule="IgnorePrefix" IgnoreLeftName="V_DEV"
  IgnoreRightName="V_PROD" />
</ViewsMappingRules>
```


Customizing Views

The View Settings step allows you to customize various view settings. You can change all or some of the following steps:

- The key that should be used to compare the view data.
- A where clause that should be used to filter the view data.
- The view columns that should be excluded from the comparison.

i The wizard steps related to views are disabled by default. To enable them, check the comparison option Compare and synchronize views.



The following fragment is the xml generated when various view settings are changed:

```
<ViewPairs>
  <!--**** ***** ****-->
  <!--**** custom views ****-->
  <!--**** HumanResources.vEmployee ****-->
  <Pair Action="Include">
    <LeftObject Schema="HumanResources" Name="vEmployee">
      <Key Name="c93f41fd-27c8-43fe-9ec9-ee2f5bb4eb1a">
        <Column>EmployeeID</Column>
      </Key>
    </LeftObject>
    <RightObject Schema="HumanResources" Name="vEmployee">
```

```

    <Key Name="7da3f3cf-762c-46ea-8e85-7a8027a83764">
      <Column>EmployeeID</Column>
    </Key>
  </RightObject>
  <Columns Action="Exclude">
    <ColumnPair>
      <LeftColumn>AdditionalContactInfo</LeftColumn>
      <RightColumn>AdditionalContactInfo</RightColumn>
    </ColumnPair>
    <ColumnPair>
      <LeftColumn>CountryRegionName</LeftColumn>
      <RightColumn>CountryRegionName</RightColumn>
    </ColumnPair>
    <ColumnPair>
      <LeftColumn>StateProvinceName</LeftColumn>
      <RightColumn>StateProvinceName</RightColumn>
    </ColumnPair>
  </Columns>
  <WhereClause>EmployeeID &gt; 500</WhereClause>
</Pair>
<!--**** *****-->
</ViewPairs>

```

Log Files and Options

The Log Files and Options step contains the command line log files and a few options:

- The top section of the page contains the source and the target database. You can flip them by clicking on the Flip link. Target database is the database on which the data synchronization will be executed.
- **Synchronize the target database.** Indicates whether the synchronization script should be executed in the target database. If unchecked, the script is generated, optionally saved to a file, but not executed.
- **Encrypt SQL Server instance, credentials and database name.** Encrypts these settings in the xml config file.
- **Log operational messages to.** Logs the operational messages, such as the data compare progress, to the specified file.
- **Save the error log to a file.** Saves the data compare errors to the specified file.
- **Create a Windows error log.** Creates an Windows Event Log for the data comparison errors, instead of logging them to a file.
- **Save sync script to.** Saves the data synchronization script to the specified file.
- **Save sync log to.** Saves the data synchronization log to the specified file.
- **Use this encoding for all files.** The encoding that should be used for all output files.
- **Override all log files.** Whether the log files should be overridden or appended to.
- **Execute quietly and hide operational messages.** If checked, does not show the operational messages on the console.

The following xml fragment contains the command line settings:

```
<CommandLineOptions>
  <ComparisonLogFile>D:\QA\Idera\SqlData\Cmd\Logs\op.txt</
ComparisonLogFile>
  <ErrorLogName>D:\QA\Idera\SqlData\Cmd\Logs\err.txt</ErrorLogName>
  <DataScriptFile>D:\QA\Idera\SqlData\Cmd\Logs\sync_script.sql</
DataScriptFile>
  <DataLogFile>D:\QA\Idera\SqlData\Cmd\Logs\sync_log.txt</DataLogFile>
  <FileEncoding>UTF8</FileEncoding>
  <OverwriteComparisonLog>true</OverwriteComparisonLog>
  <TargetDatabase>RightDatabase</TargetDatabase>
  <!--**** the synchronization script will be executed against the
target database ****-->
  <Synchronize>true</Synchronize>
  <EncryptConnectionSettings>>false</EncryptConnectionSettings>
</CommandLineOptions>
```

Return Codes

The command line returns 0 when it finishes successfully. A non-zero code indicates an error.

Comparison Toolset PDF

This page contains a direct link to the IDERA Comparison Toolset help in PDF format. This format is suitable for printing and saving on your local machine for further reference. The PDF includes all pages from the relevant product help published on wiki.idera.com.

- [IDERA SQL Comparison Toolset 7.1.5.pdf](#)
- [IDERA SQL Comparison Toolset 7.1.0.pdf](#)