**Introduction to SQL Server**

Microsoft SQL Server is a Relational Database Management System (RDBMS) developed by Microsoft. It is a highly scalable product that can be run on anything from a single laptop, to a network of high-powered cloud servers, and anything in between.

Of course, by "anything", it still needs to satisfy the usual hardware and software requirements, but these requirements are reasonably modest, considering what SQL Server is capable of.

According to Microsoft, SQL Server is the number 1 most-used database in the world. According to various other sources, it's more like number 2, 3, or 4. In any case, it's one of the most popular database management systems in the world.

Since the release of SQL Server 1.0 in 1989, SQL Server has evolved to become a true, enterprise information platform. While it's core function is that of an RDBMS, SQL Server has become much more than that. SQL Server 2016 includes built-in business intelligence tools, as well as a range of analysis and reporting tools. This is on top of the database management tools such as database creation, backup, replication, security, and more.

This tutorial focuses more on the database management tools.

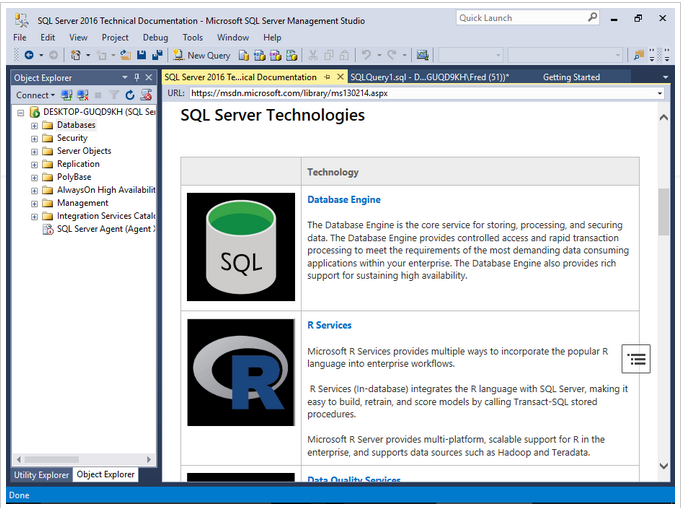
## Database Management Tools

SQL Server comes with a number of tools to help you with your database administration and programming tasks.

Some typical database administration and programming tasks could include:

* Create & maintain databases
* Create & maintain tables
* Create & maintain other database objects such as stored procedures, views, etc
* Create & maintain and schedule data backups
* Replication (eg, create a copy of the database)
* Create & maintain users, roles, etc
* Optimization tasks

These are some of many tasks that a database administrator (DBA) might need to perform. SQL Server provides the means for performing these tasks.



## Client/Server Database Systems

SQL Server is a client/server DBMS, as opposed to a desktop system such as [Access](https://www.quackit.com/microsoft_access/).

Client/server systems are designed to run on a central server - or servers - so that multiple users can access the same data simultaneously from across a network. Users normally access the database through an application.

For example, a web-based corporate CRM could be used by employees in various cities, or even countries, all reading and updating data via their browser.

Generally, it's the CRM application that provides the functionality to these users (for example, the CRM could be built using say, [ColdFusion](https://www.quackit.com/coldfusion/), [HTML](https://www.quackit.com/html/), and [JavaScript](https://www.quackit.com/javascript/)). It is the database that stores the data and makes it available. However, SQL Server does include some useful features that can assist the application in providing its functionality.

Other client/server databases include [MySQL](https://www.quackit.com/mysql/), Oracle, and PostgreSQL, to name a few.

## SQL Server 2016 Enhancements

SQL Server 2016 includes a number of enhancements from its predecessor, SQL Server 2014. It would probably take a whole new tutorial to list them in detail, but in a nutshell, the improvements in SQL Server 2016 over 2014 include:

Better security

With Always Encrypted, Row-Level Security, and Dynamic data masking

Higher availability

Including AlwaysOn Availability Groups, Cloud Witness, Storage Spaces Direct, Workgroup clusters

Improved database engine

TempDB enhancements, Query Store, Stretch Database,

More analytics

Tabular enhancements, R integration)

Various improvements to reporting

Search, custom branding, optimization for modern browsers, mobile, etc

## SQL Server 2016 Editions

SQL Server 2016 comes in five editions:

* Enterprise
* Standard
* Developer
* Web
* Express

# Install SQL Server 2016

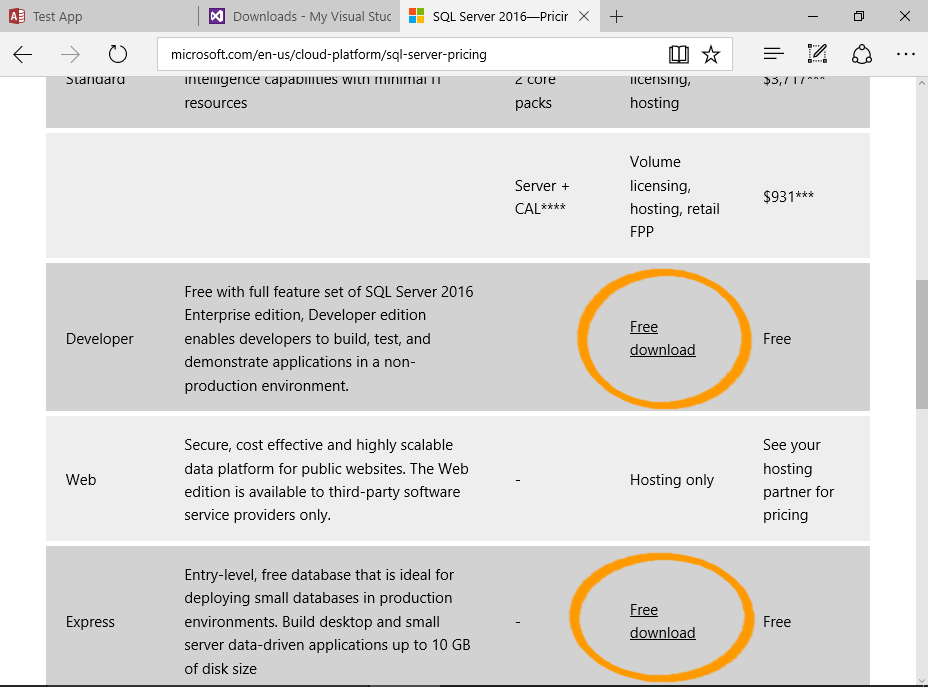
SQL Server 2016 can be installed with the SQL Server Installation Wizard, which allows you to install all SQL Server components in one go.

If you don't have a copy of SQL Server 2016, you can download the installation files from the [Microsoft website](https://www.microsoft.com/en/server-cloud/products/sql-server/Purchasing.aspx). The Express edition is a free download, as is the Developer edition.

The examples in this tutorial are all done using the free Developer edition. You'll need to become a Visual Studio Dev Essentials member first (if you're not already a member). If you don't want to become a Visual Studio Dev Essentials member, you can download the free SQL Server 2016 Express edition without that requirement.

Here's a high level overview of the steps required to download and install SQL Server 2016.

### Download SQL Server

[](https://www.microsoft.com/en/server-cloud/products/sql-server/Purchasing.aspx)

<https://www.microsoft.com/en-ca/sql-server/sql-server-editions-express>

Microsoft.com/en-us/cloud-platform/sql-server-pricing

Download SQL Server from the [Microsoft website](https://www.microsoft.com/en/server-cloud/products/sql-server/Purchasing.aspx).

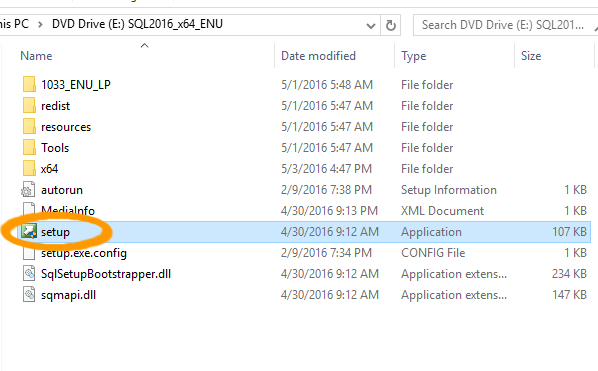
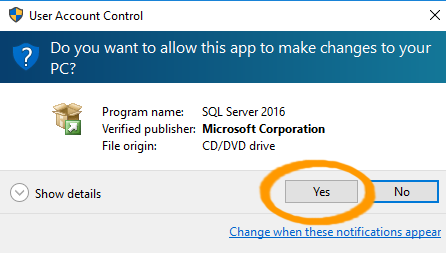
For a free download, choose either the Express or Developer edition.

Choosing the Developer edition will direct you to the Visual Studio Dev Essentials website. You will need to log in or become a member in order to download SQL Server Developer edition.

Membership is free, and you get access to other free downloads too.

If you don't want to become a Visual Studio Dev Essentials member, download the free SQL Server 2016 Express edition.

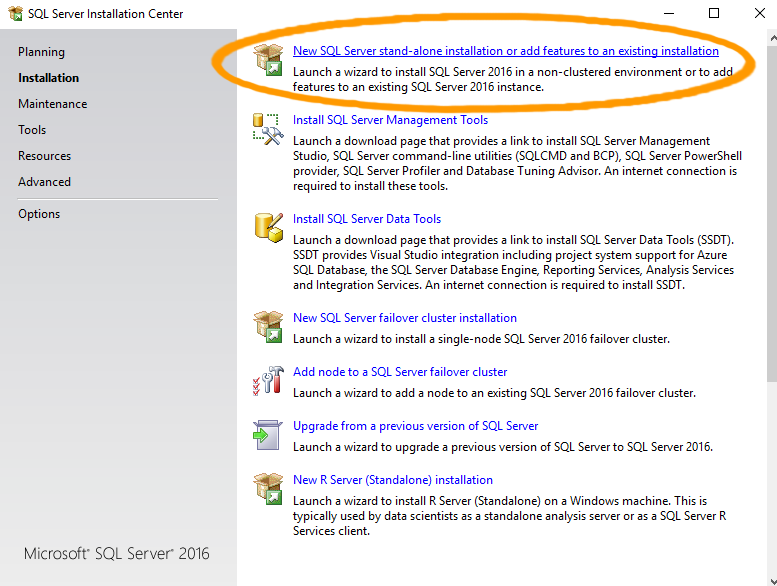
### Launch the SQL Server Installation Center

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/installation/install_sql_server_2016_1a.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/installation/install_sql_server_2016_1b.png)

Once the package has downloaded, open the folder in the File Explorer and double-click Setup to launch the SQL Server Installation Center.

Click Yes if you get a security message asking if you want the app to make changes to your PC.

### Start the Installation

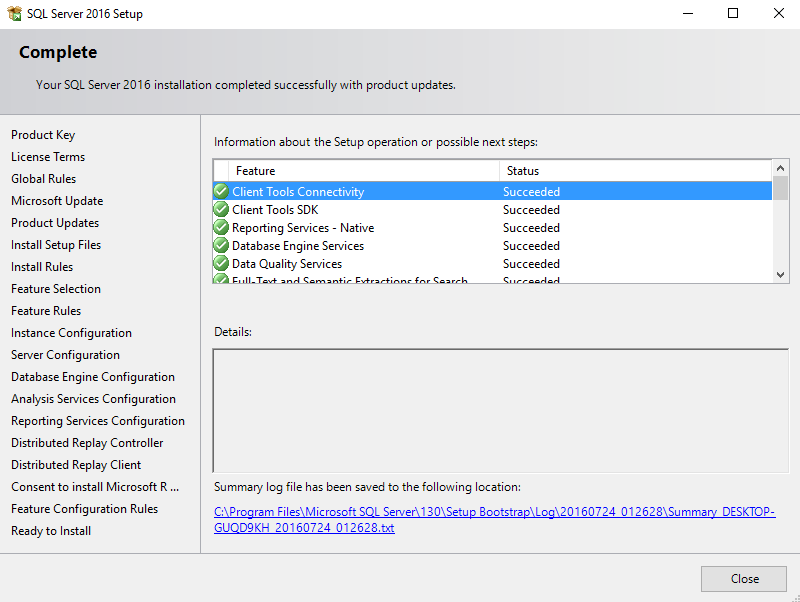
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/installation/install_sql_server_2016_2.png)

The SQL Server Installation Center will now open.

Ensuring that Installation is selected in the left menu, click the first option, New SQL Server stand-alone installation or add features to an existing installation to start the installation process.

This will open an installation wizard that guides you through the process.

### Complete the Installation

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/installation/install_sql_server_2016_19.png)

Continue through [all steps of the installation wizard](https://www.quackit.com/sql_server/sql_server_2016/tutorial/sql_server_2016_installation_with_screenshots.cfm).

Once the installation is complete, a screen will show that it has completed successfully.

Click Close.

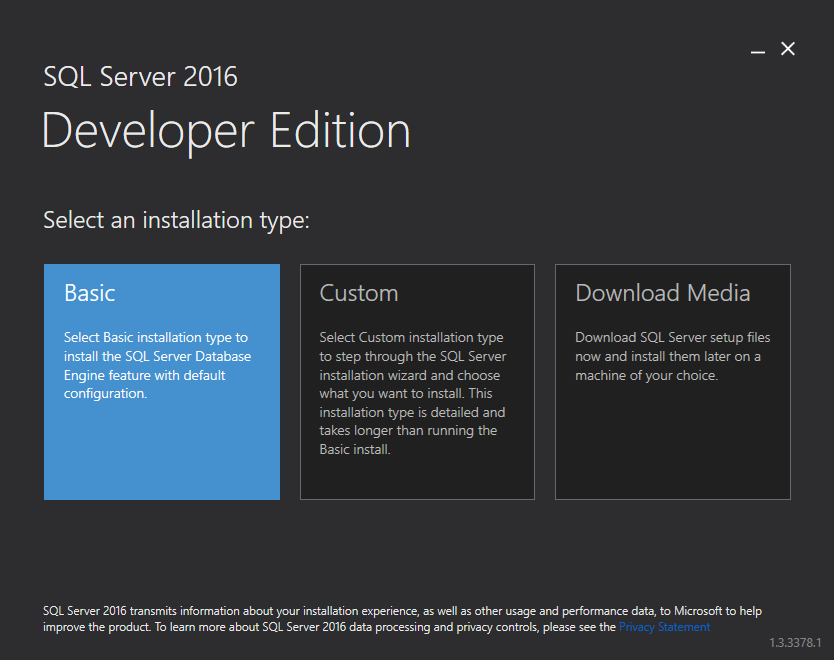
### More Screenshots

Check out [SQL Server 2016 Installation with Screenshots](https://www.quackit.com/sql_server/sql_server_2016/tutorial/sql_server_2016_installation_with_screenshots.cfm) for a step-by-step guide through the installation wizard, complete with screenshots.

### New Installer

There's a new installer that makes installation much easier. Using this installer, a basic installation can be done in 3 clicks (although a customized install will require more steps).

If you see this option on the download page, feel free to use it instead of the steps above.

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/installation/install_sql_server_2016_installer_option_1.png)

# SQL Server Management Studio (SSMS)

SQL Server Management Studio is the main administration console for SQL Server. It allows you to create and administer databases, security, replication, backups, and more.

There are many ways to connect to SQL Server. The method you use will depend on your environment, and where you're connecting from. For example, you can connect via a web application, an online admin tool, via a command line utility, or via SQL Server Management Studio.

SQL Server Management Studio (SSMS) is a graphical admin tool that allows you to perform all sorts of tasks with SQL Server. You can create and manage databases, create user accounts, configure advanced security options, schedule regular backups, and more.

SSMS also provides Transact-SQL, MDX, DMX, and XML language editors for editing and debugging scripts.

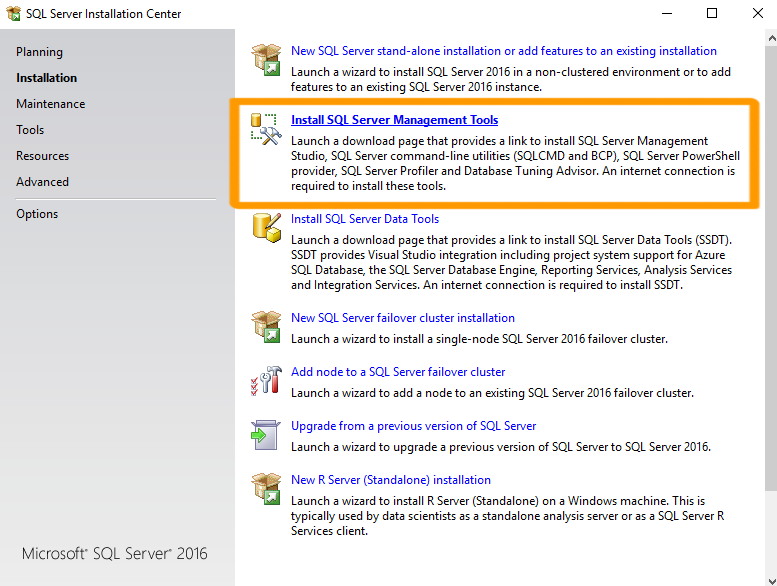
## Getting Started with SQL Server Management Studio

Despite the many steps involved in installing SQL Server, none of those steps involved installing SQL Server Management Studio.

In earlier versions of SQL Server, you could select SQL Server Management Studio (SSMS) as an option in the installation wizard, but this is not the case with SQL Server 2016. So we must install SSMS separately.

Here's how to get started with SSMS.

### Install SSMS

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/sql_server_management_studio_2.png)

You can install SSMS from the same SQL Server Installation Center that you used to install SQL Server.

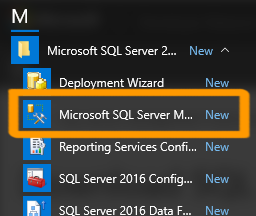
Ensuring that Installation is selected in the left menu, select Install SQL Server Management Tools and follow the prompts.

This actually downloads SSMS before installing it. So it can take some time.

Here are the [screenshots](https://www.quackit.com/sql_server/sql_server_2016/tutorial/install_sql_server_management_studio_with_screenshots.cfm) if you need them.

You can also [download SSMS](https://msdn.microsoft.com/en-us/library/mt238290.aspx) from the Microsoft website if you need to.

### Open SSMS

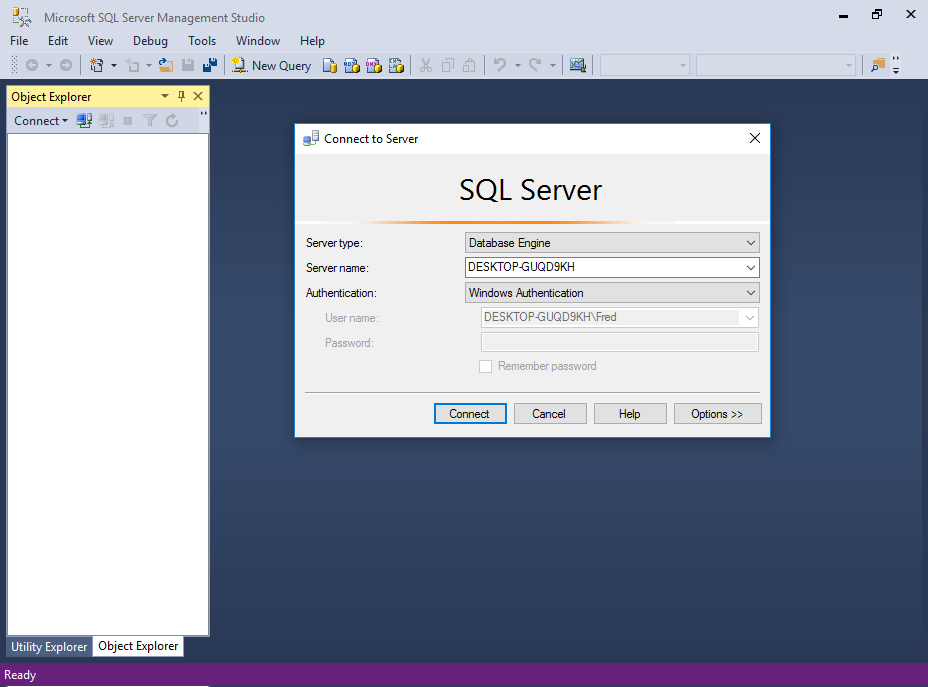
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/sql_server_management_studio_3.png)

You can now open SQL Server Management Studio by finding it in your Windows apps list.

It is located under Microsoft SQL Server 2016.

Depending on your operating system, you could also type SSMS into the Start Page and it should pop up.

### Connect to SQL Server

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/sql_server_management_studio_4.png)

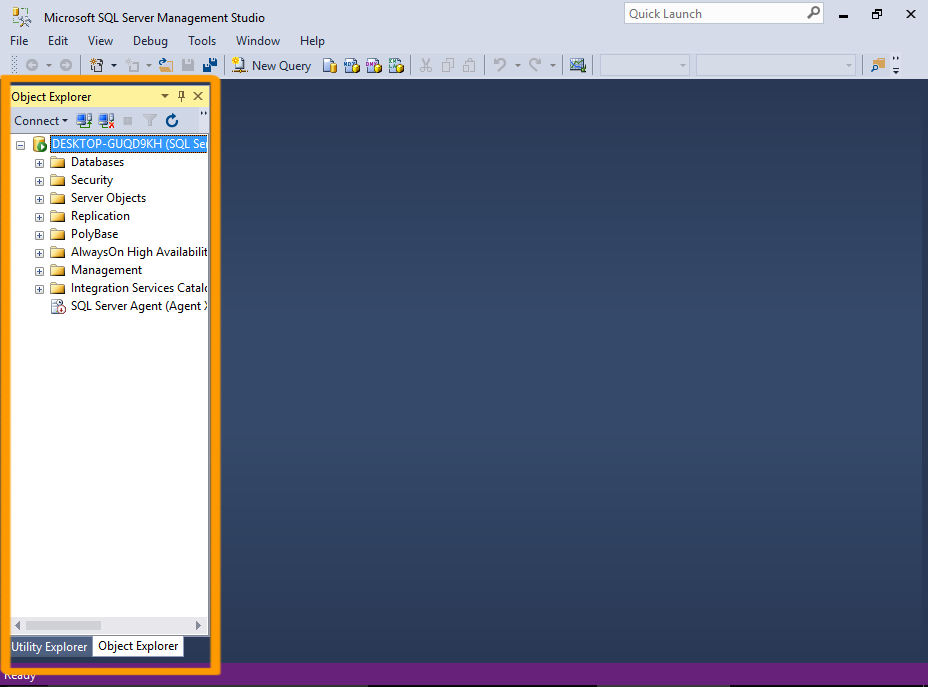
SSMS will prompt you to connect to SQL Server.

Select the server type, server name, and the authentication details and click Connect.

## The SSMS Interface

Here's a quick overview of the SQL Server Management Studio interface.

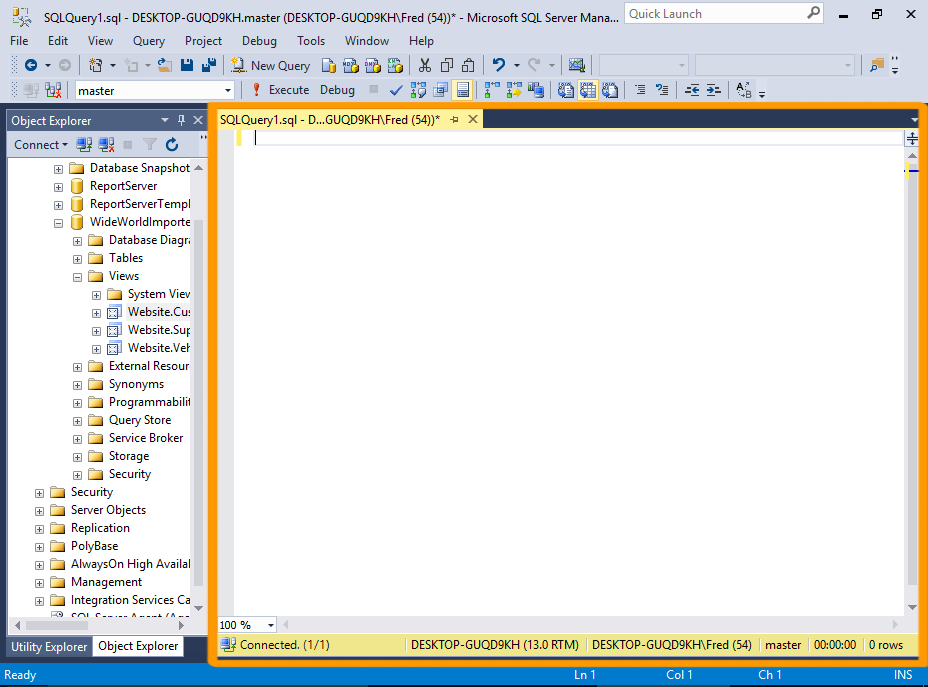
### The Object Explorer

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/sql_server_management_studio_5.png)

Once successfully connected to a server, you will see the server's objects displayed in the Object Explorer on the left.

Object Explorer is a tree view of all the database objects in a server. This can include the databases of the SQL Server Database Engine, Analysis Services, Reporting Services, and Integration Services.

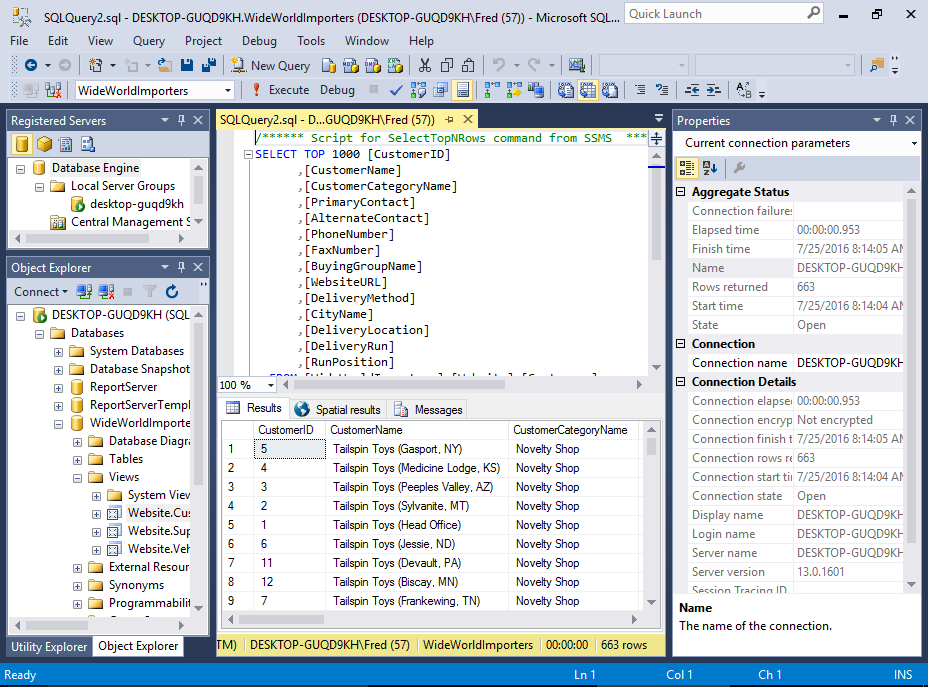
### Document Windows

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/sql_server_management_studio_6.png)

The large area to the right of the Object Explorer is allocated to document windows. This area can contain query editors and browser windows. Its contents depend on the current context.

For example, clicking on New Query in the toolbar will add a new query window, so that you can enter a query. Opening a table in Design view will display the table's fields and their properties. And clicking View Help will open a browser window with SQL Server 2016 Technical Documentation from MSDN.

### Other Windows

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/sql_server_management_studio_7.png)

Various other windows and wizards are available in SSMS.

For example, the Registered Servers window lists the servers that you manage frequently.

The Properties window describes the state of an item in SQL Server Management Studio, such as a connection or a Showplan operator, and information about database objects such as tables, views, and designers.

Many properties are read-only in the Properties window but can be changed elsewhere in the Management Studio.

## Servers and Databases

You can use SQL Server Management Studio to create as many databases as you like. You can also connect to as many databases on as many servers as you like. These all appear in the Object Explorer.

So you could run a query on your development environment, then switch to your test or production environment and run a query there. Because of this, you need to be careful that you don't accidentally run a script against the wrong server.

Most of the tasks performed with SQL Server Management Studio are initiated either from the top menu, or by right-clicking on an icon/object.

# SQL Server 2016: Create a Database

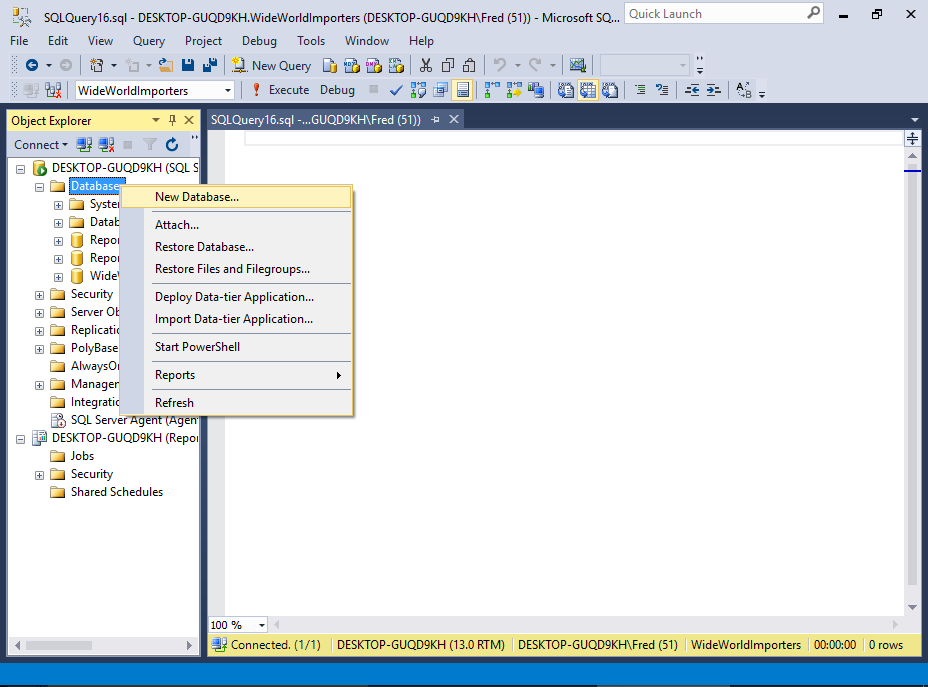
Create a database using the SQL Server Management Studio GUI.

In SQL Server, you can create a new database either via the SQL Server Management Studio graphical user interface (GUI), or by running an SQL script.

## Create a Database using the GUI

Here, we will create a database using the GUI method.

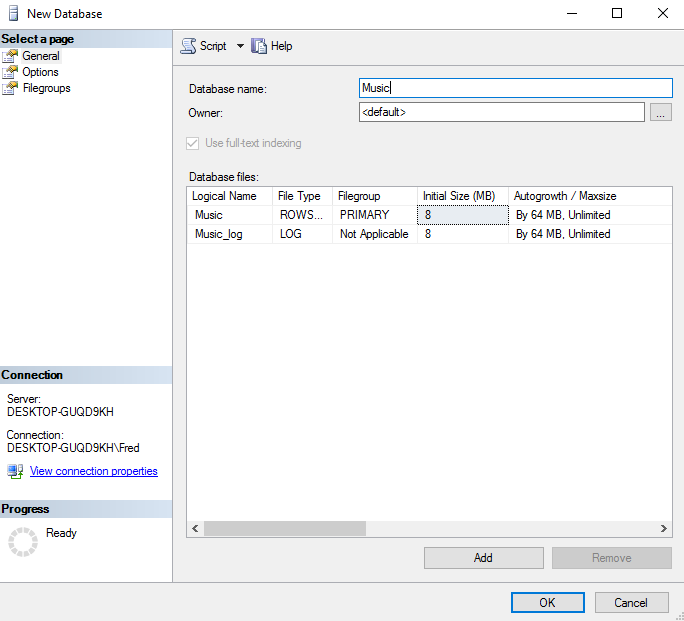
### Create a New Database

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_database_in_sql_server_2016_1.png)

Ensure that the correct server is expanded in the Object Explorer.

Right-click on Databases and select New Database... from the contextual menu.

### Name the Database

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_database_in_sql_server_2016_2.png)

Enter a name for the database and click OK.

For this tutorial, I named mine Music, as we'll be creating a database to store data related to music (i.e. artists, albums, etc).

You can change any of the settings in the New Database dialog before you click OK.

For this tutorial, we'll just leave it at the default settings.

### The New Database

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_database_in_sql_server_2016_3.png)

The new database is now listed in the Object Explorer, under Databases.

If you expand it, you'll see further expandable nodes, each for a specific purpose. For example, one for tables, another for views, etc.

## Create a Database by Script

SQL Server accepts Transact-SQL (which is an extended version of the SQL standard), so you could create the same database by running the following SQL script.

USE master;

GO

CREATE DATABASE Music;

GO

To do this, open a new query by clicking New Query in the toolbar and run an SQL [CREATE DATABASE](https://www.quackit.com/sql/tutorial/sql_create_database.cfm) statement.

If you're not sure how to run a query, we'll cover that soon, when we create a table via script.

Just as you can specify certain properties when creating a database via the GUI, you can include those same properties when creating a database by script. Here's an example of specifying settings for the data and log files.

USE master ;

GO

CREATE DATABASE Music

ON

( NAME = Music\_dat,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Music.mdf',

SIZE = 10,

MAXSIZE = 50,

FILEGROWTH = 5 )

LOG ON

( NAME = Music\_log,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Music\_log.ldf',

SIZE = 5MB,

MAXSIZE = 25MB,

FILEGROWTH = 5MB ) ;

GO

## System Databases

When you create a new database, the database is created based on the model database. This database provides the template for any new database that you create.

The model database is a system database. When you [install SQL Server](https://www.quackit.com/sql_server/sql_server_2016/tutorial/install_sql_server_2016.cfm), the following five system databases are created.

master

This database stores system level information such as user accounts, configuration settings, and info on all other databases.

model

This database is used as a template for all other databases that are created. Any modifications to this database are applied to any new databases created thereafter.

msdb

This database is used by the SQL Server Agent for configuring alerts and scheduled jobs etc

Resource

This is a hidden, read-only database that contains system objects that are included with SQL Server. System objects are physically persisted in the Resource database, but they logically appear in the sys schema of every database.

tempdb

This one holds all temporary tables, temporary stored procedures, and any other temporary storage requirements generated by SQL Server.

Note that you'll only see four databases listed under System Databases in the Object Explorer, because the Resource database is hidden (although it can be accessed programatically).

# SQL Server 2016: Create a Table

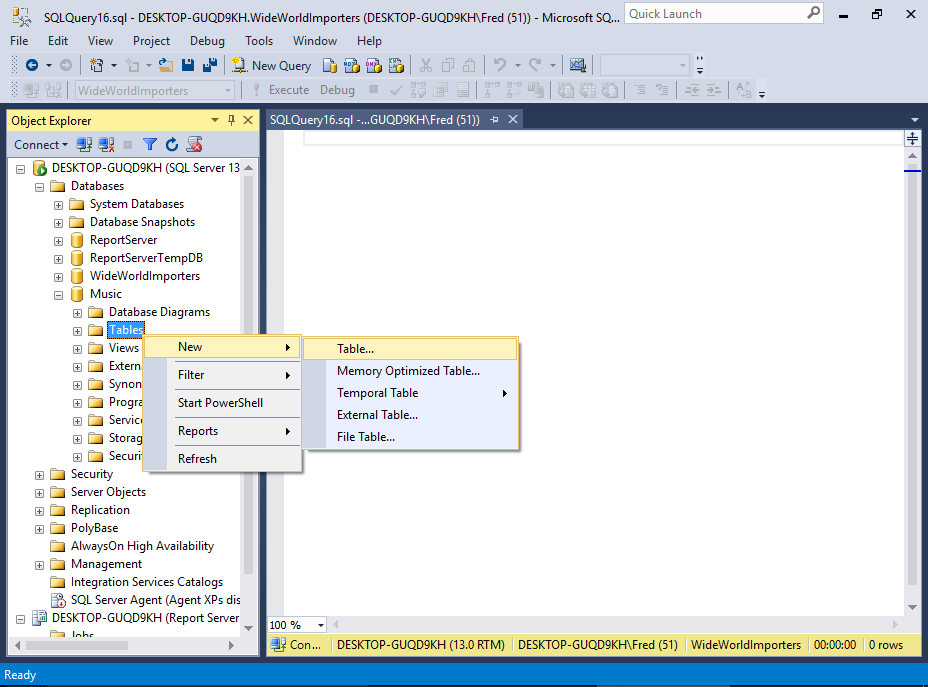
Create a new database table using the SQL Server Management Studio GUI.

In SQL Server, you can create a new database table either via the SQL Server Management Studio graphical user interface (GUI), or by running an SQL script.

## Create a Table using the GUI

Here, we will create a database table using the GUI method.

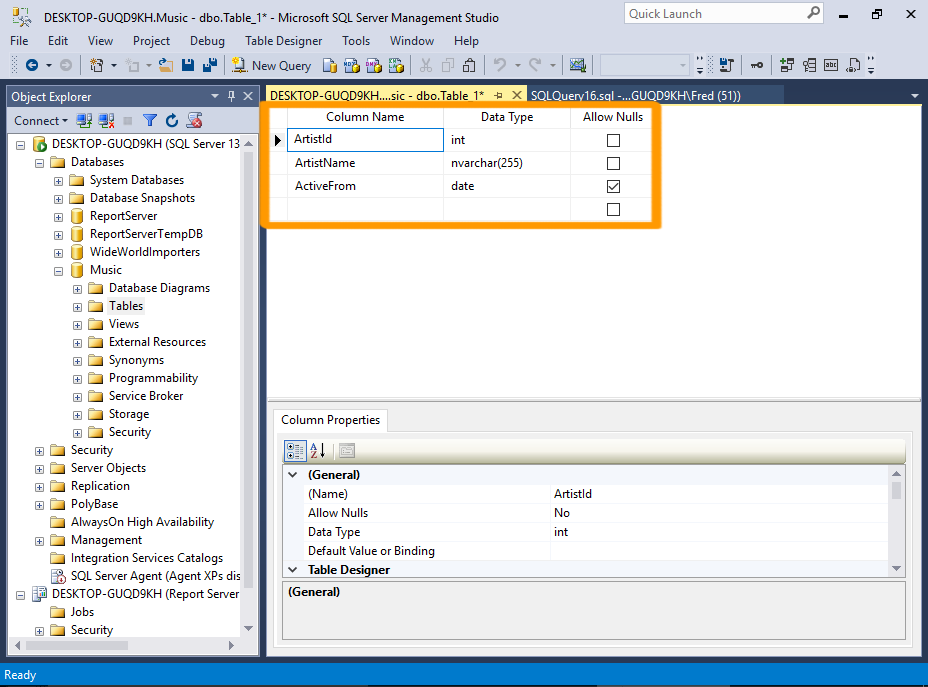
### Create a New Table

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_1.png)

Ensure that the correct database is expanded in the Object Explorer. In our example, we'll expand the Music database.

Right-click on Tables and select New > Table... from the contextual menu.

### Add Columns

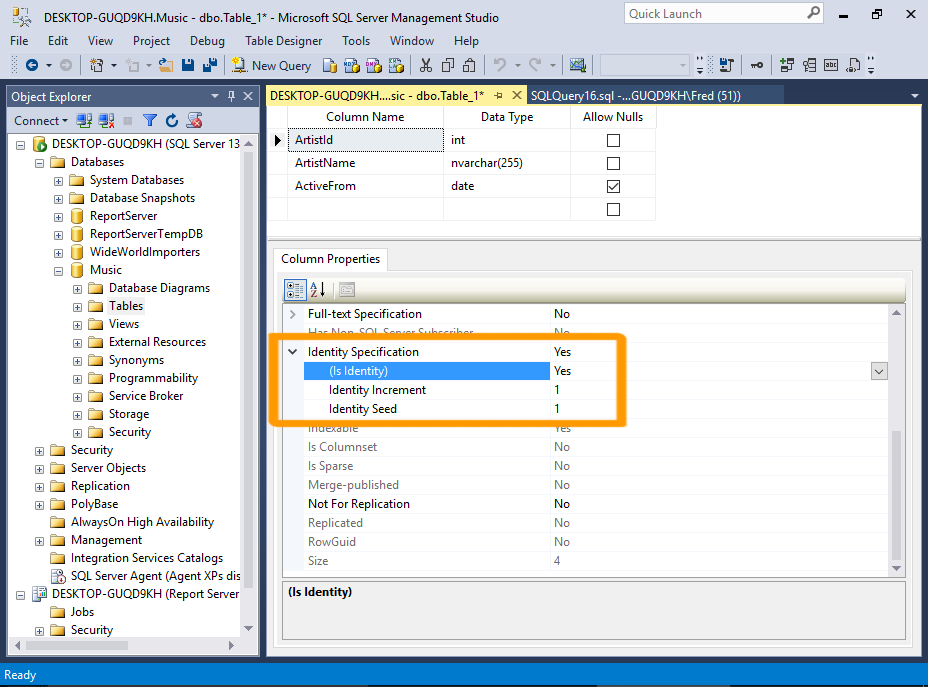
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_2.png)

Add the columns that need to go into the table. Also specify the data type for each column, as well as any other properties that you need to.

For this tutorial, we'll add the following columns/data types:

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Allow Nulls** |
| ArtistId | int | No |
| ArtistName | nvarchar(255) | No |
| ActiveFrom | Date | Yes |

### Set an Identity Column

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_3.png)

We'll now set the ArtistId column to be an identity column. This will make it an auto-incrementing value, so that the user doesn't have to insert this value.

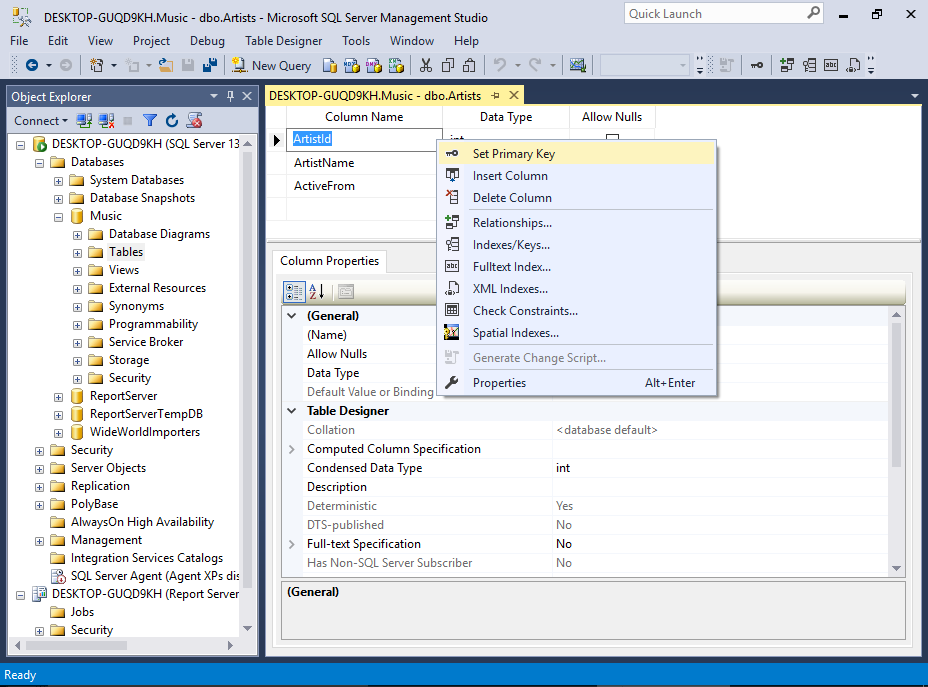
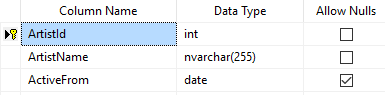
In the top pane, click somewhere in the ArtistId field.

Now, in the bottom pane, scroll down until you see Identity Specification. Expand it, and change (Is Identity) to Yes using the drop-down list.

Leave the other identity properties as they are.

You can increase the height of the bottom Properties pane by clicking and dragging the top edge upwards.

### Set a Primary Key

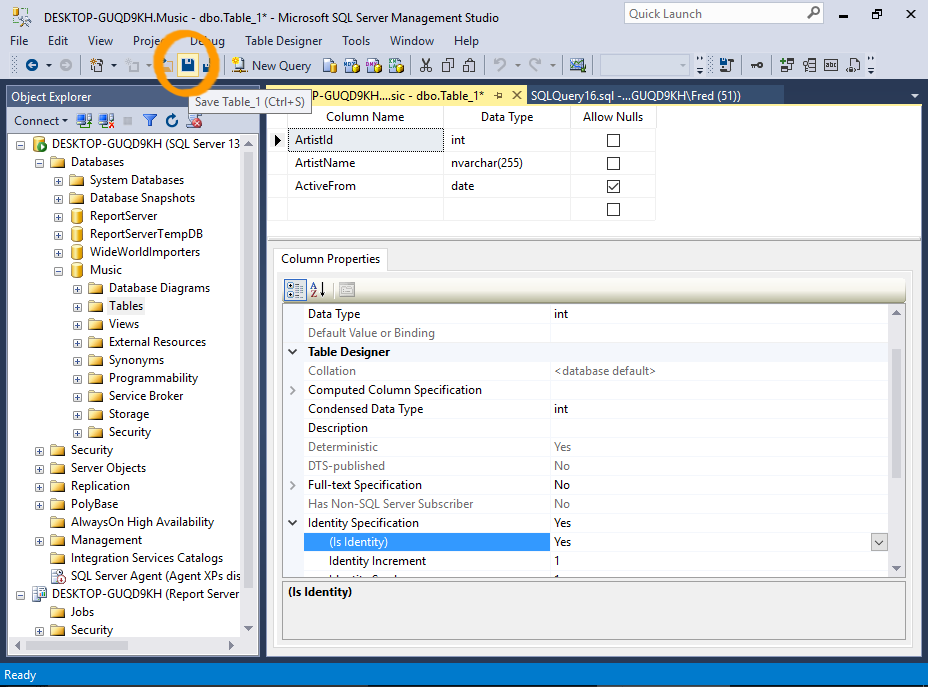
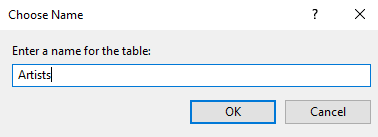
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_4.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_4a.png)

We'll also set the ArtistId column to be the primary key for this table. This will be important for when we set up a relationship between this table and another.

In the top pane, right-click in the ArtistId field and select Primary Key from the contextual menu.

Once done, you will see a little key icon next to the ArtistId column.

### Save the Table

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_5.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_5a.png)

Save the table by clicking on the Save icon in the toolbar.

For this tutorial, we'll call the table Artists.

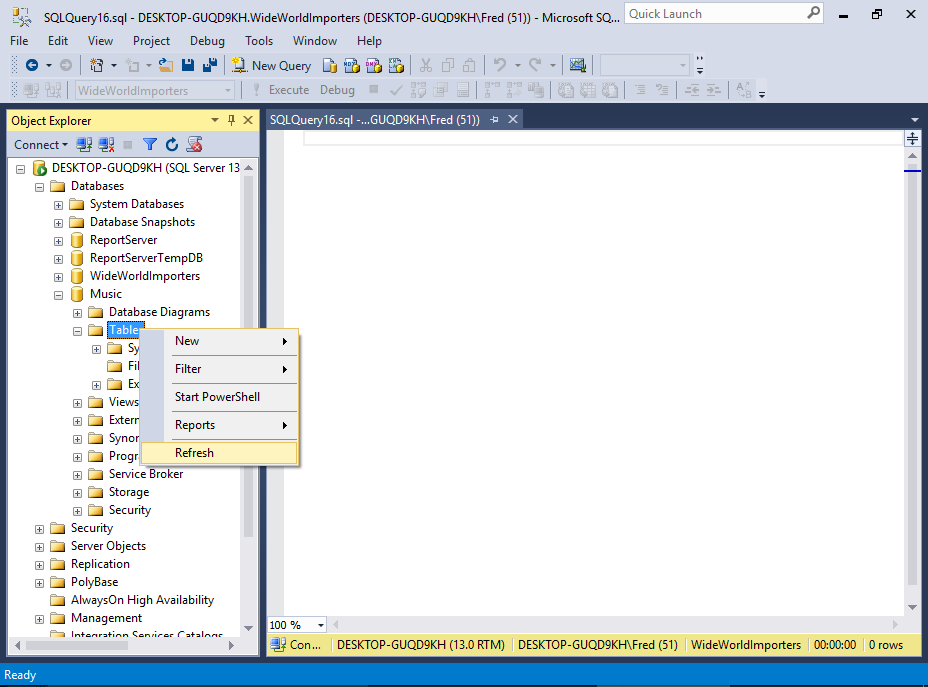
Alternatively, you can save the table by using the File menu, pressing Ctrl+S on your keyboard, or simply clicking on the X to close the table's tab. All of these will prompt you to save the table.

## Check that the Table was Created

When you first open the Tables node in the Object Browser, your initial response might be that the table wasn't even created. Your table is nowhere to be seen!

But relax... all you need to do is refresh the Object Browser.

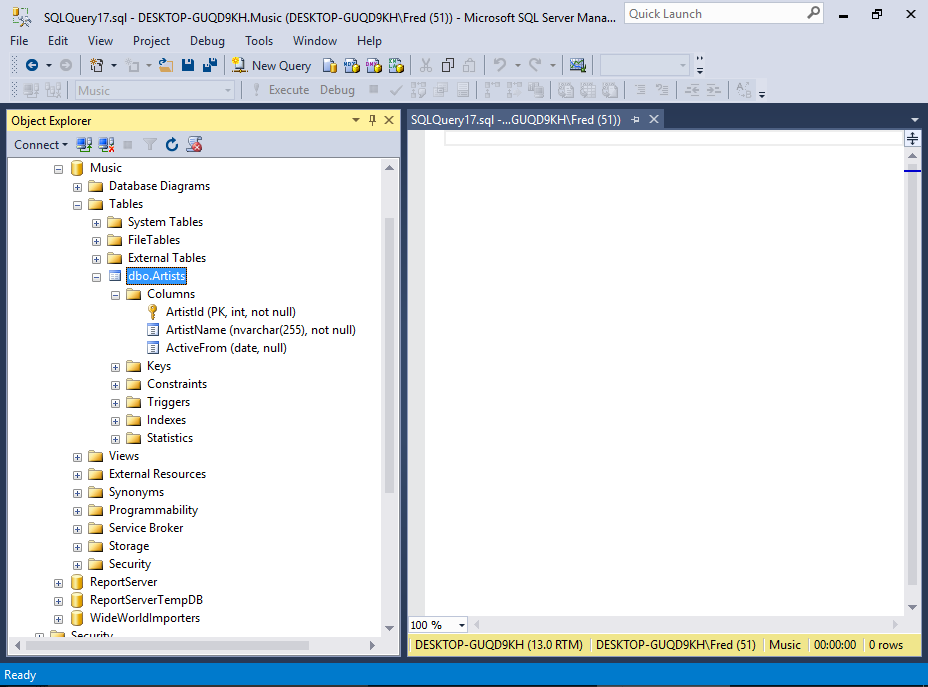
### Refresh the Tables Node

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_6.png)

Ensure that the correct database is open.

Right-click on the Tables node and select Refresh from the contextual menu.

### Your New Table

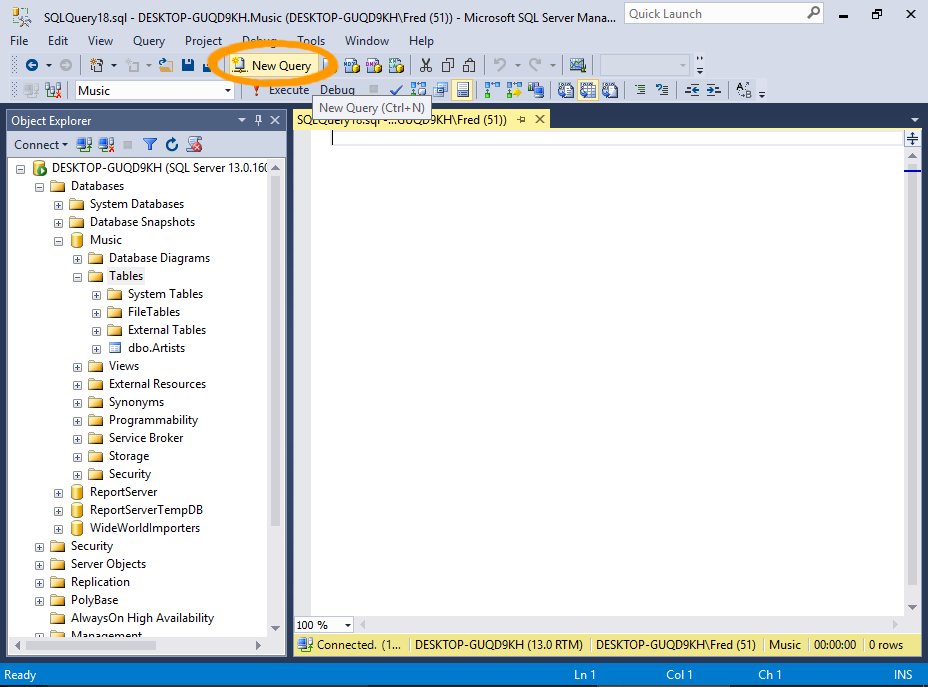
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016_7.png)

# SQL Server 2016: Create a Table from an SQL Script

In SQL Server, you can create tables by running an SQL script with the CREATE TABLE statement. Having just [created a table via the SSMS GUI](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016.cfm), we will now create a table (or two) using an SQL script.

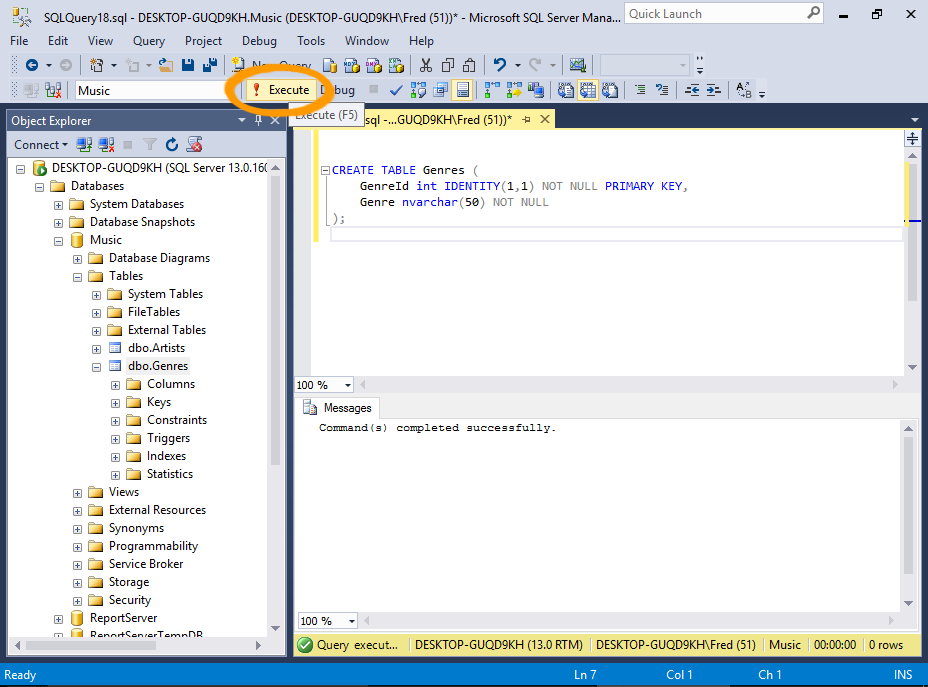
To create a table using SQL, use the CREATE TABLE statement. Provide the column names and their data types, plus any other properties as required.

### Open a New Query Window

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_from_an_sql_script_in_sql_server_2016_1.png)

Open a new query window by clicking on New Query in the toolbar.

### Run the SQL Script

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_from_an_sql_script_in_sql_server_2016_2.png)

Run the SQL script (below) by pasting it into the query window, then clicking the Execute button on the toolbar.

You can also run a query by:

* + Pressing F5 on your keyboard.
  + Clicking Query > Execute from the top menu.
  + Right-clicking in the actual query window and selecting Execute from the contextual menu.

#### The SQL Script

Here's the code to copy and paste:

CREATE TABLE Genres (

GenreId int IDENTITY(1,1) NOT NULL PRIMARY KEY,

Genre nvarchar(50) NOT NULL

);

### Check that the Table was Created

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_table_from_an_sql_script_in_sql_server_2016_3.png)

Let's also use SQL to check that the table was created.

Run SELECT \* FROM sys.tables; to return a list of all tables in the database.

Of course, you can also refresh the Object Explorer by right-clicking on the Tables node and selecting Refresh like we did when we created our previous table.

## Table Options

The CREATE TABLE statement accepts many additional options that allow you to specify the exact properties of the table.

There are too many to go into detail in this tutorial, but they include options for encryption, replication, indexes, and more.

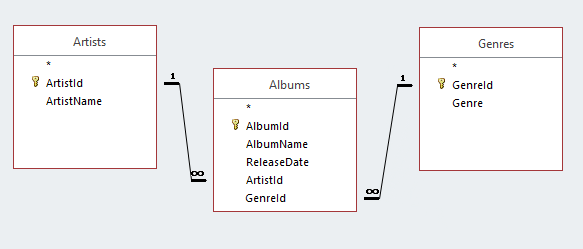
However, we will be covering one of these options next — foreign key constraints — when we [create a relationship](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016.cfm) between tables.

# SQL Server 2016: Create a Relationship

You can create a relationship between tables by using the GUI or SQL script. Here, I demonstrate both methods.

In relational database design, a relationship is where two or more tables are linked together because they contain related data. This enables usersÂ to run queries for related data across multiple tables.

Here, we will create the following relationships.

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_1.png)

## The Method

Here's how we'll do it:

* We'll use **SQL** to create the Albums table and one relationship.
* We'll use the **GUI** to create the other relationship.

That way, you'll get to see both methods of creating a relationship.

We only need to create one table because we've already created two of these tables previously in this tutorial (the Artists table [via the GUI](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_table_in_sql_server_2016.cfm) and the Genres table [using SQL](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_table_from_an_sql_script_in_sql_server_2016.cfm)).

## Create a Relationship using SQL

Open a new query window in SSMS and run the following code:

CREATE TABLE Albums

(

AlbumId int IDENTITY(1,1) NOT NULL PRIMARY KEY,

AlbumName nvarchar(255) NOT NULL,

ReleaseDate date NOT NULL,

ArtistId int NOT NULL,

GenreId int NOT NULL

CONSTRAINT FK\_Albums\_Artists FOREIGN KEY (ArtistId)

REFERENCES dbo.Artists (ArtistId)

ON DELETE NO ACTION

ON UPDATE NO ACTION

);

The first part of that statement creates the table.

The last part defines the relationship. This part:

CONSTRAINT FK\_Albums\_Artists FOREIGN KEY (ArtistId)

REFERENCES dbo.Artists (ArtistId)

ON DELETE NO ACTION

ON UPDATE NO ACTION

The first two lines create the relationship. They create a foreign key constraint between the Albums.ArtistId column and the Artists.ArtistId column.

The last two lines specify what SQL Server should do if someone tries to delete or update a parent record that is being referenced by a record in the child table. In this case, NO ACTION means that the delete/update won't go ahead. The user will just get an error.

You could change this to ON DELETE CASCADE if you want to be able to delete the parent and the child in one go (i.e. the delete will cascade from the parent to the child). The same logic applies to updates, by using ON UPDATE CASADE.

NO ACTION is the default value, so we could've done without those last two lines of code. However, I included it, because it's an important factor to think about when creating foreign key constraints.

### What's a Foreign Key Constraint?

A foreign key constraint defines a relationship between this table and another table. When you create a foreign key constraint, you create it against a specific column in the child table, to reference a specific column in parent table.

This makes the column in the child table a foreign key. The constraint ensures that any value that goes into this (foreign key) column corresponds with a value in the primary key column of the parent table. If someone tries to enter a value that doesn't correspond with a value in the parent table's primary key column, SQL Server will throw an error.

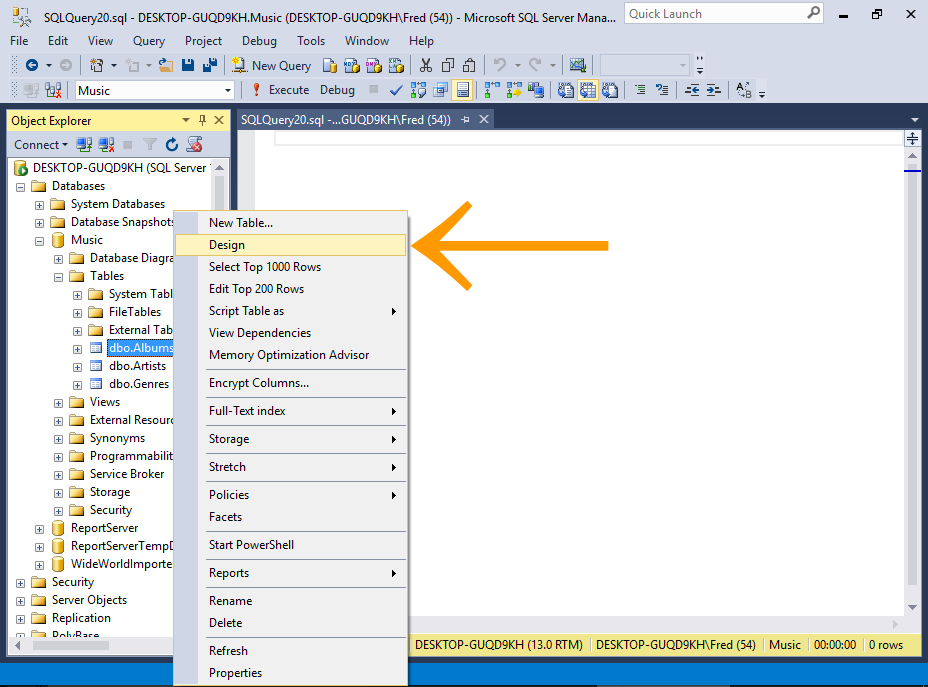
This helps enforce referential integrity. It prevents us from having orphaned records (child records that have no parent). Or in our example, albums that aren't associated with any artist.

## Create a Relationship via the GUI

Now we'll create the other relationship via the SQL Server Mangement Studio's GUI.

It would've been easier to include this in the above script but I wanted to demonstrate both methods of creating a relationship.

### Open the Child Table in the Table Designer

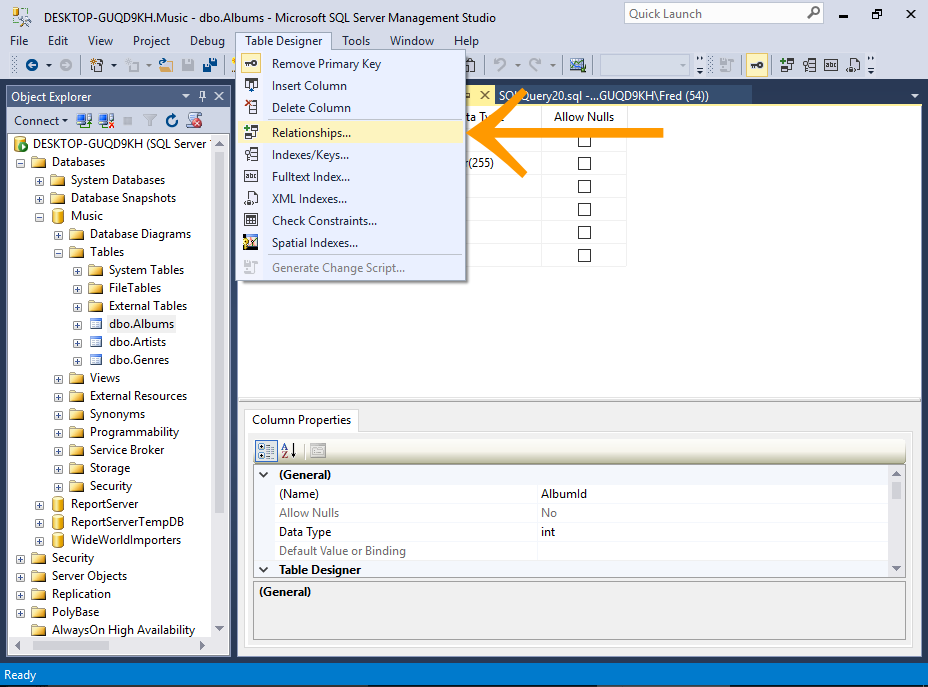
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_2.png)

Right-click on the child table (our newly created Albums table) and select Design from the contextual menu.

If you can't see your newly created table in the Object Browser, you probably need to refresh the Object Browser.

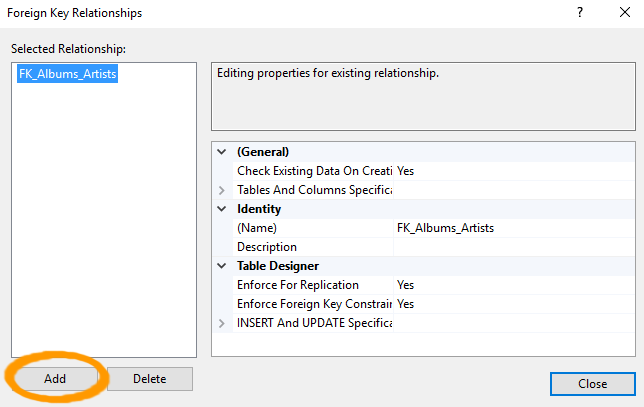
Right-click on the Tables node and select Refresh.

### Open the Foreign Key Relationships Dialog

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_3.png)

Select Table Designer > Relationships... from the top menu.

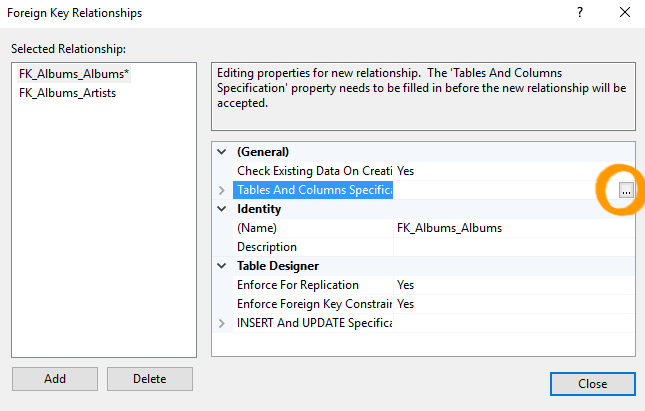
### Add the Relationship

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_4.png)

The Foreign Key Relationships dialog will show you any existing relationships for the table. We can see the relationship that we established just before, when we created the table.

Click Add to add another relationship.

### Select Tables And Columns Specification

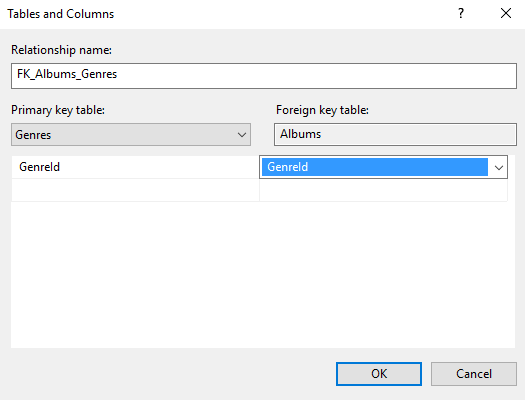
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_5.png)

A new relationship appears above the other one in the Selected Relationship list with a name of FK\_Albums\_Albums.

Ensuring that the the new relationship is selected, click Tables And Columns Specification in the right pane. An ellipses appears to the right of the property.

Click the ellipses (...) to launch the Tables and Columns dialog box.

### The Tables and Columns Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_6.png)

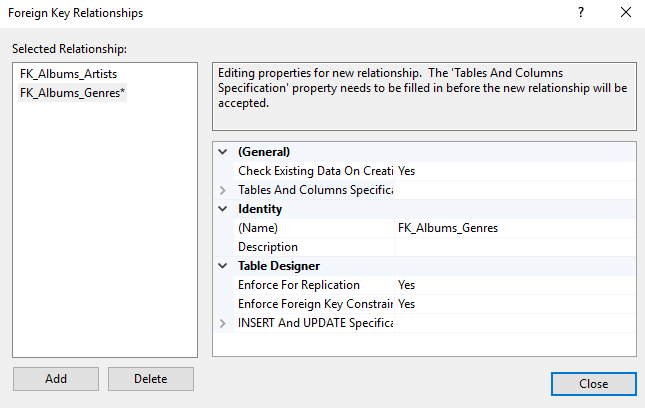
Here, you select the primary key table on the left pane, and the foreign key table on the right.

* + Under Primary key table: select Genres as the table and GenreId as the column.
  + Under Foreign key table: select Albums as the table and GenreId as the column.

Click OK.

SQL Server will suggest a name for the relationship. You can edit this if you wish. Otherwise, leave it as it is.

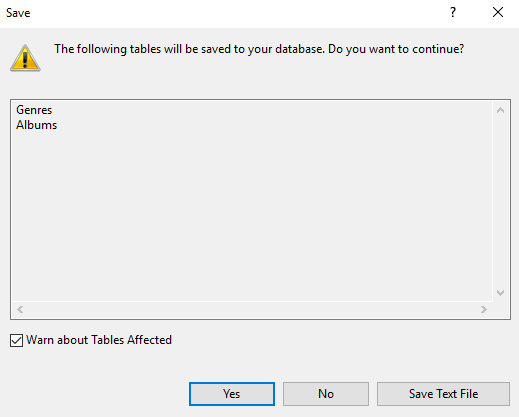
### The Relationship

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_7.png)

Your relationship will now be displayed correctly in the Foreign Key Relationships dialog box.

Click Close.

### Saving The Relationship

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016_8.png)

Your relationship won't be saved until you save the table. When you save the table, you will probably get a warning that two tables will be saved. This is to be expected, as the relationship affects two tables.

Click Yes to save both tables.

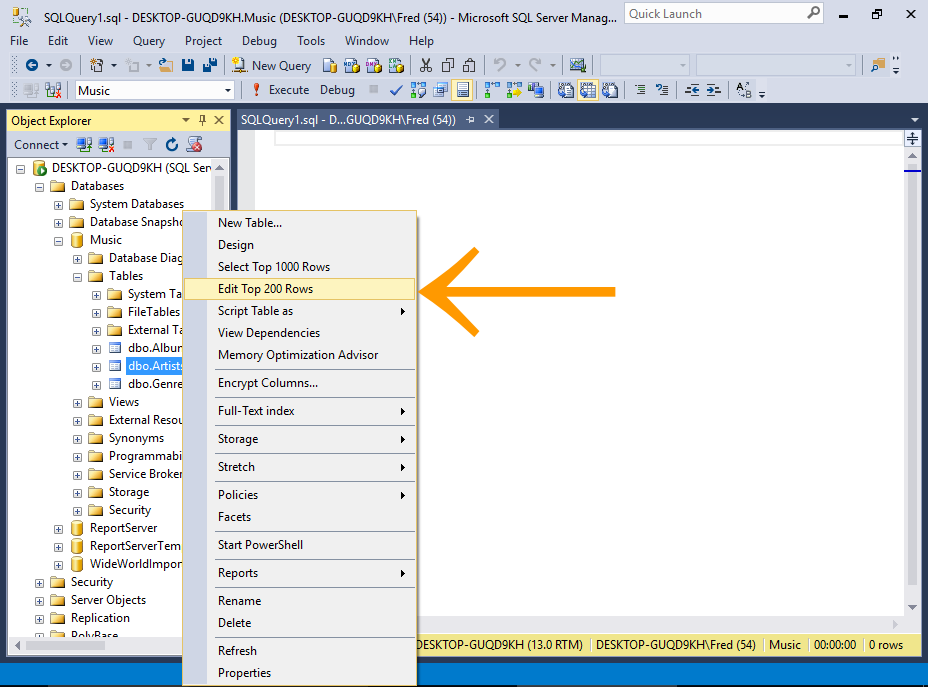
If you select Table Designer > Relationships... for the parent table, you'll also see the relationship there.

# SQL Server 2016: Insert Data

You can insert data using a number of methods. You can insert it directly, copy/paste, import, use SQL, or use an external application.

Here, we'll take a quick look at the various methods of inserting data into SQL Server.

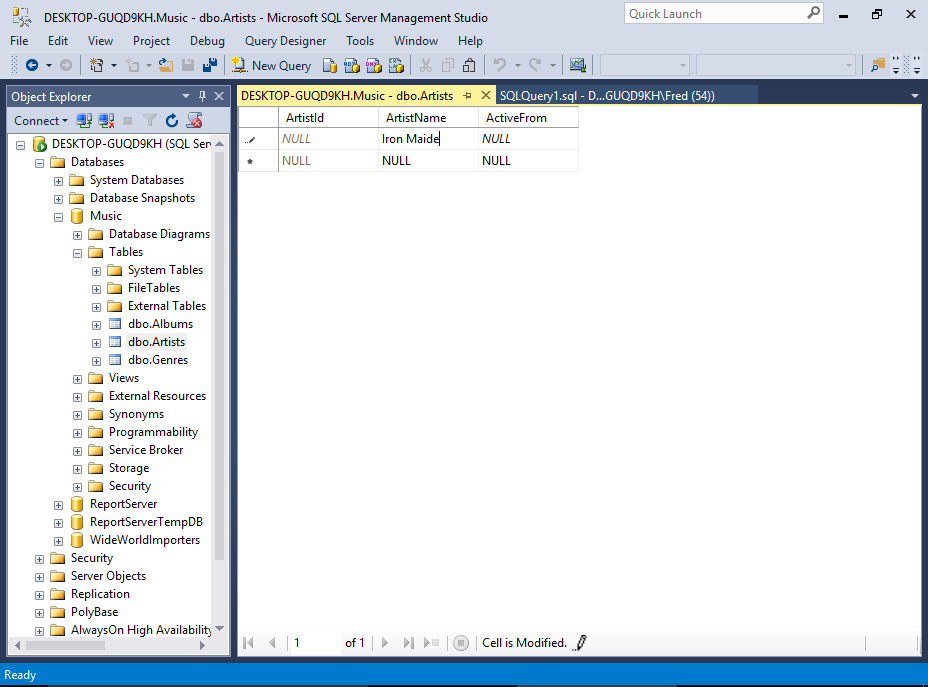
### Direct Input

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_1.png)

Using this method, you enter text directly into the table using the SSMS GUI.

First, using the SSMS GUI, navigate to the table you want to enter data into.

Right-click on the table and select Edit Top 200 Rows.

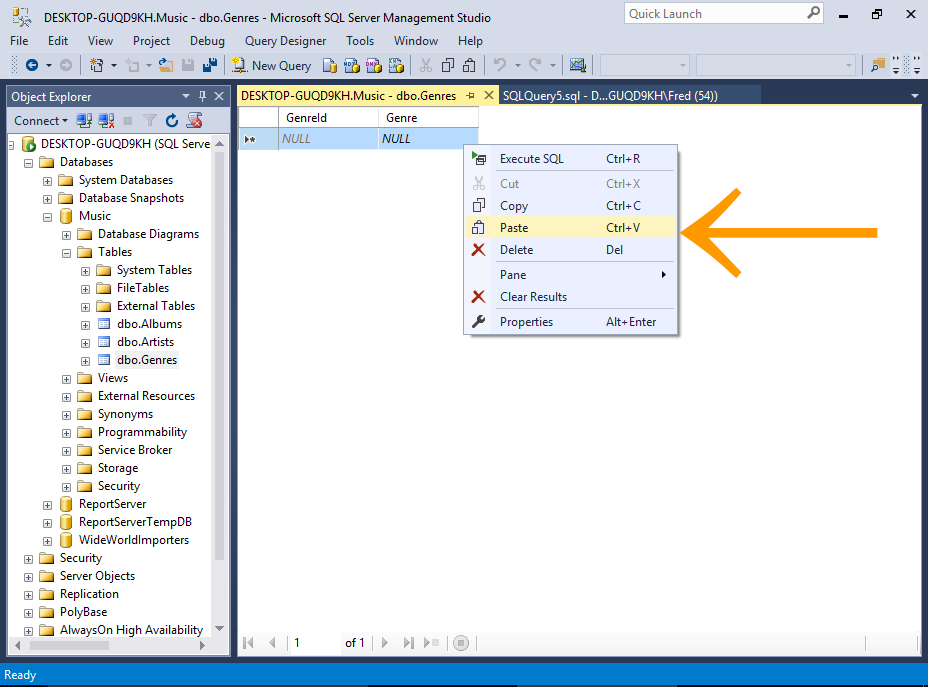
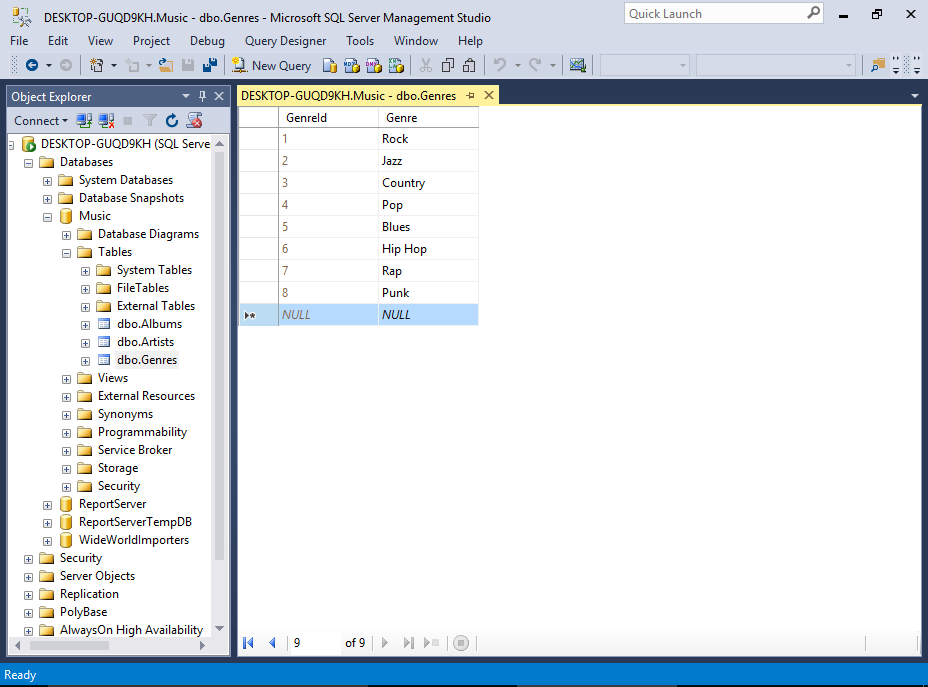
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_1a.png)

The table will open, allowing you to type data directly into the cells.

You won't need to enter data into columns with auto-generated values, such as identity columns, timestamp columns, etc.

Actually, SQL Server will usually prevent you from entering data into such columns.

### Copy/Paste

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_2.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_2a.png)

First, copy the data from the external source.

Now, switch to SSMS and open the table by using the previous method (i.e. right-click on the table and select Edit Top 200 Rows).

Once the table has opened, right-click in the document window and select Paste.

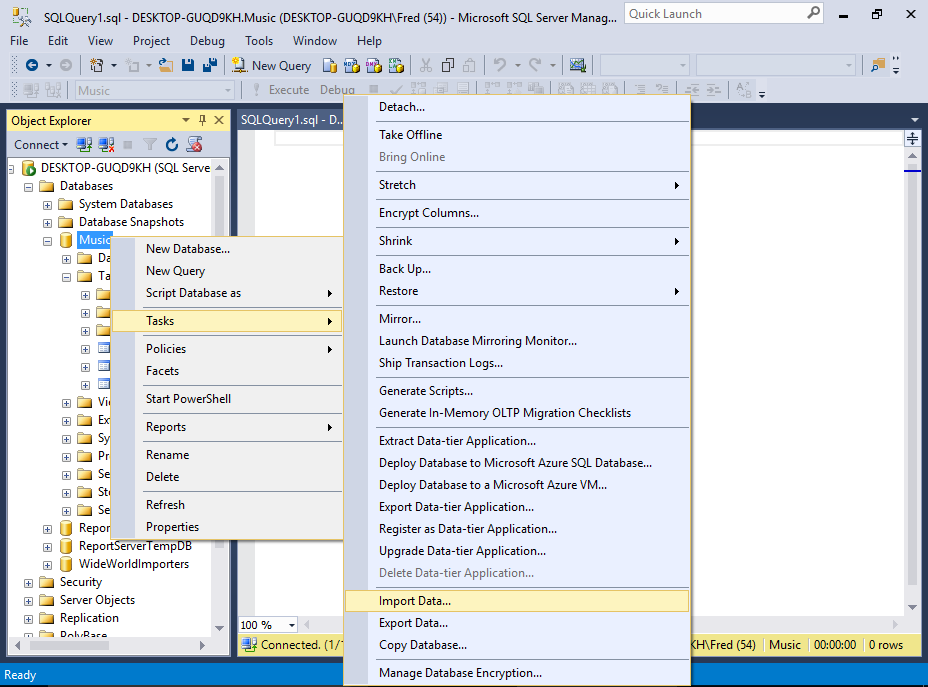
#### Sample Data

Copy and paste the following data into the Genres table:

|  |  |
| --- | --- |
| 1 | Rock |
| 2 | Jazz |
| 3 | Country |
| 4 | Pop |
| 5 | Blues |
| 6 | Hip Hop |
| 7 | Rap |
| 8 | Punk |

SQL Server will ignore the first column, because it's an identity column. So if you already have records in that table, the resulting IDs for the new data will be different to the above.

### Import Data

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_3.png)

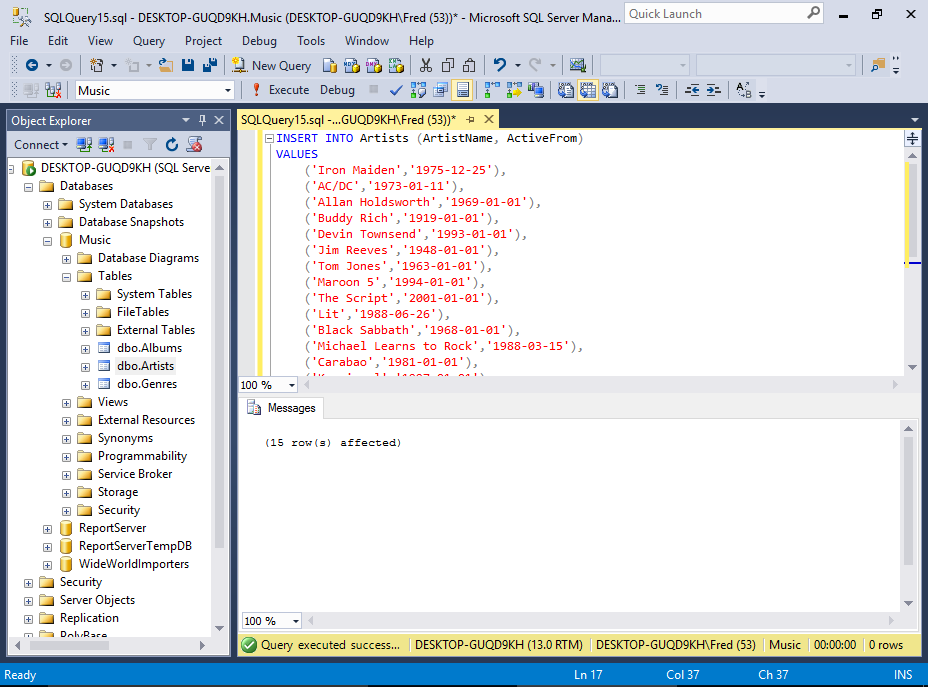
SQL Server Management Studio includes an import wizard that helps you import data into a database.

We'll be covering this method [next](https://www.quackit.com/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016.cfm), but here's a hint of what's to come.

Right-click on the database and select Tasks > Import Data...

More details and screenshots at [Import Data in SQL Server](https://www.quackit.com/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016.cfm) (the next page of this tutorial).

### Use SQL to Insert the Data

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_4.png)

You can use the SQL [INSERT](https://www.quackit.com/sql/tutorial/sql_insert.cfm) statement to insert data into a table.

To do this, open a new query window, type the SQL, then execute the statement (sample statement below).

In our case, the first column is an identity column, so we won't insert any data for that column. Therefore we need to specify the actual columns that we want to insert the data into (i.e. the table has three columns but we're only inserting two).

To insert multiple rows, there are a few ways to go about this. One way is to add multiple [INSERT](https://www.quackit.com/sql/tutorial/sql_insert.cfm) statements — one for each row to be inserted.

Another way is to separate each set of values with a comma. Like our sample SQL statement below.

#### Sample SQL Statement

INSERT INTO Artists (ArtistName, ActiveFrom)

VALUES

('Iron Maiden','1975-12-25'),

('AC/DC','1973-01-11'),

('Allan Holdsworth','1969-01-01'),

('Buddy Rich','1919-01-01'),

('Devin Townsend','1993-01-01'),

('Jim Reeves','1948-01-01'),

('Tom Jones','1963-01-01'),

('Maroon 5','1994-01-01'),

('The Script','2001-01-01'),

('Lit','1988-06-26'),

('Black Sabbath','1968-01-01'),

('Michael Learns to Rock','1988-03-15'),

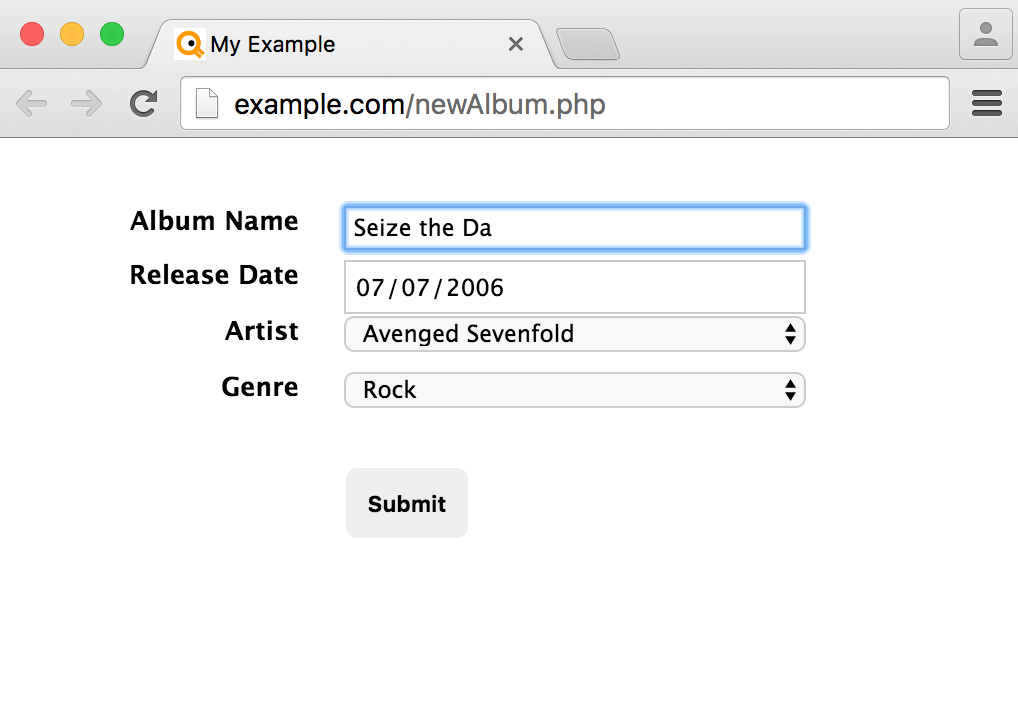
('Carabao','1981-01-01'),

('Karnivool','1997-01-01'),

('Birds of Tokyo','2004-01-01'),

('Bodyjar','1990-01-01');

### Website/Application

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/insert_data_in_sql_server_2016_5.png)

One of the most common methods of adding data to a client/server database like SQL Server is via an application.

The application could be anything, from a corporate CRM to a customer facing website. Data is added to SQL Server via the application, which generates the code to insert the data into the database.

This method is similar to the above SQL method, because most applications generate SQL code in order to insert data into the database. The same applies when reading from the database.

# SQL Server 2016: Import Data

SQL Server Management Studio includes an import wizard to help you import data from an external source.

You can import data to your SQL Server database from various other sources, including Excel spreadsheets, CSV files, and other databases such as Oracle, Access, and other SQL Server instances.

## Import a CSV File

Right now we'll import a CSV file into our Albums table that we created [earlier](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016.cfm) when we created a relationship.

You can download the CSV file here: [Albums.csv](https://www.quackit.com/sql_server/sql_server_2016/tutorial/Albums.csv)

The file contains the following contents:

AlbumId,AlbumName,ReleaseDate,ArtistId,GenreId

1,Powerslave,1984-09-03,1,1

2,Powerage,1978-05-05,2,1

3,Singing Down the Lane,1956-01-01,6,3

4,Ziltoid the Omniscient,2007-05-21,5,1

5,Casualties of Cool,2014-05-14,5,1

6,Epicloud,2012-09-18,5,1

7,Somewhere in Time,1986-09-29,1,1

8,Piece of Mind,1983-05-16,1,1

9,Killers,1981-02-02,1,1

10,No Prayer for the Dying,1990-10-01,1,1

11,No Sound Without Silence,2014-09-12,9,4

12,Big Swing Face,1967-06-01,4,2

13,Blue Night,2000-11-01,12,4

14,Eternity,2008-10-27,12,4

15,Scandinavia,2012-06-11,12,4

16,Long Lost Suitcase,2015-10-09,7,4

17,Praise & Blame,2010-06-26,7,4

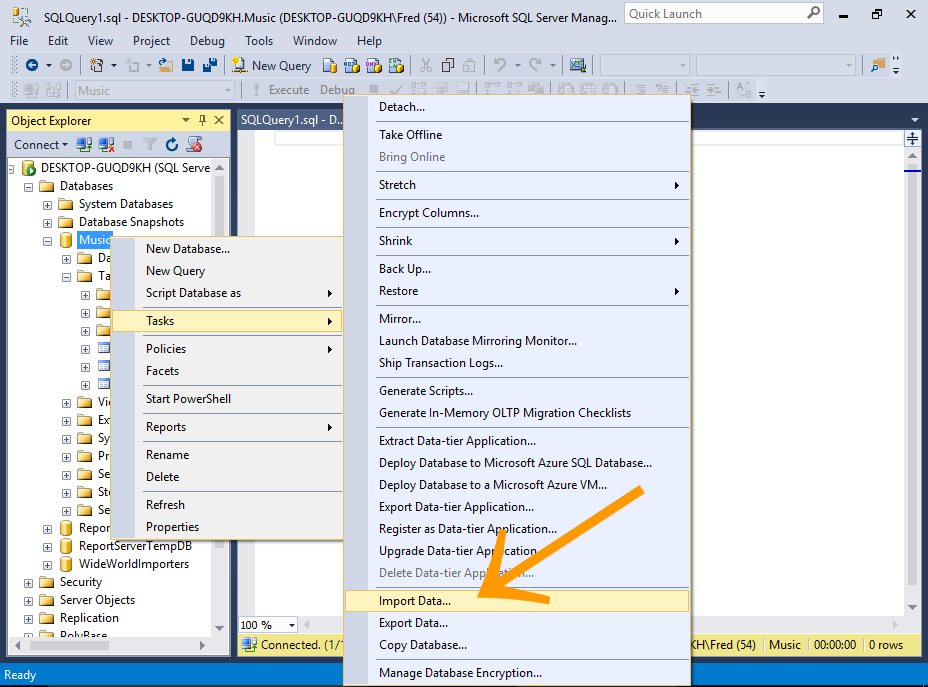
18,Along Came Jones,1965-05-21,7,4

19,All Night Wrong,2002-05-05,3,2

20,The Sixteen Men of Tain,2000-03-20,3,2

So save the above file into a location that you'll be able to get to from the SQL Server Import and Export Wizard, and let's get started.

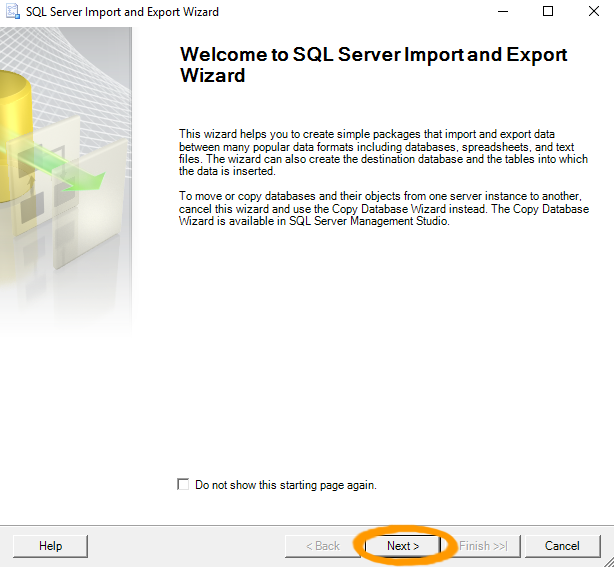
### Import Data

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_1.png)

Right-click on the database and select Tasks > Import Data...

You can also access the import wizard from various other places, including the Windows Start menu, the command prompt, and Visual Studio with SQL Server Data Tools (SSDT).

### Welcome to the SQL Server Import and Export Wizard

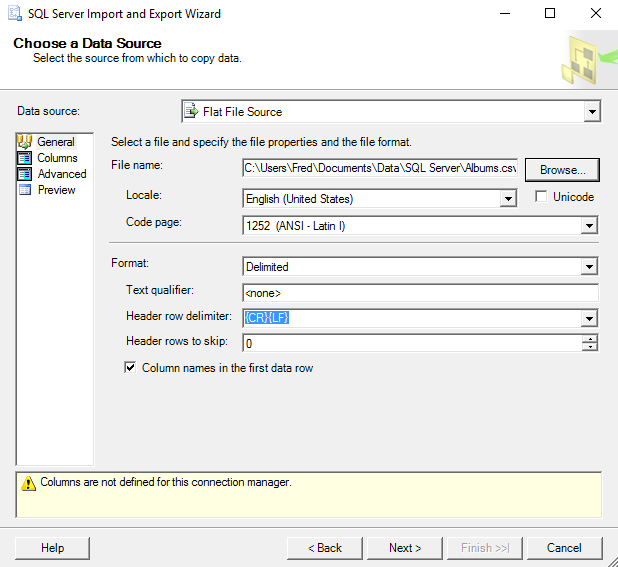
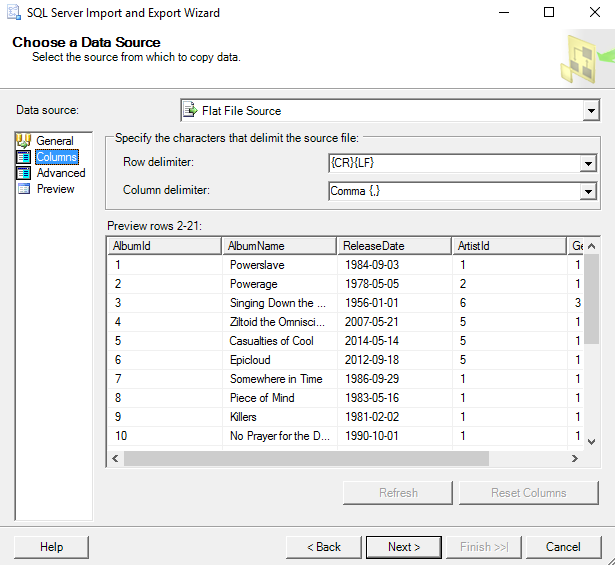
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_2.png)

The SQL Server Import and Export Wizard start page is displayed.

Click Next >.

Check Do not show this starting page again if you want to save time and clicks in the future.

### Choose a Data Source

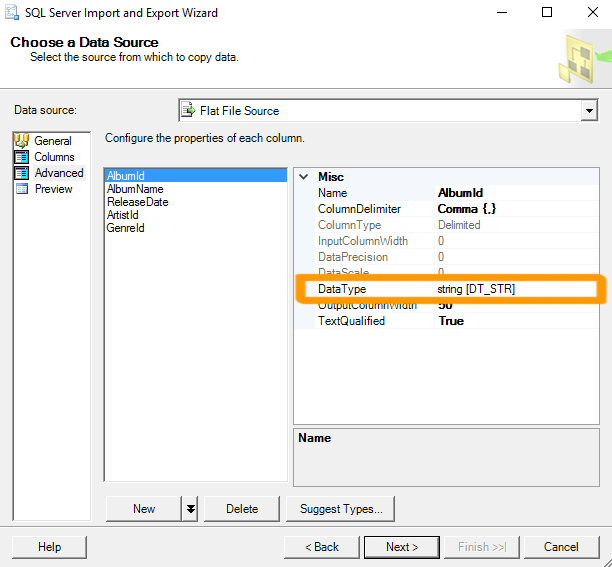
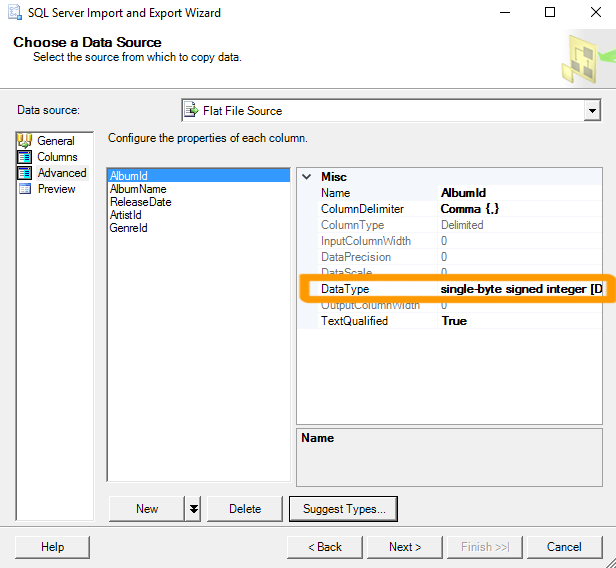
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_3.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_3a.png)

Select the file type and its location.

Ours is a CSV file, so select Flat File Source then browse to the file's location.

Also click on Columns in the left menu to check the delimiter settings, and the other options too if you like. The default settings should be fine in our case.

### Advanced Settings

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_3b.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_3c.png)

Check the advanced settings (by clicking on Advanced in the left menu) to make sure nothing looks out of the ordinary.

Click on each column name to view the properties for that column.

In our case, you may find that all columns are listed with a data type of string [DT\_STR], which could cause problems later on in the wizard.

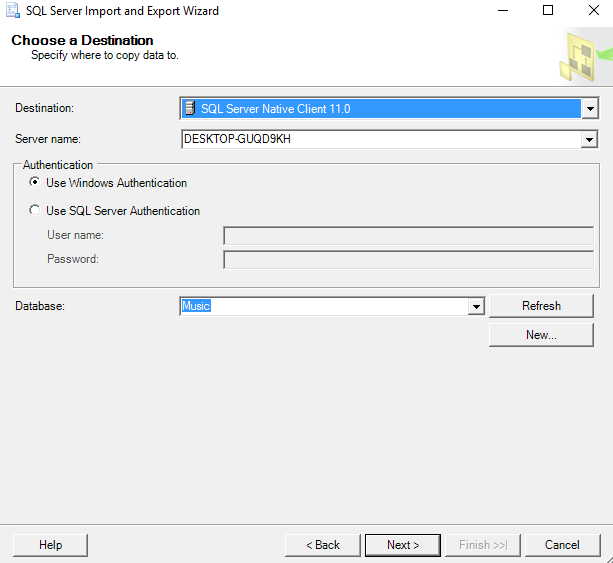
Take this opportunity to change the data types to the following:

|  |  |
| --- | --- |
| AlbumId | single-byte unsigned integer [DT\_UI1] |
| AlbumName | Unicode string [DT\_WSTR] |
| ReleaseDate | database date [DT\_DBDATE] |
| ArtistId | single-byte unsigned integer [DT\_UI1] |
| GenreId | single-byte unsigned integer [DT\_UI1] |

You can also use the Suggest Types... option. However, you may still need to modify some columns manually.

The wizard will warn you later on if it thinks there will be any issues converting data types from the source file. If that happens, you may need to come back and modify your selection here.

### Choose a Destination

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_4.png)

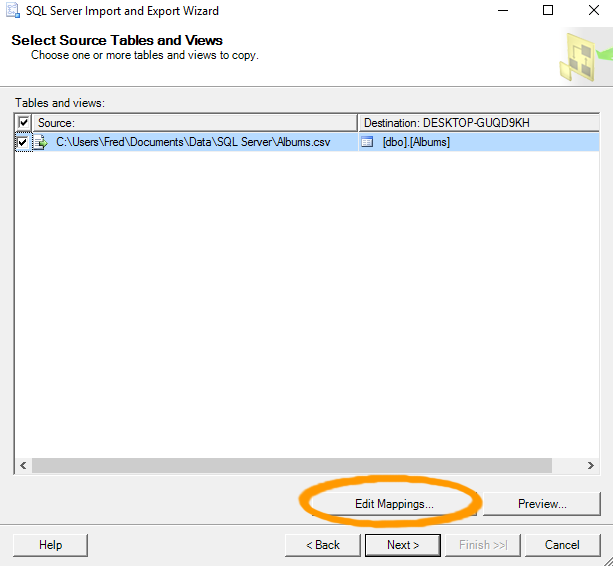
Here, you choose the data provider that matches the data storage format of the destination.

There are various options for SQL Server (SQL Server Native Client, the Microsoft OLE DB Provider for SQL Server, or the .NET Framework Data Provider for SQL Server). We'll choose the native client.

Select the SQL Server Native Client option drop-down list. If you need to change the Authentication mode and/or the destination database, do so now.

Click Next > to continue.

### Select Source Tables and Views

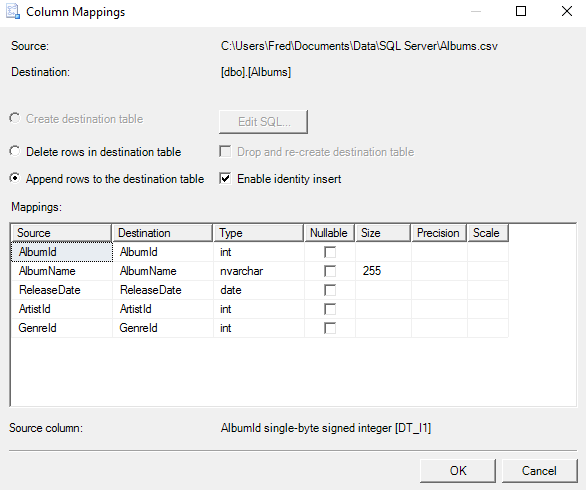
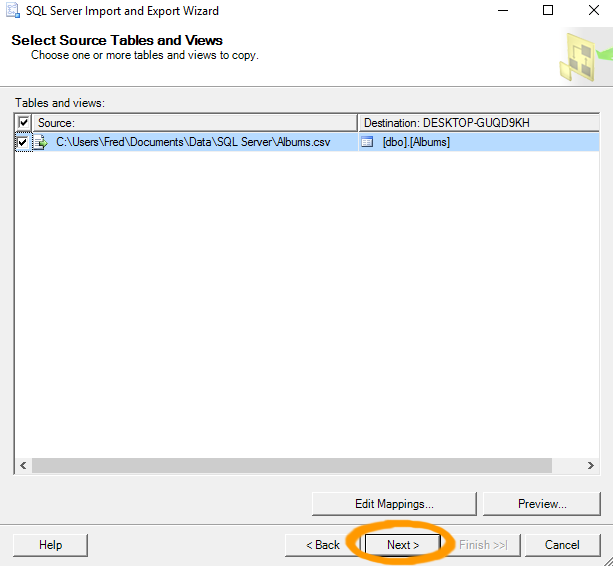
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_5.png)

Here, you get to select the source and destination.

Ours is a simple one and everything is selected correctly by default.

However, before continuing, click Edit Mappings... to review the column mappings.

### Column Mappings

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_5a.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_5b.png)

Check that the column mappings and other details are correct.

For our example, we need to make sure that Append rows to destination table is checked, and that the source and destination columns match, and with the correct data type.

Also check Enable identity insert to keep our existing values for the primary key column. This will insert the values from our file, rather than have SQL Server auto-generate them.

Click OK or Cancel to close the dialog box.

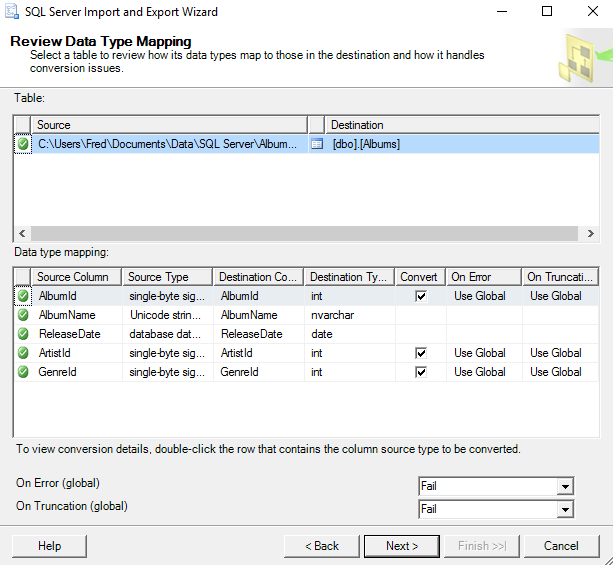
Then, once back in the Select Source Tables and Views dialog box, click Next > to continue.

If Create destination table is selected and the Append rows to destination table option is grayed out, check that the name of the source file matches the name of the destination table.

Also check its case (eg. artists.csv vs Artists.csv).

If your import file doesn't contain an identity column, leave Enable identity insert unchecked.

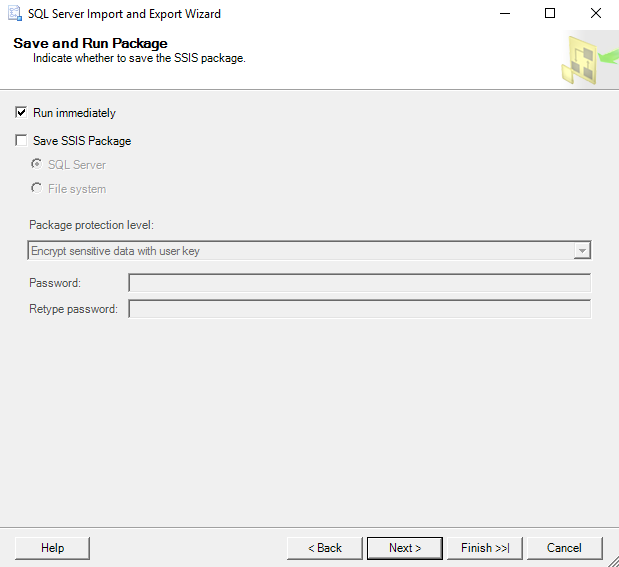
### Review Data Type Mapping

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_6.png)

Here's our chance to review the data type mappings that we set up earlier. If the wizard thinks there could be a problem with converting data types, it will warn you on this screen.

If all looks good, click Next >.

### Save and Run the Package

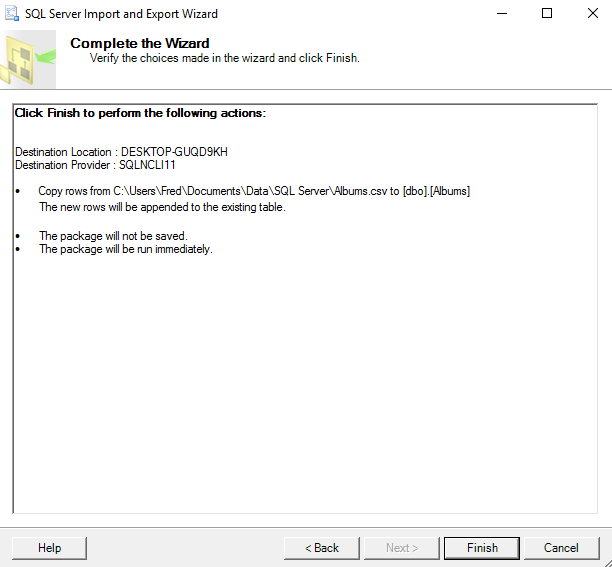
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_7.png)

The wizard gives you the option of saving the package so that you can use it again later.

In our case, we won't bother saving it.

Just keep Run immediately selected and click Next >.

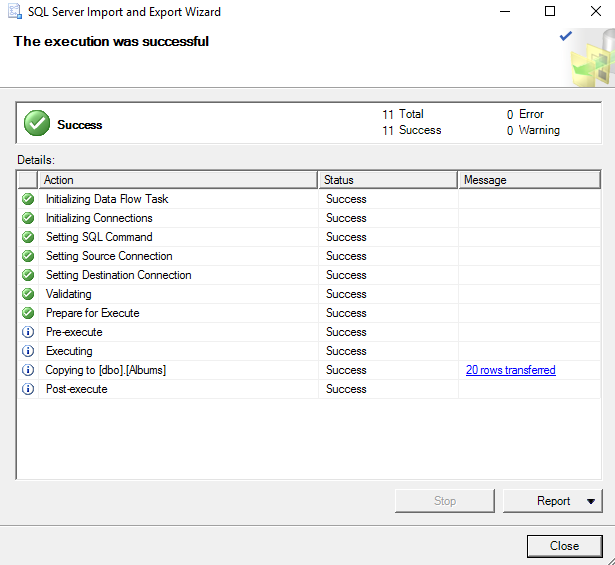
### Complete the Wizard

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_8.png)

One last chance to review all actions.

If it all looks good, click Finish.

### Execution Report: Success or Failure

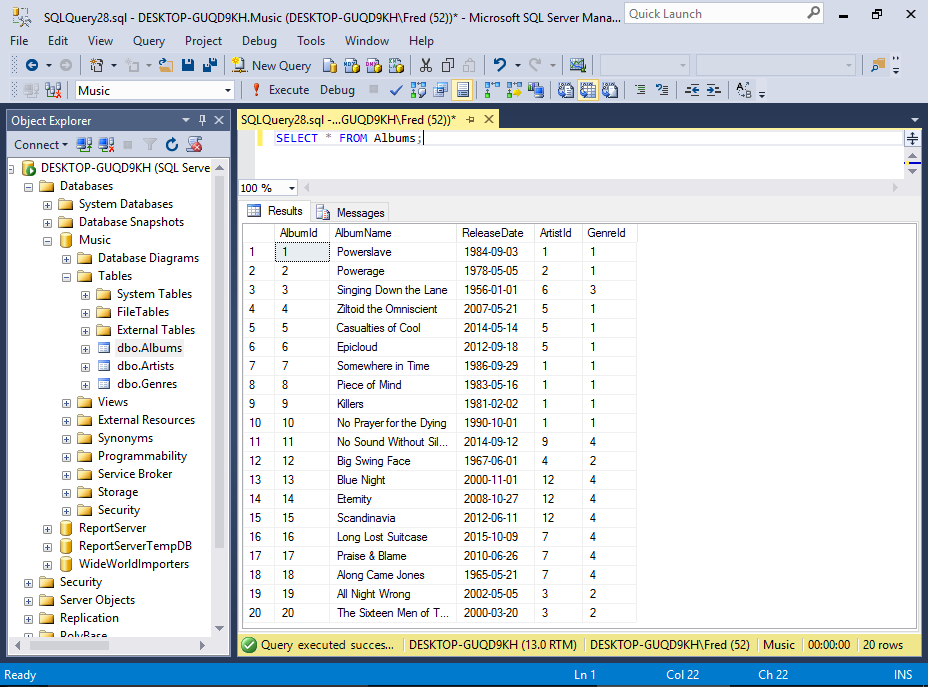
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_9.png)

The wizard will now try to execute the import. If there were any errors, they will be listed on this screen, and you will need to go back and correct the cause, before running it again.

If you get The execution was successful, click Close.

You can also save a report of the import by using the Report option on this screen.

### Check the Data

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/import_data_in_sql_server_2016_10.png)

# SQL Server 2016: Query Designer

Use the Query Designer to build complex queries across multiple tables without writing any code.

SQL Server Management Studio includes the Query Designer to assist in building queries. It is a visual tool that allows you to select the tables and columns you want in your query, as well as any filtering criteria.

No need to write any SQL code — the Query Designer will generate that for you.

## Use the Query Designer to Build a Simple Query

We'll now use the Query Designer to build a simple query. If you've been following along in this [tutorial](https://www.quackit.com/sql_server/sql_server_2016/tutorial/), you'd now have a database with three tables — all of which contain data. And because we've established a [relationship between these tables](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_relationship_in_sql_server_2016.cfm), we can now run queries across all three, returning related records.

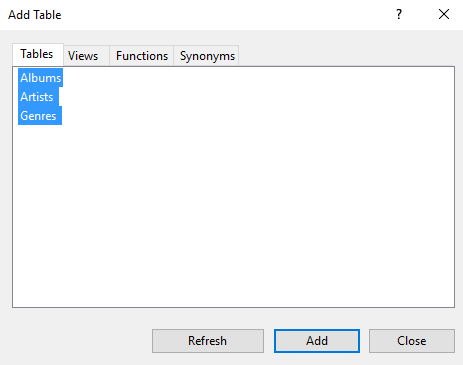
### Open the Query Designer[Screenshot of selecting the Query Designer in SSMS.](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016_1.png)[Screenshot of selecting the Query Designer in SSMS.](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016_1a.png)

Open a new query window by clicking on New Query in the toolbar.

Then select Query > Design Query in Editor... from the top menu.

If you can't see the Query option in the top menu, click inside the query window first. This will change the top menu items to be query-related options.

### Add the Tables

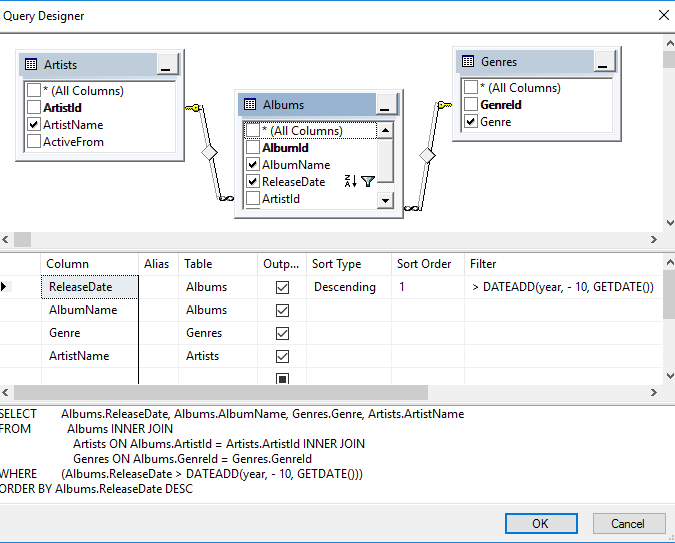
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016_2.png)

Here, you select which tables you'd like in your query.

Select all three and click Add.

Click Close to close the dialog box.

### Design the Query

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016_3.png)

You will now see the selected tables, and their relationships, in the Query Designer. Feel free to click and drag them around to provide a better visualization of their relationship with each other.

You can also re-size each pane by clicking its edge and dragging it up or down.

#### How to Design a Query

In the top pane (the Diagram Pane), click each column that you want to include in the query (whether you want to display it or not). Each column you select in the top pane will automatically appear in the middle pane.

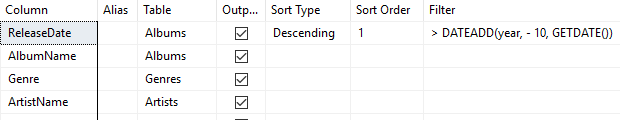
In the middle pane (the Grid Pane or Criteria Pane), use the Output checkbox to indicate which columns will be returned in the results. Use Sort Type to specify the order of the results by a given column. You can use Sort Order to specify which column will be sorted first, second, etc. Use Filter to add filtering criteria to filter the records returned.

The bottom pane (the SQL Pane) dynamically generates the SQL statement that your query produces. This is the statement that will be run when you close the Query Designer and execute the query.

#### Our Example

In our example, our query will return all albums (along with their genre, artist, and genre) that were released in the last ten years. The criteria to achieve this is >DATEADD(year, - 10, GETDATE()). The query will sort the results by the release date in descending order.

Here's a close-up of the Criteria Pane:

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016_3a.png)

You can change the order of the columns by clicking and dragging them up or down.

If you're reading this long after this tutorial was written, you might need to adjust the criteria to go back 20 years or more before you get any results.

Alternatively, you could add something a bit more modern to the music collection :)

Make sure you keep this query open in the query window because next, we will [save it as a view](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016.cfm).

# SQL Server 2016: Create a View

* [Query Designer](https://www.quackit.com/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016.cfm)
* [View Designer](https://www.quackit.com/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016.cfm)

In SQL Server, you can queries as views. Views are beneficial for many reasons, including security, usability, and convenience.

In SQL Server, a view is a virtual table whose contents are defined by a query. It is basically a pre-written query that is stored on the database.

A view consists of a [SELECT](https://www.quackit.com/sql/tutorial/sql_select.cfm) statement, and when you run a query against the view, you see the results of it like you would when opening a table. Views are referred to as virtual tables because they can pull together data from multiple tables, as well as aggregate data, and present it as though it is a single table.

## Benefits of Views

A view can be useful when there are multiple users with different levels of access, who all need to see portions of the data in the database (but not necessarily all of the data). Views can do the following:

* Restrict access to specific rows in a table
* Restrict access to specific columns in a table
* Join columns from multiple tables and present them as though they are part of a single table
* Present aggregate information (such as the results of the COUNT() function)

## How to Create a View

You create a view by using the CREATE VIEW statement, followed by the SELECT statement.

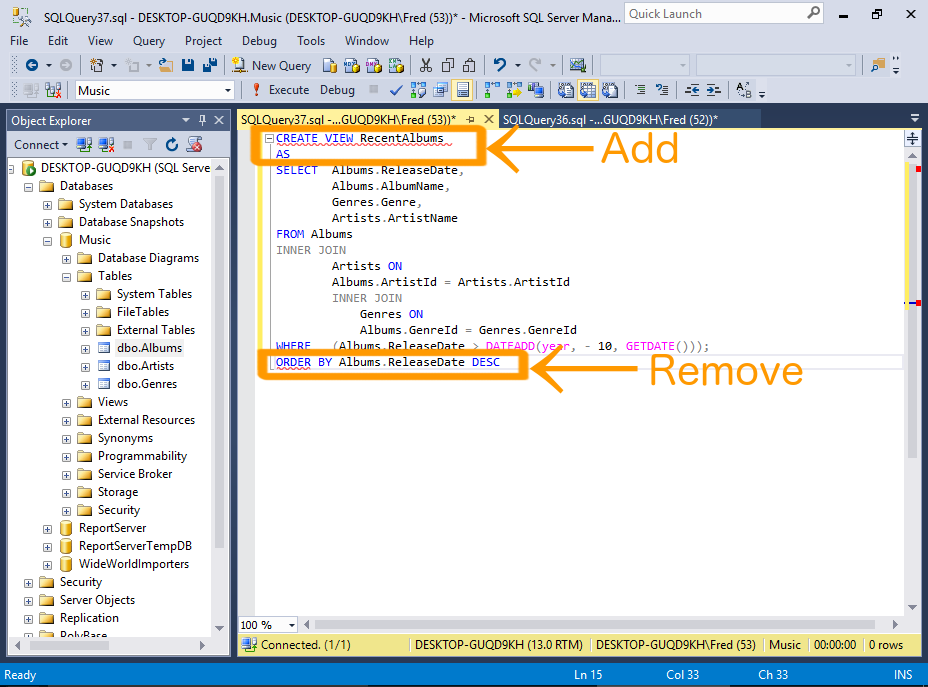
CREATE VIEW ViewName

AS

SELECT ...

We'll now create a view from our [previous query](https://www.quackit.com/sql_server/sql_server_2016/tutorial/query_designer_in_sql_server_2016.cfm).

### Design the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016_1.png)

Take the query from our previous example, and prefix it with CREATE VIEW RecentAlbums AS.

Also remove the ORDER BY clause, because views don't support this clause (unless TOP, OFFSET or FOR XML is also specified).

Also add a semicolon to the end of the statement, as a statement terminator (more on that below).

#### Sample Code

Below is the code from our example, with the ORDER BY clause removed, and a semicolon added as a statement terminator.

I've also reformatted it a bit to make it more readable).

CREATE VIEW RecentAlbums

AS

SELECT Albums.ReleaseDate,

Albums.AlbumName,

Genres.Genre,

Artists.ArtistName

FROM Albums

INNER JOIN

Artists ON

Albums.ArtistId = Artists.ArtistId

INNER JOIN

Genres ON

Albums.GenreId = Genres.GenreId

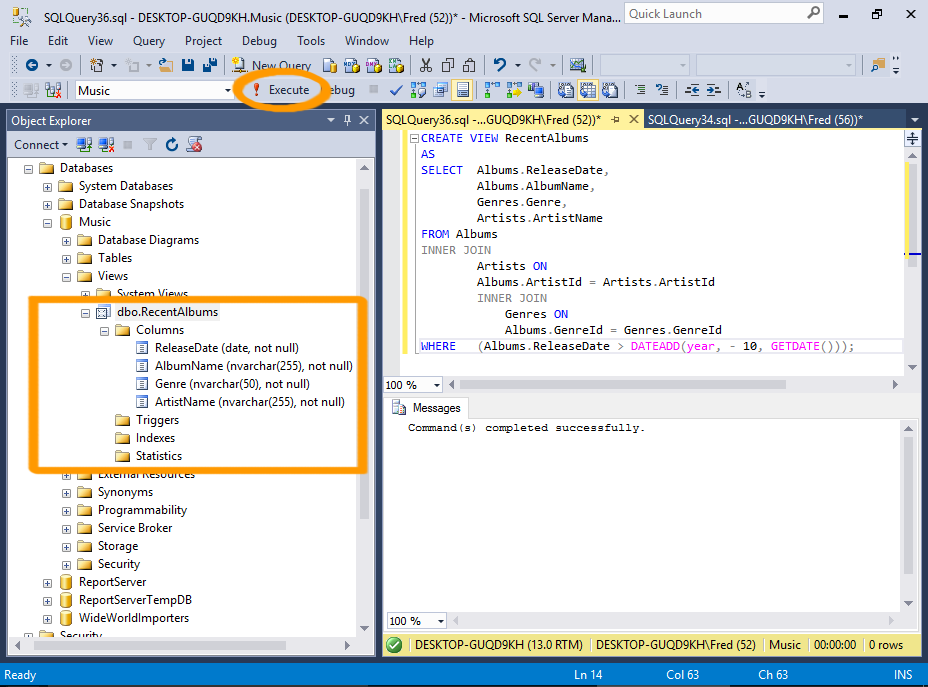
WHERE (Albums.ReleaseDate > DATEADD(year, - 10, GETDATE()));

In this example I added a semicolon to the end of the view. The Query Designer didn't include this, but it's good practice to include it.

The semicolon is part of the ANSI SQL-92 standard. It is a statement terminator character.

Also, Microsoft has [announced](https://msdn.microsoft.com/en-us/library/ms143729.aspx) that non-semicolon ending Transact-SQL statements are deprecated in SQL Server 2016, and they won't be supported in a future version (SQL Server has historically used the GO keyword as a statement terminator instead of the semicolon).

### Execute the View

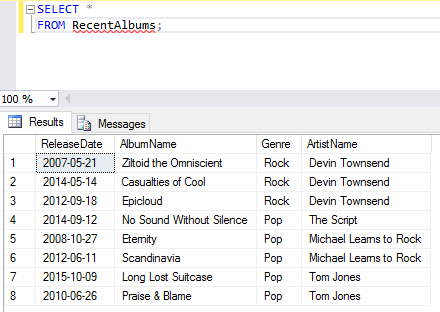
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016_2.png)

Now execute the view just as you would execute any other query.

Click Execute on the toolbar.

You can now navigate to the view in the Object Explorer. Expand it and you'll see the columns and their data types and properties — as though it was a table.

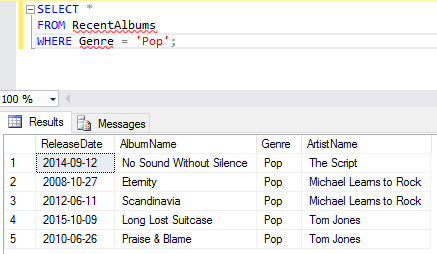
### Query the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016_3.png)

Now that the view has been created, you can query the view by running a SELECT statement against it.

So you can query our newly created view by using SELECT \* FROM RecentAlbums;.

### Filter the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016_4.png)

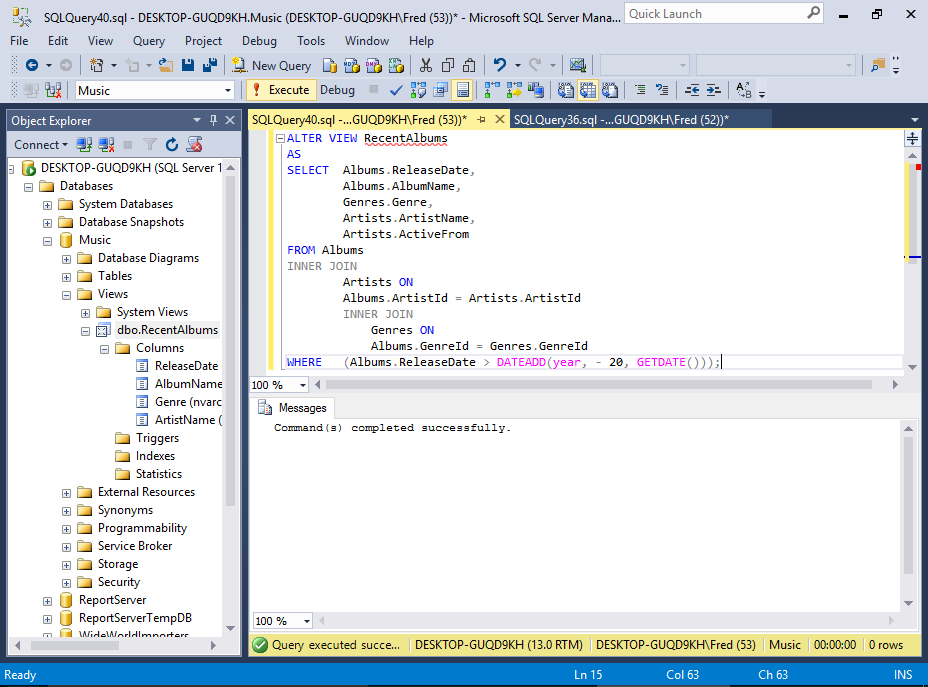
One of the good things about views is, you can apply your own filtering criteria against them — further filtering the results.

For example, you could add WHERE Genre = 'Pop', so that the view only returns pop albums from the last 10 years.

## Alter a View

You can modify your view by using the ALTER VIEW statement instead of the CREATE VIEW statement.

### Design the Altered View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016_5.png)

Here we will modify our view to return albums over the past 20 years instead of just 10.

It's a pretty old school collection, so any album released within the last 20 years is classified as "recent" :)

We'll also return another column: Artists.ActiveFrom

#### Sample Code

Here's the code we use for the example:

ALTER VIEW RecentAlbums

AS

SELECT Albums.ReleaseDate,

Albums.AlbumName,

Genres.Genre,

Artists.ArtistName,

Artists.ActiveFrom

FROM Albums

INNER JOIN

Artists ON

Albums.ArtistId = Artists.ArtistId

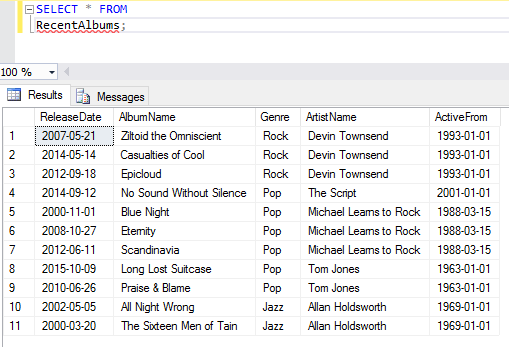
INNER JOIN

Genres ON

Albums.GenreId = Genres.GenreId

WHERE (Albums.ReleaseDate > DATEADD(year, - 20, GETDATE()));

### Query the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016_6.png)

Now, querying the view will return 20 years worth of albums. It will also display the date the artist was active from.

# SQL Server 2016: View Designer

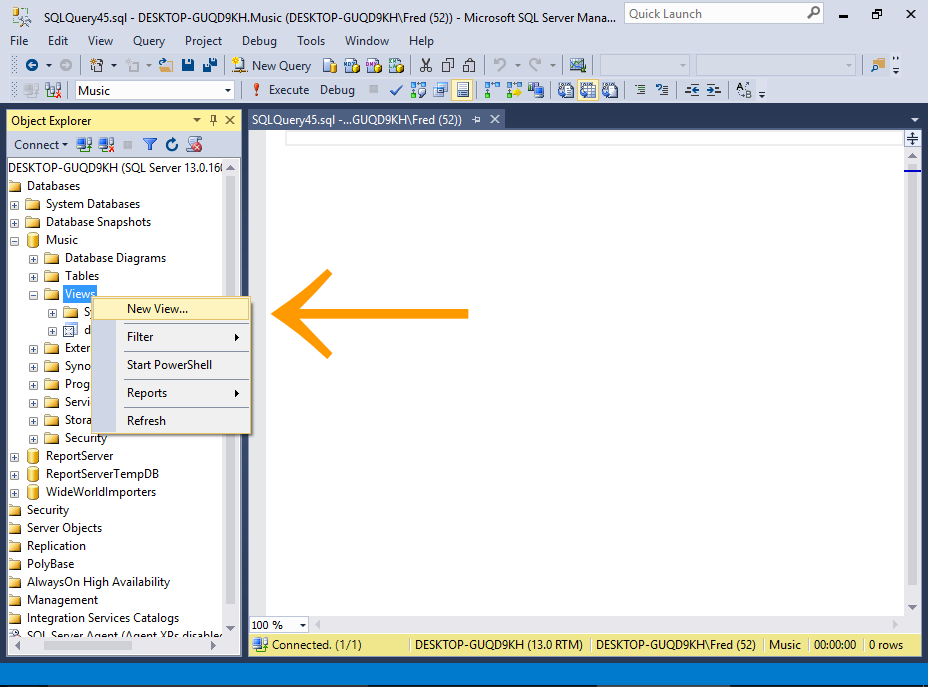
* [Create a View](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016.cfm)
* [Create a Stored Procedure](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016.cfm)

The View Designer is a visual tool that can help you build views.

The View Designer is just like the Query Designer, in that it provides a visual way of designing a query. It saves you the trouble of designing the query in Query Designer, then doing the extra coding to convert the query into a view (like we did [previously](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016.cfm)a).

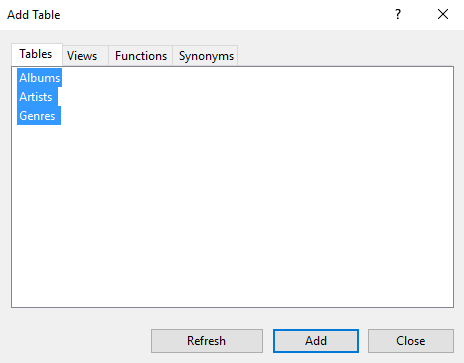
Now we will use the View Designer to create another view.

### Launch the View Designer

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_1.png)

Right-click on the Views node and select New View....

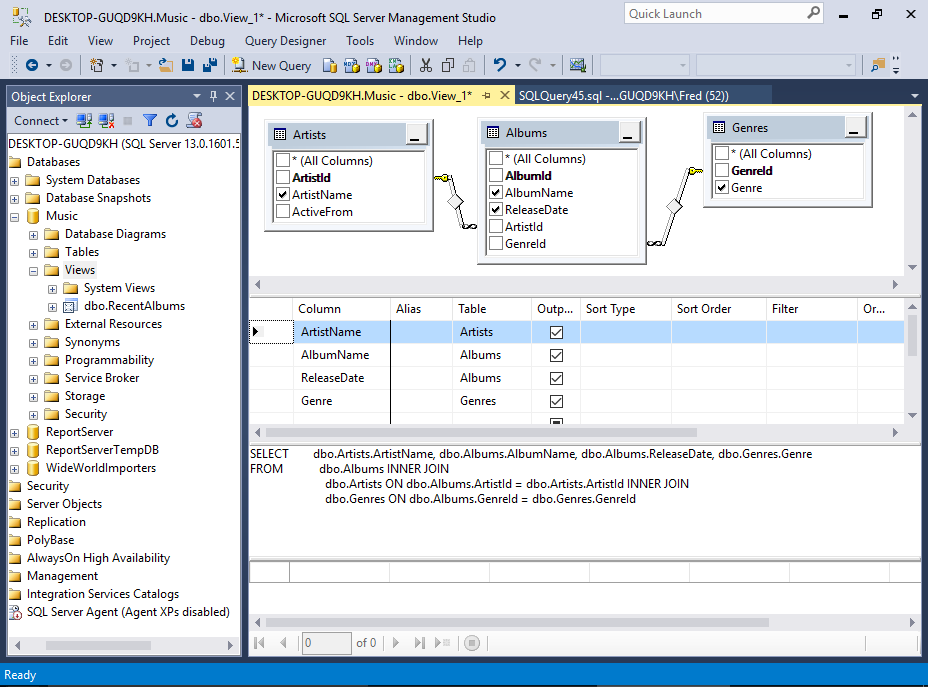
### Add the Tables

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_2.png)

Select the tables that you want to include in your view.

In our case, select all and click Add, then click Close to close the dialog box.

### Design the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_3.png)

You will now see the selected tables, and their relationships — just like in the Query Designer.

One difference is that the View Designer is sitting inside a query window. When we used the Query Designer, it was opened in a pop-up dialog, which prevented us from accessing any of the toolbar options, etc.

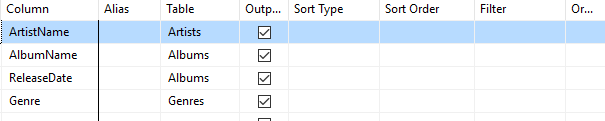
But having the View Designer opened inside a query window allows us to access the toolbar and other options as required.

Another difference is that we now have a fourth pane — the Results Pane. This allows us to execute the SQL to see what effect it has on the results before we save the view.

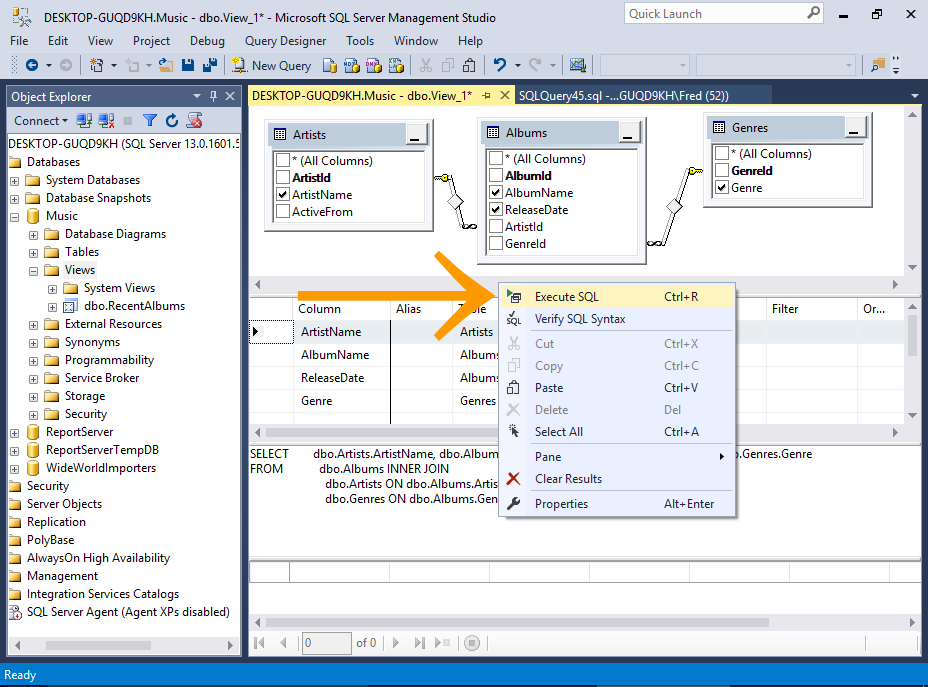
#### Our Example

We'll keep it simple this time and select four columns from the three tables. We won't provide any criteria.

Here's a close-up of the Criteria Pane:

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_3a.png)

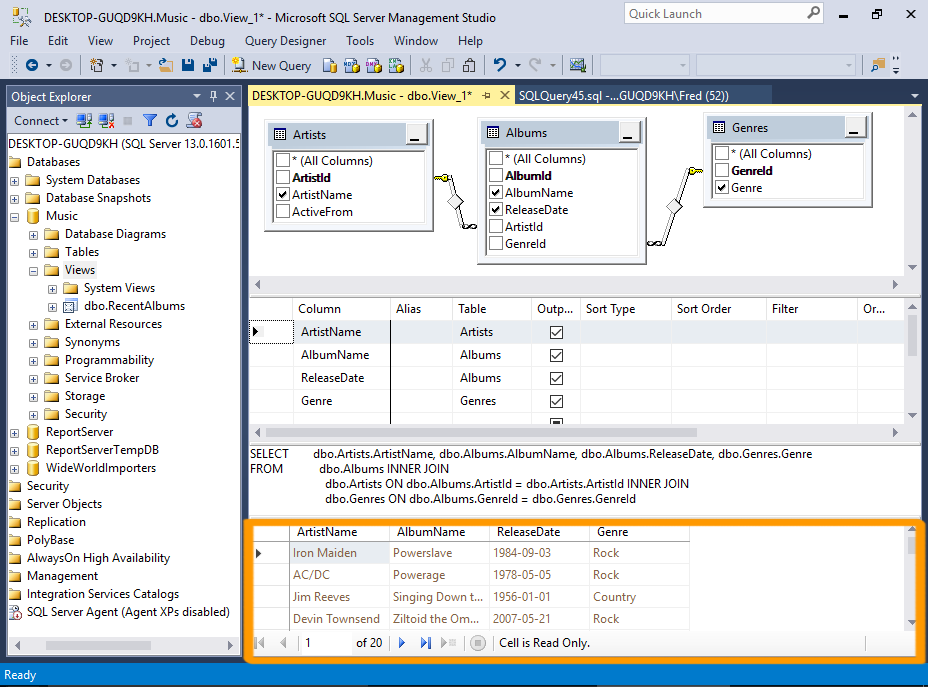
### Execute the SQL

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_4.png)

You can test the view before you save it by executing the SQL while in the View Designer.

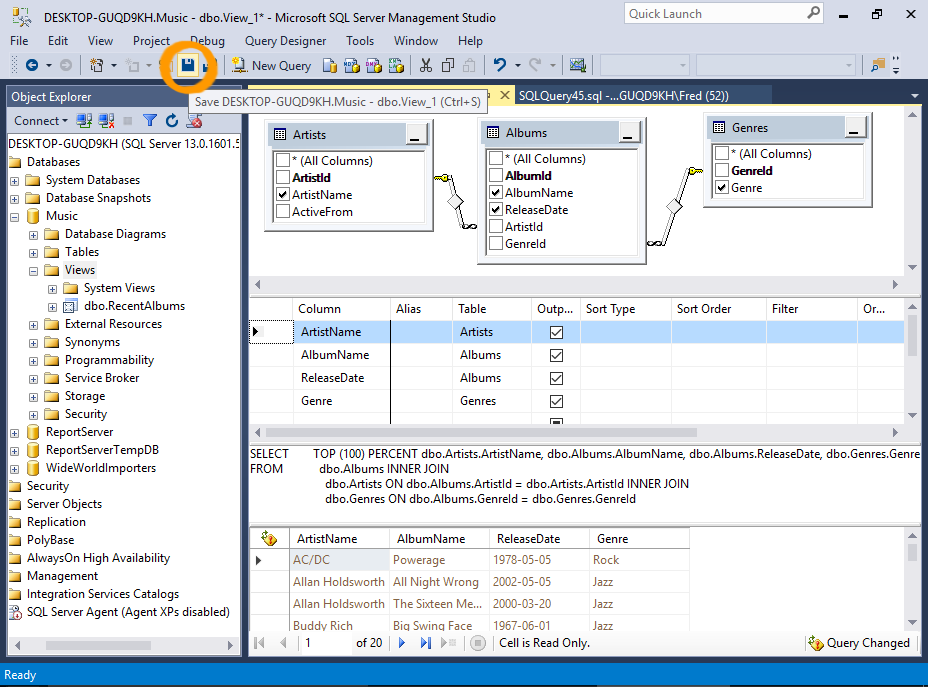
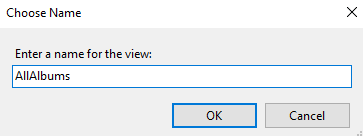
To do this, right-click anywhere in the design area and select Execute SQL (or press Ctrl+R on your keyboard).

### The Results

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_5.png)

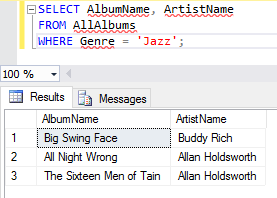
The results appear in the Results Pane at the bottom.

### Save the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_6.png)[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_6a.png)

To save the view, click the Save icon on the toolbar, then name the view at the prompt.

### Query the View

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016_7.png)

Now you can query the view with a SELECT statement as if it were a table.

**Naming Conventions**

Some database developers like to prefix their name view names with v or v\_ or similar to help distinguish between views and other database objects.

# SQL Server 2016: Create a Stored Procedure

* [View Designer](https://www.quackit.com/sql_server/sql_server_2016/tutorial/view_designer_in_sql_server_2016.cfm)
* [Save Query to CSV](https://www.quackit.com/sql_server/sql_server_2016/tutorial/save_query_results_to_csv_file_in_sql_server_2016.cfm)

A stored procedure is a group of SQL statements compiled into one. Stored procedures can include business logic and other programming constructs.

In SQL Server, a stored procedure is a group of one or more Transact-SQL statements or a reference to a Microsoft .NET Framework common runtime language (CLR) method.

## Programmability

But a stored procedure is more than just a long script. It's a script that has been saved in SQL Server specifically under the Stored Procedures node, and it can:

* Accept input parameters (and return multiple values in the form of output parameters to the calling program).
* Contain programming statements.
* Return a status value to a calling program to indicate success or failure, and the reason for any failure.

Stored procedures often contain business logic. For example, a stored procedure can accept parameters that are passed to it and test against those parameters using IF statements. Eg, if the parameter is one value, do this, if it's another value, do that.

Stored procedures can improve performance in an application, because the stored procedure is parsed and optimized as soon as it's created, and then stored in memory. Running a conditional query via stored procedure can be extremely quick - compared to an application that sends a query across the network, to the SQL Server, then has all the data returned to it across the network so it can filter through it, and pick out only the records it's interested in.

## Benefits of Stored Procedures

Here are some of the main benefits in using stored procedures:

|  |  |
| --- | --- |
| **Benefit** | **Explanation** |
| Modular programming | You can write a stored procedure once, then call it again and again, from different parts of an application (and even from multiple applications). |
| Performance | Stored procedures provide faster code execution and reduce network traffic.   * Faster execution: Stored procedures are parsed and optimized as soon as they are created and the stored procedure is stored in memory. This means that it will execute a lot faster than sending many lines of SQL code from your application to the SQL Server. Doing that requires SQL Server to compile and optimze your SQL code every time it runs. * Reduced network traffic: If you send many lines of SQL code over the network to your SQL Server, this will impact on network performance. This is especially true if you have hundreds of lines of SQL code and/or you have lots of activity on your application. Running the code on the SQL Server (as a stored procedure) eliminates the need to send this code over the network. The only network traffic will be the parameters supplied and the results of any query. |
| Security | Users can execute a stored procedure without needing to execute any of the statements directly. Therefore, a stored procedure can provide advanced database functionality for users who wouldn't normally have access to these tasks, but this functionality is made available in a tightly controlled way. |

## How to Create a Stored Procedure

To create a stored procedure, use the CREATE PROCEDURE statement, followed by the code that makes up the stored procedure. If your stored procedure is going to accept parameters, they need to be included after the name.

CREATE PROCEDURE myStoredProcedure AS

...

​

OR

​

CREATE PROCEDURE myStoredProcedure @ParameterName DataType} AS

...

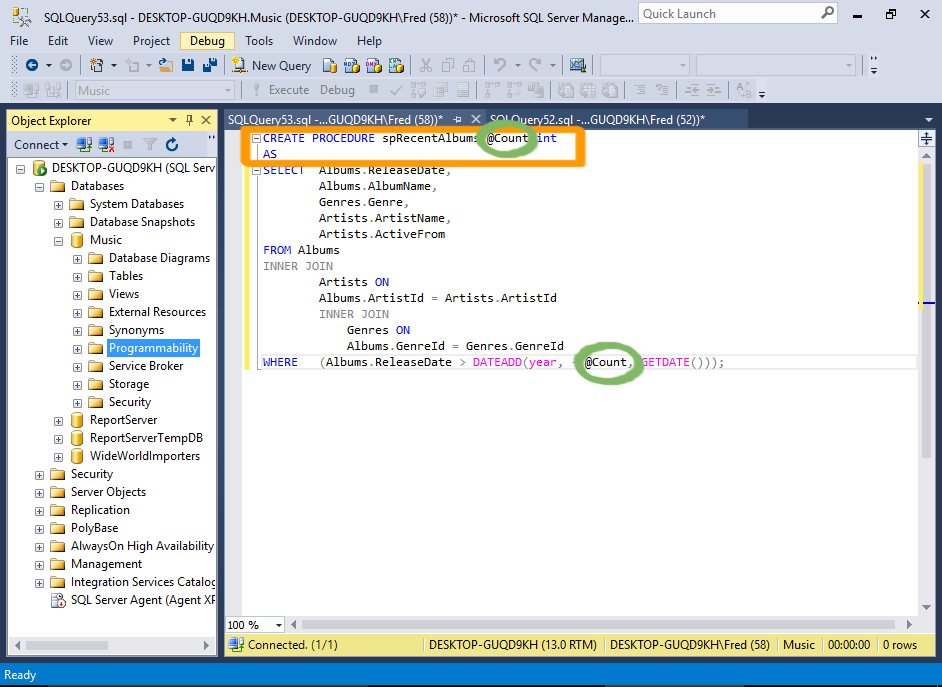
We'll now create a stored procedure based on one of the views that we created [earlier](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_view_in_sql_server_2016.cfm) — the RecentAlbums view.

This view returns all albums released in the last 20 years. This is fine as long as it will only ever need to look back 20 years. But what if you want the user to choose how many years it should cover?

A stored procedure can solve this problem.

We will create a stored procedure that accepts a parameter. The value of the parameter will be the number of years to search back over. Therefore, this value can be specified by the user whenever they execute the stored procedure.

### Design the Stored Procedure

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_1.png)

Open a new query window and add the code for the stored procedure.

In our case, we will copy/paste the code from the RecentArtists view and modify the top part, so that it becomes a stored procedure.

We will add a parameter called @Count that will determine how many years the stored procedure should look back.

So we will replace the hardcoded value of 20 with @Count

#### Sample Code

Here's the code from our example:

CREATE PROCEDURE spRecentAlbums @Count int

AS

SELECT Albums.ReleaseDate,

Albums.AlbumName,

Genres.Genre,

Artists.ArtistName,

Artists.ActiveFrom

FROM Albums

INNER JOIN

Artists ON

Albums.ArtistId = Artists.ArtistId

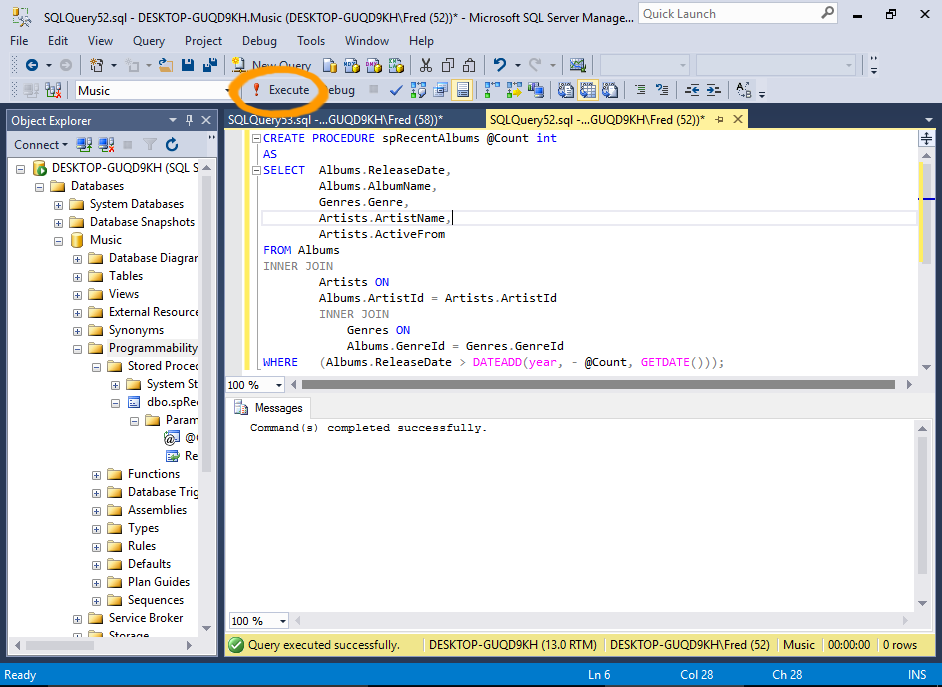
INNER JOIN

Genres ON

Albums.GenreId = Genres.GenreId

WHERE (Albums.ReleaseDate > DATEADD(year, - @Count, GETDATE()));

### Create the Stored Procedure

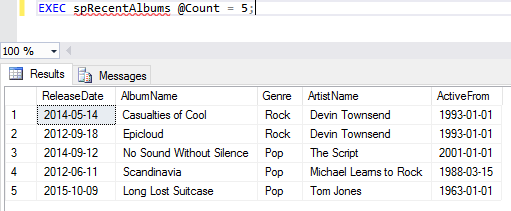
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_2.png)

Once it all looks good, you can execute the statement to create the stored procedure.

Click Execute to create the stored procedure.

Once the stored procedure has been created, you can see it in the Object Explorer (you might need to refresh the Stored Procedures node first).

### Execute the Stored Procedure

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_3.png)

Now that it has been created, you can execute the stored procedure.

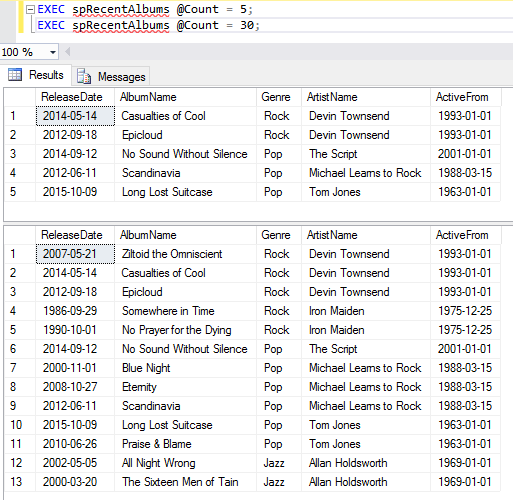
Open a new query window, add this:

EXEC spRecentAlbums @Count = 5;

Now click Execute from the toolbar.

The stored procedure will return all albums released in the last 5 years.

### Try different Parameters

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_3a.png)

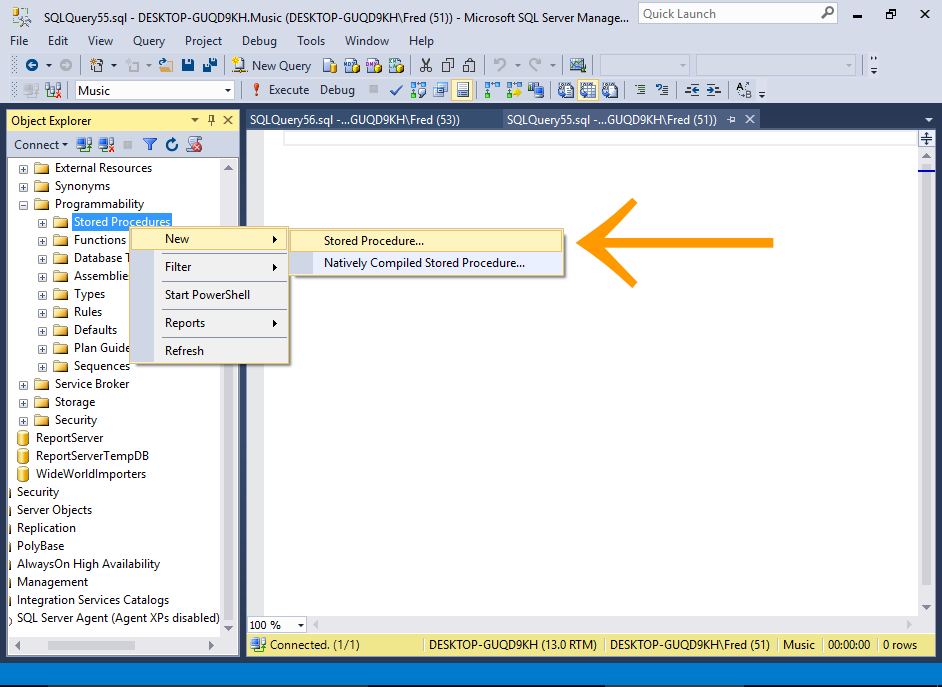
Try changing the value of the parameter to see how this affects the results.

You can also run multiple statements one after the other. A new results pane will appear for each statement.

### Stored Procedure Template

In SQL Server 2016, you can create a stored procedure by right-clicking on the Stored Procedures node in the Object Explorer and selecting New > Stored Procedure... or New > Natively Compiled Stored Procedure....

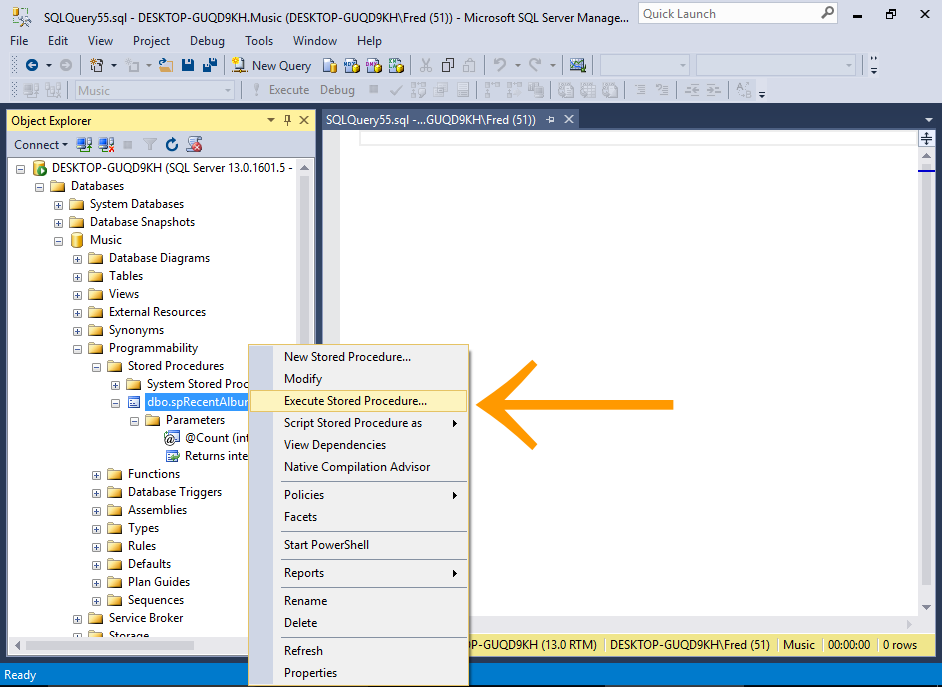
This will open a template that's ready to be populated with your own specific procedure.

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_3b.png)

### Execute a Stored Procedure via the GUI

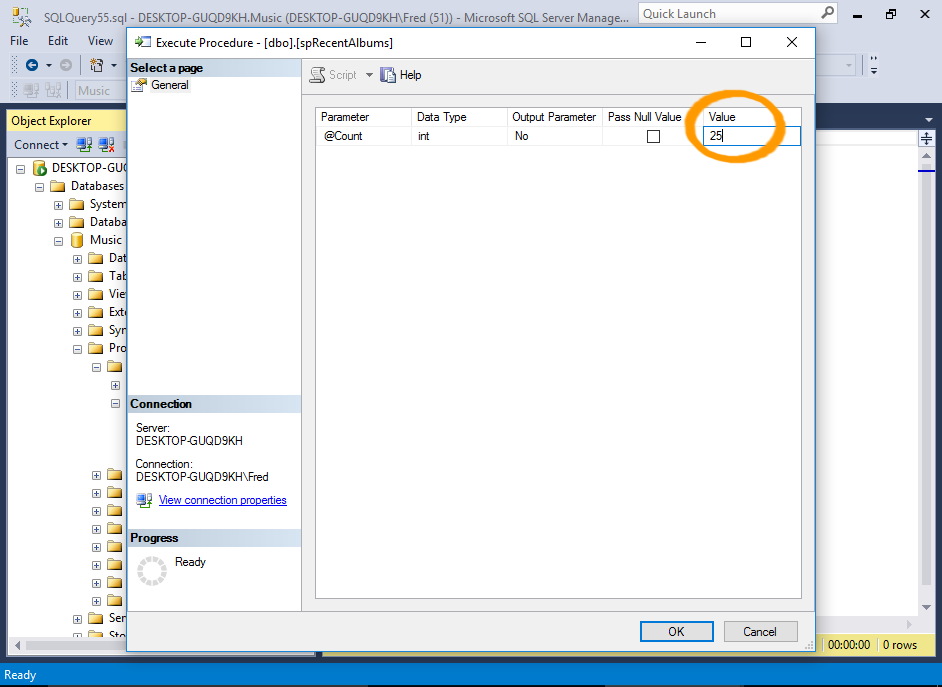
You can also use the graphical user interface to execute a stored procedure.

### Launch the Execute Procedure Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_4.png)

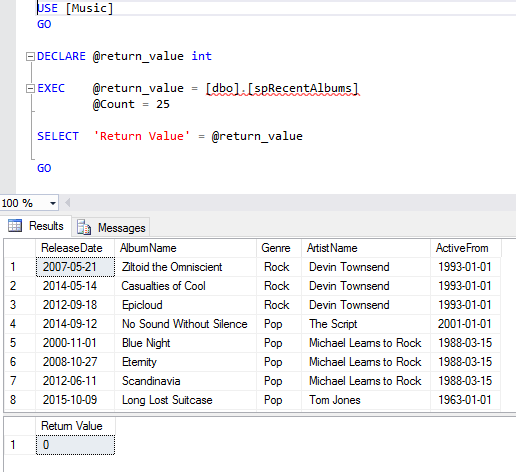
In the Object Explorer, right-click on the stored procedure and select Execute Stored Procedure....

### Supply Parameters

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_5.png)

Enter a value for any parameters that the stored procedure requires, then click OK.

### The Results

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016_6.png)

The results are displayed.

## Modify a Stored Procedure

If you need to modify an existing stored procedure, simply replace CREATE with ALTER (along with the updated procedure).

This example modifies the stored procedure so that the results are sorted by release date in descending order:

ALTER PROCEDURE spRecentAlbums @Count int

AS

SELECT Albums.ReleaseDate,

Albums.AlbumName,

Genres.Genre,

Artists.ArtistName,

Artists.ActiveFrom

FROM Albums

INNER JOIN

Artists ON

Albums.ArtistId = Artists.ArtistId

INNER JOIN

Genres ON

Albums.GenreId = Genres.GenreId

WHERE (Albums.ReleaseDate > DATEADD(year, - @Count, GETDATE()))

ORDER BY Albums.ReleaseDate DESC;

## System Stored Procedures

SQL Server includes a large number of system stored procedures to assist in database administration tasks. Many of the tasks you can perform via the GUI can be done via a system stored procedure. For example, some of the things you can do with system stored procedures include:

* Configure security accounts
* Set up linked servers
* Create a database maintenance plan
* Create full text search catalogs
* Add remote login
* Configure replication
* Set up scheduled jobs
* and much more...

System stored procedures are prefixed with sp\_, so it's best to avoid using that prefix for your own procedures.

## Naming Conventions

It is a good idea to develop a consistent naming convention for your stored procedures (and for all other objects in your database).

Some people prefix their stored procedures with usp\_ (to indicate a user-defined stored procedure), others begin it with a SQL keyword such as select, insert, update, delete. Others use an abbreviation for the application.

Some use underscores to separate each word in the stored procedure (eg, recent\_albums), while others will use title case (eg, RecentAlbums).

Therefore, it is possible that our stored procedure could been named any of the following, depening on the naming convention being used.

* RecentAlbums
* recent\_albums
* uspRecentAlbums
* usp\_recent\_albums
* selectRecentAlbums
* select\_RecentAlbums
* select\_recent\_albums
* getRecentAlbums
* get\_recent\_albums

You get the picture. The important thing is consistency. Choose one and stick with it. It will make it easier when you need to use a stored procedure. Imagine having scores, or even hundreds of stored procedures, and every time you go to execute one, you need to navigate to it in the Object Explorer purely because you can't remember whether you called it usp\_RecentAlbums or uspRecentAlbums.

As mentioned, avoid using sp\_ as the prefix for your stored procedure names. SQL Server uses this prefix for the system stored procedures.

SQL Server searches the system stored procedures first, so at best, you'll take a performance hit. At worst, your procedure won't run (if it shares a name with a system stored procedure).

# SQL Server 2016: Save Query Results to a CSV File

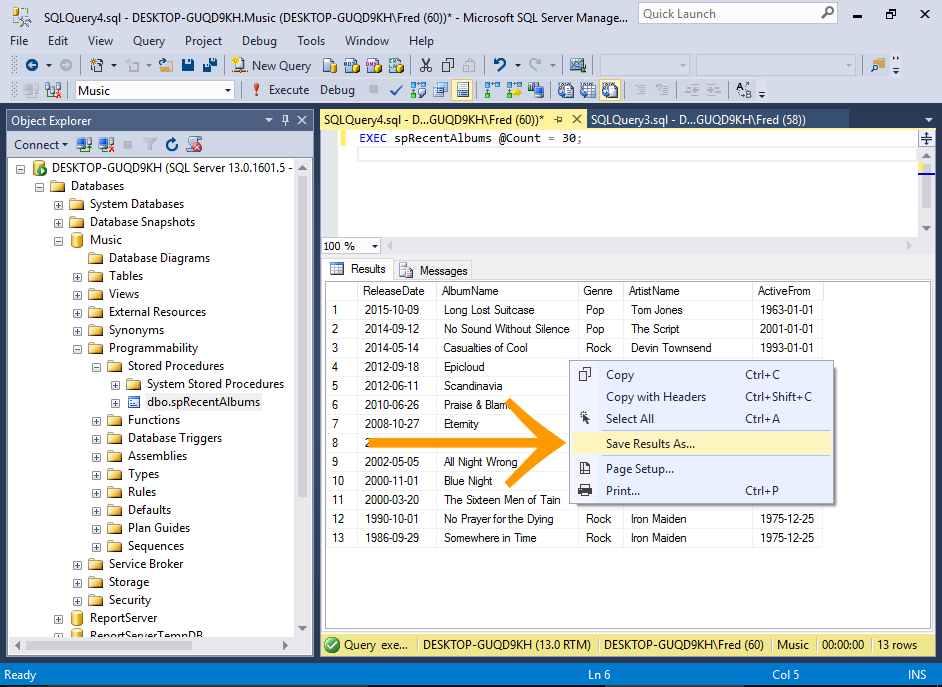
* [Create a Stored Procedure](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_stored_procedure_in_sql_server_2016.cfm)
* [Create a Login](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016.cfm)

You can save the results of any query to a CSV file with little more than a click of the mouse.

SQL Server makes it easy for you to save the results of a query to a CSV file (or other text file).

Here's how.

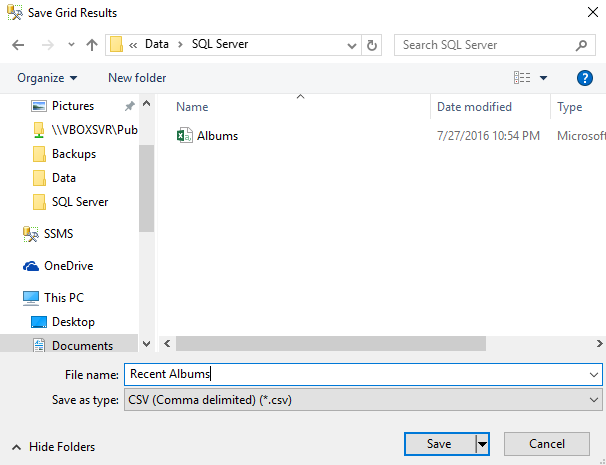
### Query Results

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_1.png)

Run a query.

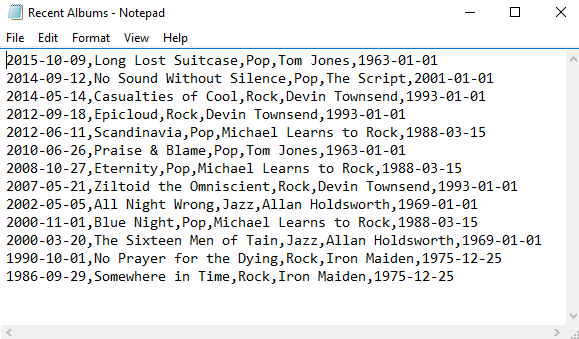
Now right-click in the Results Pane and select Save Results As... from the contextual menu.

### Save the File

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_2.png)

Name the file and location and click Save.

### Open the File

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_3.png)

Now locate the file and open it in Notepad (or your preferred application for opening CSV files).

You can see the results have been saved in CSV format.

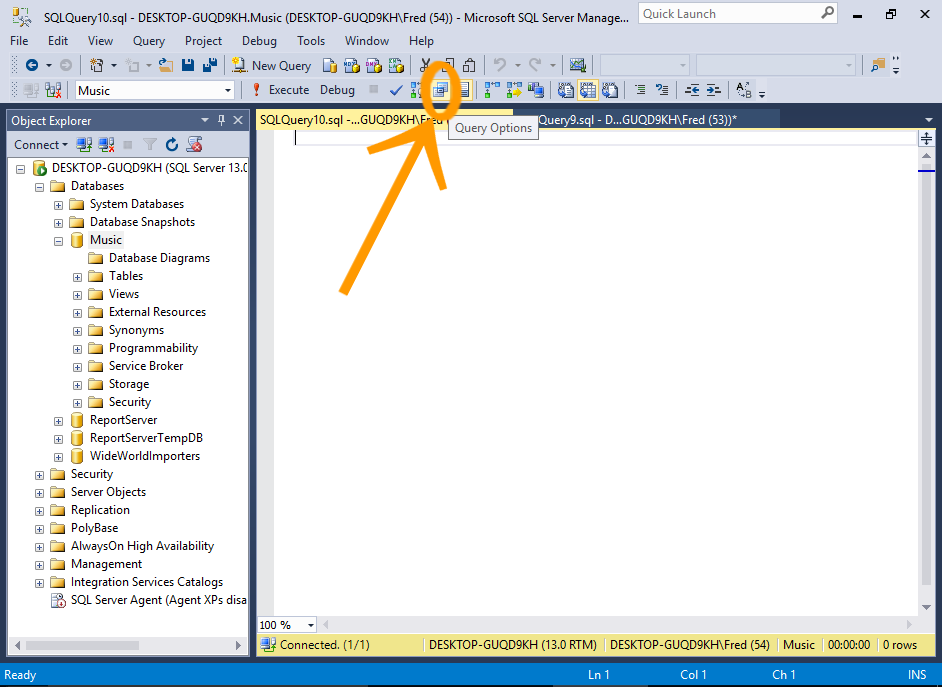
## Add a Header Row

You might have noticed that the above CSV file doesn't contain a header row.

This may or may not be what you want.

If you need it to contain a header row, do the following.

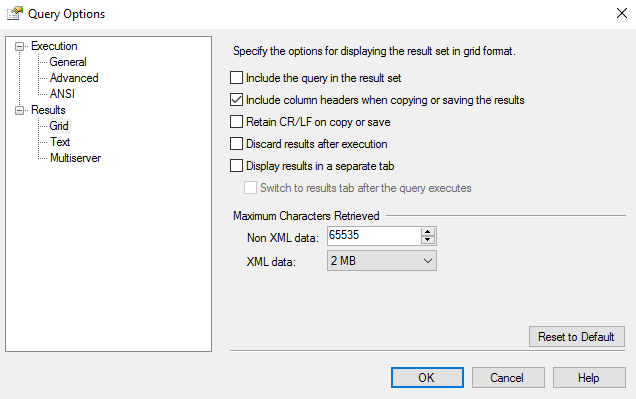
### Launch the Query Options Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_4.png)

Open a new query window, and click the Query Options icon in the toolbar.

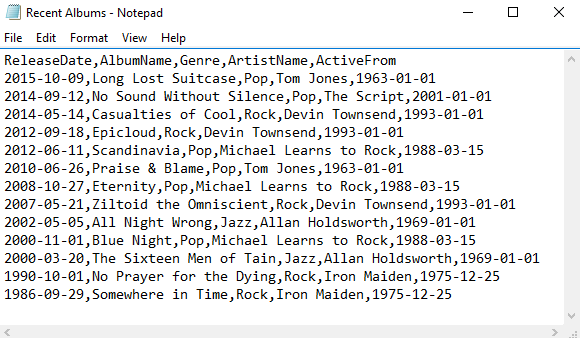
You can also go to Tools > Options in the top menu.

### Include Column Headers

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_5.png)

Under Results > Grid, put a check mark next to Include column headers when copying or saving the results.

### Save another Query and Check the File

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_6.png)

Now, run another query, save it, then check the file.

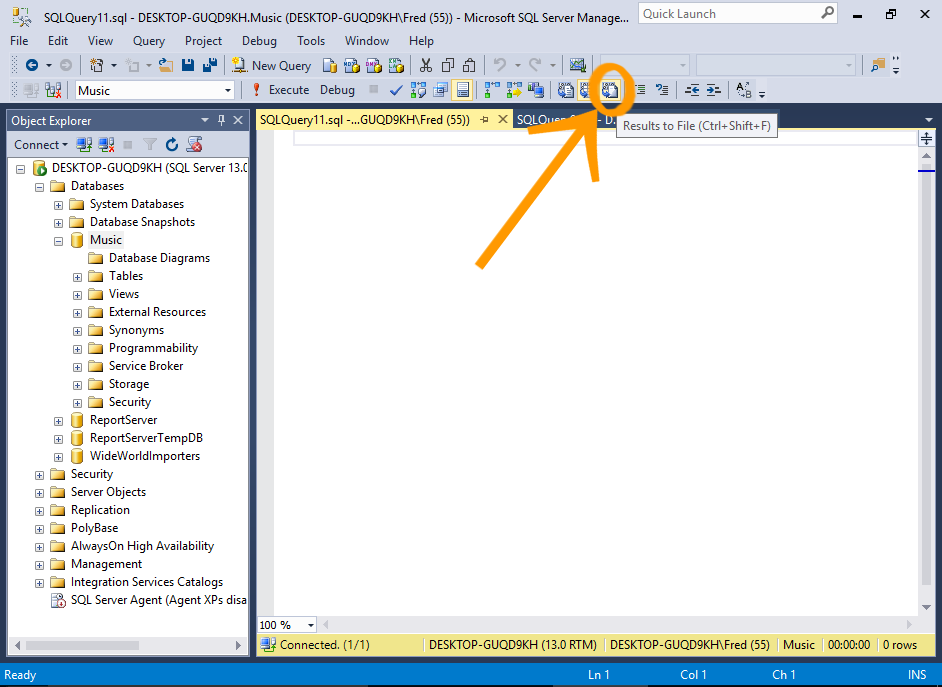
When you open the file, it will contain column headers.

## Results to File

You can also use the Results to File option for saving your query results to a file.

This option will prompt you to save the results whenever you run a query. So this can be handy if you have many queries to save.

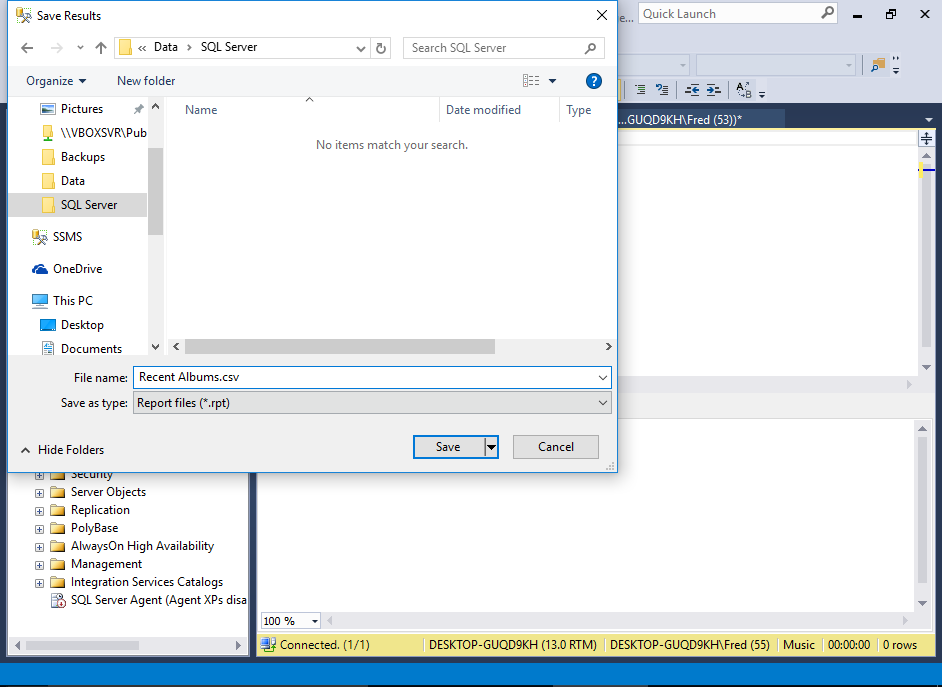
### Select the Results to File Option

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_7.png)

Open a new query window, and click the Results to File icon in the toolbar.

You can also right-click in the query window and select Results To > Results to File.

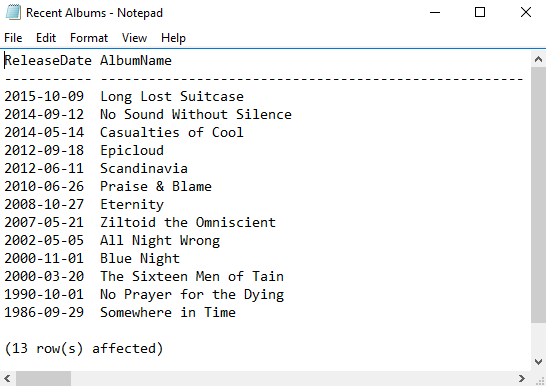
### Run a Query

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_8.png)

Now run a query. You will be prompted to save the file (instead of the results being shown in the bottom pane).

You may have the option to save it as a .rpt file. This is fine if you can open it in Crystal Reports or a similar application. If not, you can change the extension to .csv.

### View the File

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_9.png)

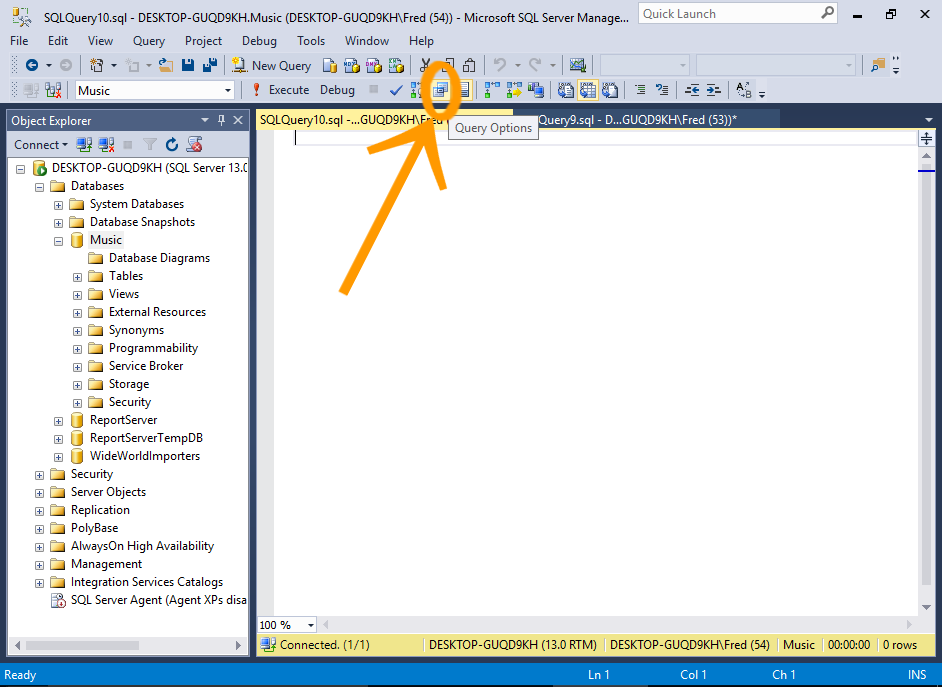
Check the file to verify that its contents are correct, and in the required format.

## Results to File Options

You might have noticed that the above screenshot shows that the text file displays the results in column aligned format (not comma delimited like in the previous file). This can be changed if required.

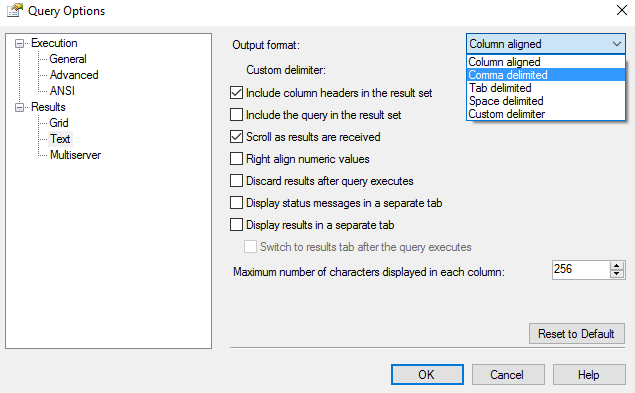
You can use the Query Options dialog box to change various options for when using the Results to File method.

### Launch the Query Options Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_4.png)

Open a new query window, and click the Query Options icon in the toolbar.

### Configure the Desired Options

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_10.png)

Select Results > Text and review/change any options you need to change.

For example, you could change the output format to be comma delimited instead of column aligned. You can also specify whether or not to include headers.

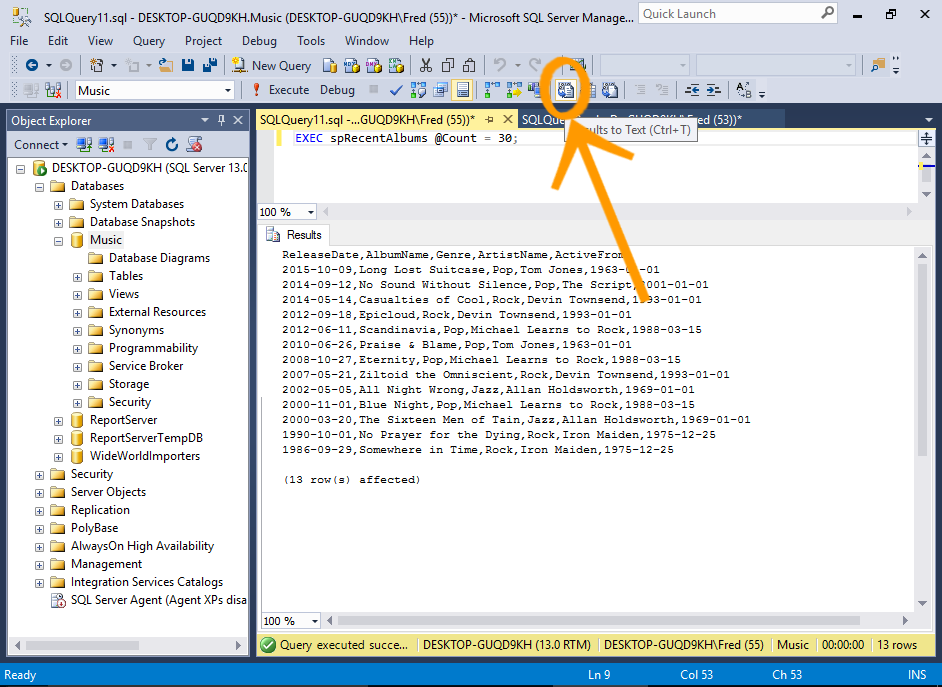
Once configured, click OK.

## Results to Text

You can also use Results to Text for outputting the results.

This can be handy when configuring the query formatting options for Results to File (as it uses the same query options). This will enable you to check the formatting without having to open an external file.

To switch to Results to Text, click the Results to Text icon in the toolbar. You can toggle between output options whenever you need.

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/save_query_to_csv_file_in_sql_server_2016_11.png)

# SQL Server 2016: Create a Login

* [Save Query Results to CSV](https://www.quackit.com/sql_server/sql_server_2016/tutorial/save_query_results_to_csv_file_in_sql_server_2016.cfm)
* [Create a Database User](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_database_user_in_sql_server_2016.cfm)

In SQL Server, you can create a login to allow different users or processes to connect to the SQL Server instance.

As with any client/server database management system, a given instance of SQL Server will typically be accessed by many different users and processes. These could include database developers, database administrators, web applications, corporate CRMs, reporting applications, etc.

Each of these users need a login to connect to SQL Server. A login can be used by one, or a group of users.

When you create a login, you can choose whether it uses Windows Authentication or SQL Server Authentication login.

Windows Authentication

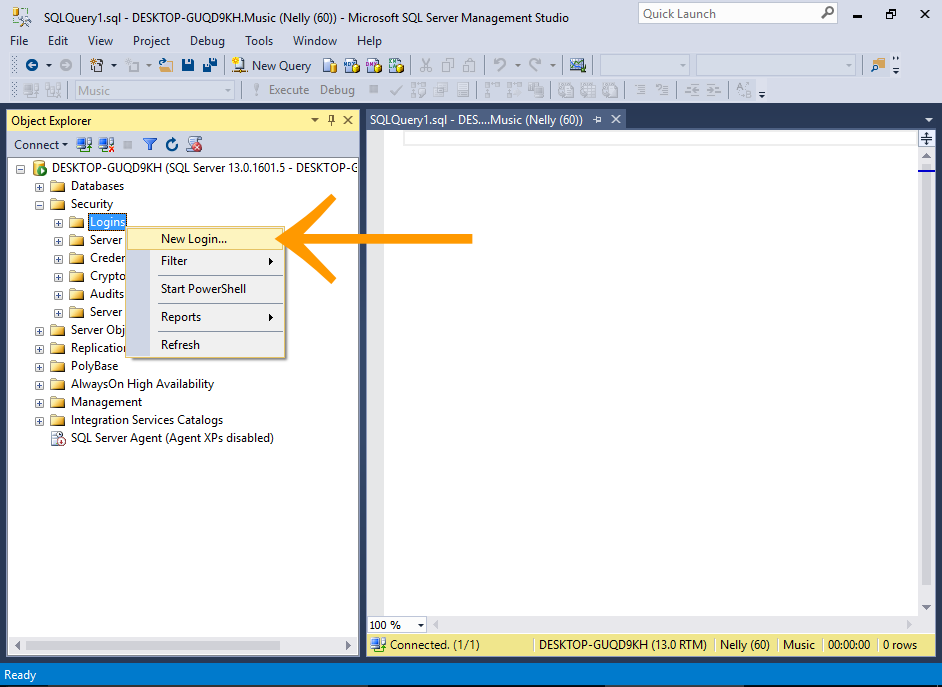
When using Windows authentication, SQL Server validates the account name and password using the Windows principal token in the operating system. Therefore the user's identity is confirmed by Windows. SQL Server does not ask for the password, and does not perform the identity validation.

SQL Server Authentication

When using SQL Server Authentication, logins are created by using SQL Server and they are stored in SQL Server. Users connecting using SQL Server Authentication must provide their login credentials every time they connect.

You can also apply other settings for the login, including adding it to one or more roles, mapping it to database users, and more.

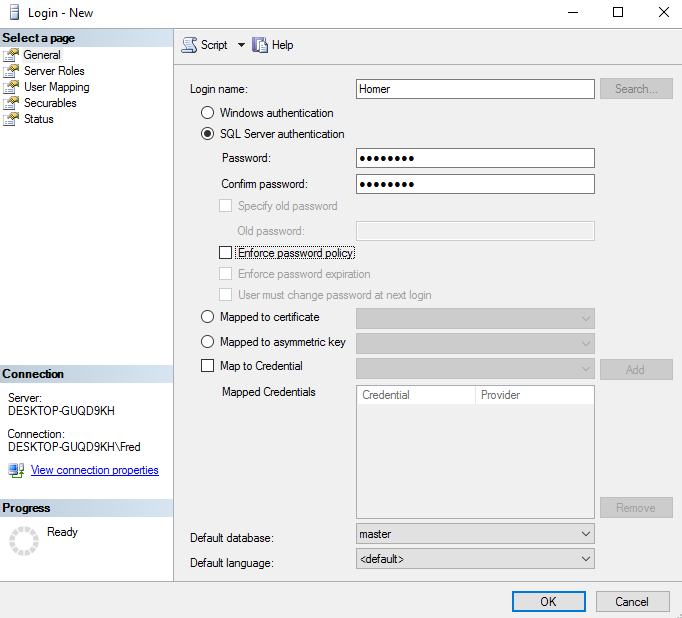
### Launch the New Login Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_1.png)

In the Object Explorer, expand the Security node to reveal its subnodes.

Right-click on the Logins node and select New Login....

### Configure the new Login

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_2.png)

Enter a login name and choose an authentication method.

For this tutorial, use SQL Server authentication and enter a password (and confirm it).

Click OK to create the login and close the dialog box.

For this example, I disabled Enforce password policy and its associated options. I did this to make it easier for demonstration purposes.

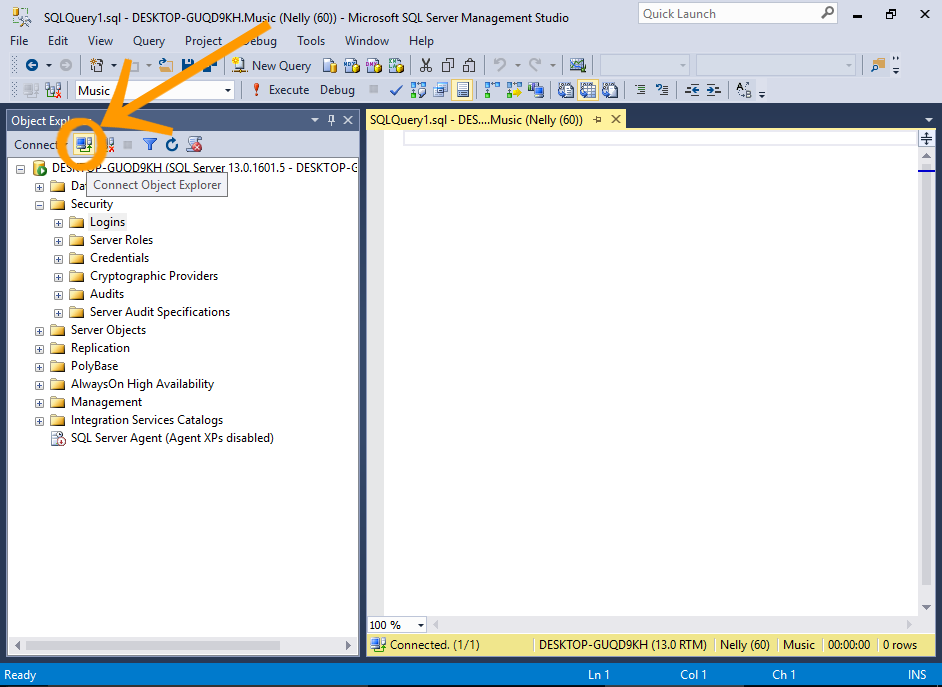
However, it's good practice to enable these options in a real world setting.

Microsoft recommends that you use Windows Authentication whenever possible, as it's more secure than SQL Server Authentication.

## Test the new Login

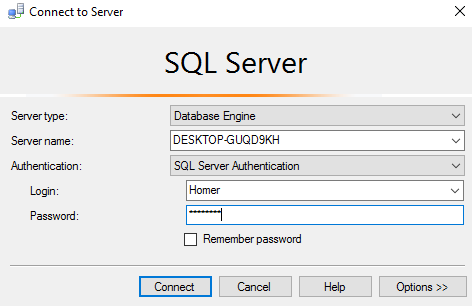
Now, let's test out the new login by using it to connect to SQL Server.

### Open the Connection Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_3.png)

In the Object Explorer, click the little connection icon at the top of the Object Explorer (the one with a tool tip that reads Connect Object Explorer).

### Connect using the New Login

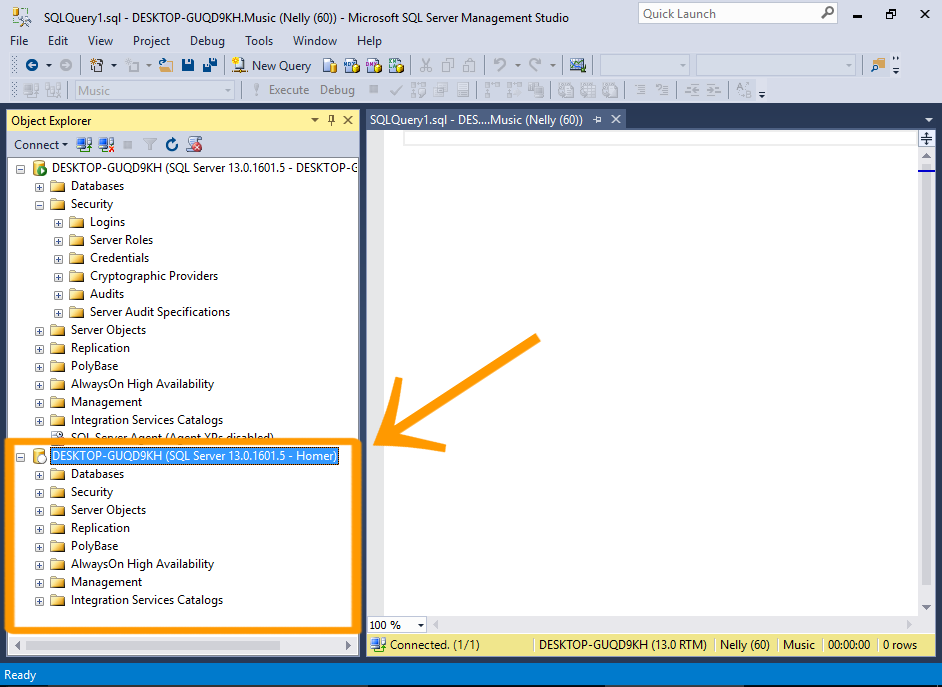
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_4.png)

Select SQL Server Authentication from the Authentication drop-down list.

Enter the login credentials and click Connect.

If you receive a 18456 authentication error when trying to log in, see *Server Authentication Mode* below.

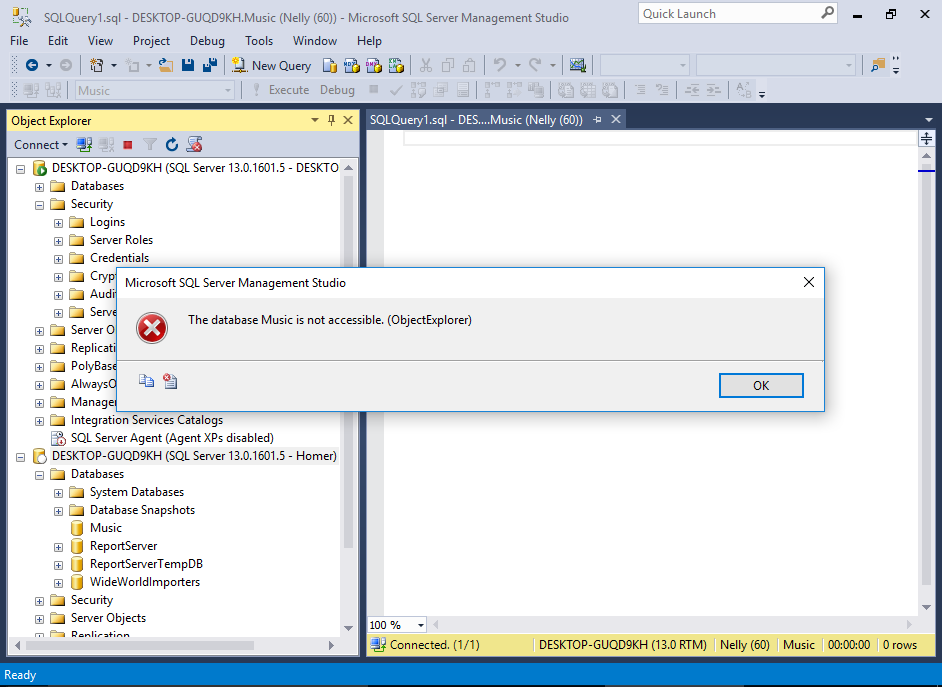
### The Connection

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_5.png)

Observe that the new connection now appears in the Object Explorer (under the other one).

You can see that it has the login name at the end of the connection name.

### Access the Database Objects

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_6.png)

Now try to access the Music database. Navigate to it and try to expand the node so that you can see the tables.

You will receive an error telling you that the database is not accessible.

This is because we haven't assigned any database users to this login yet. We'll do that [next](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_database_user_in_sql_server_2016.cfm).

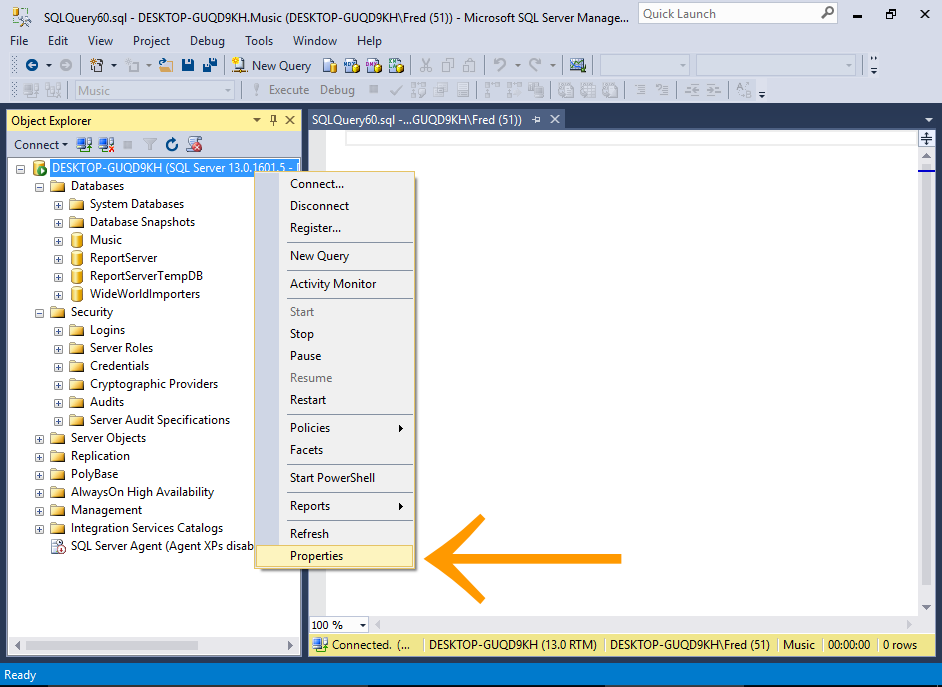
## Server Authentication Mode

Before a user can log in using SQL Server Authentication, the server must support this mode.

If you receive a 18456 authentication error when trying to log in, check the server's authentication mode. You need to ensure it is configured to support SQL Server Authentication.

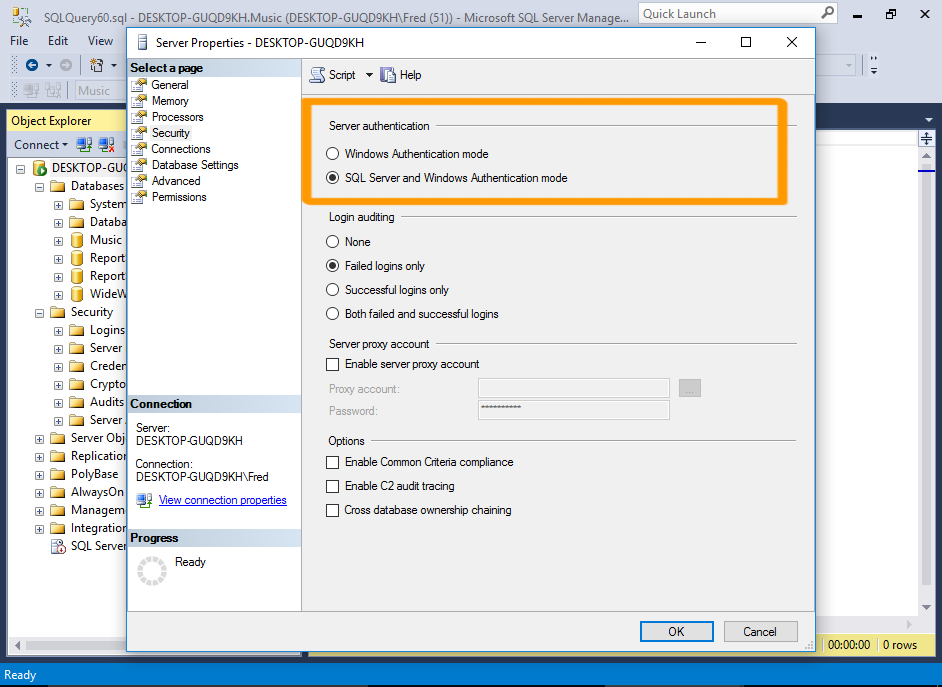
Here's how to change the authentication mode in SQL Server.

### Launch the Server Properties Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_7.png)

In the Object Explorer, right-click on the server node and select Properties.

### Change the Authentication Mode

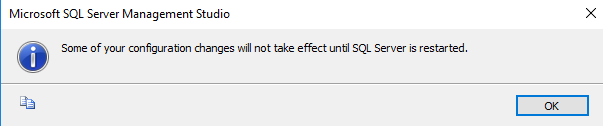
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_8.png)

Click on Security in the left menu.

Under the Server authentication heading, select SQL Server and Windows Authentication mode.

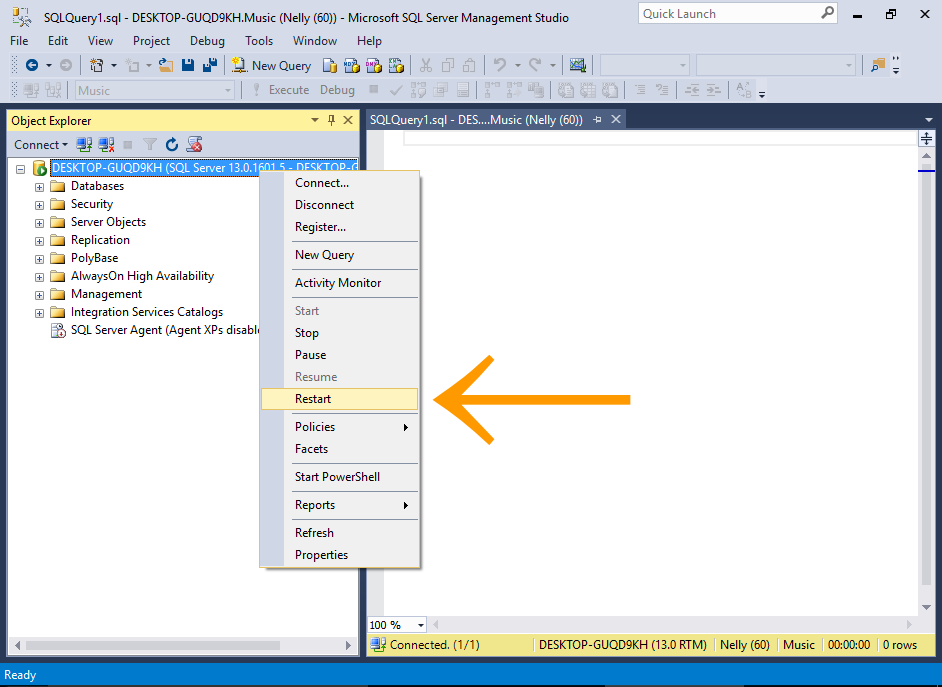
Click OK.

### Acknowledge Warning

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_8a.png)

Click OK again if you get a warning that the changes won't take effect until the server has been restarted.

### Restart the Server

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_login_in_sql_server_2016_9.png)

In the Object Explorer, right-click on the server and select Restart from the contextual menu.

Once the server has been restarted, you should be able to log in using SQL Server Authentication.

## Logins vs Users

Logins and users are two different things in SQL Server.

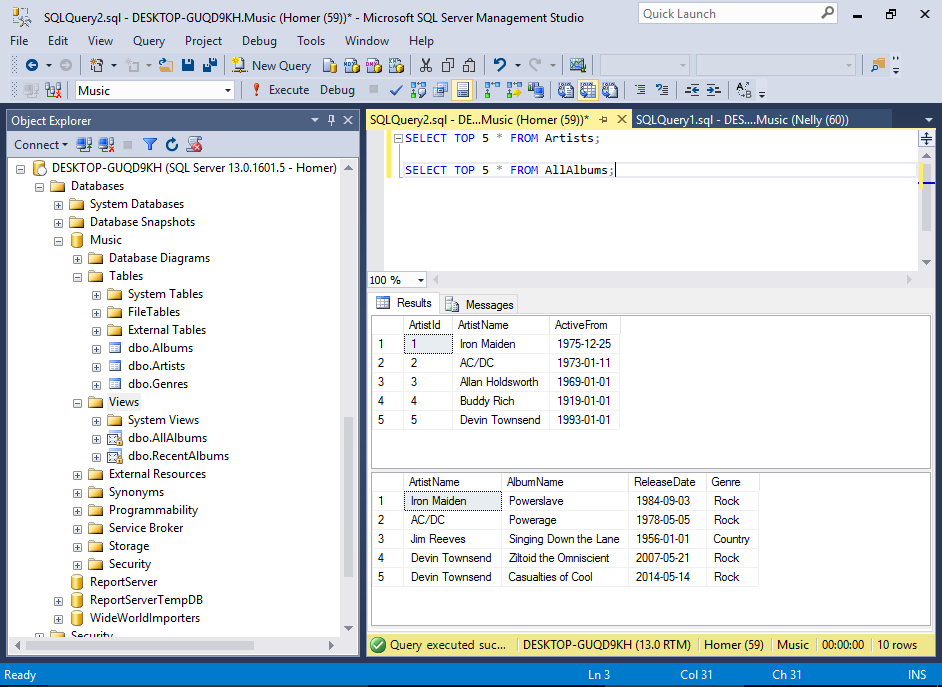
* A login grants the principal access to the server.
* A user grants a login access into a particular database.

So the above login is allowed to connect to SQL Server. However, at this stage, none of the databases on the server have a user associated with this login.

We'll do that next. We'll [create a database user, and associate it with the above login](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_database_user_in_sql_server_2016.cfm).

1. Also refresh the Views node in the same way.

### The Result

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/create_a_database_user_in_sql_server_2016_7.png)

We can now see that all the user tables and views have now appeared in the Object Explorer.

We can also retrieve data from the tables and views using a [SELECT](https://www.quackit.com/sql/tutorial/sql_select.cfm) statement.

# SQL Server 2016: Backup a Database

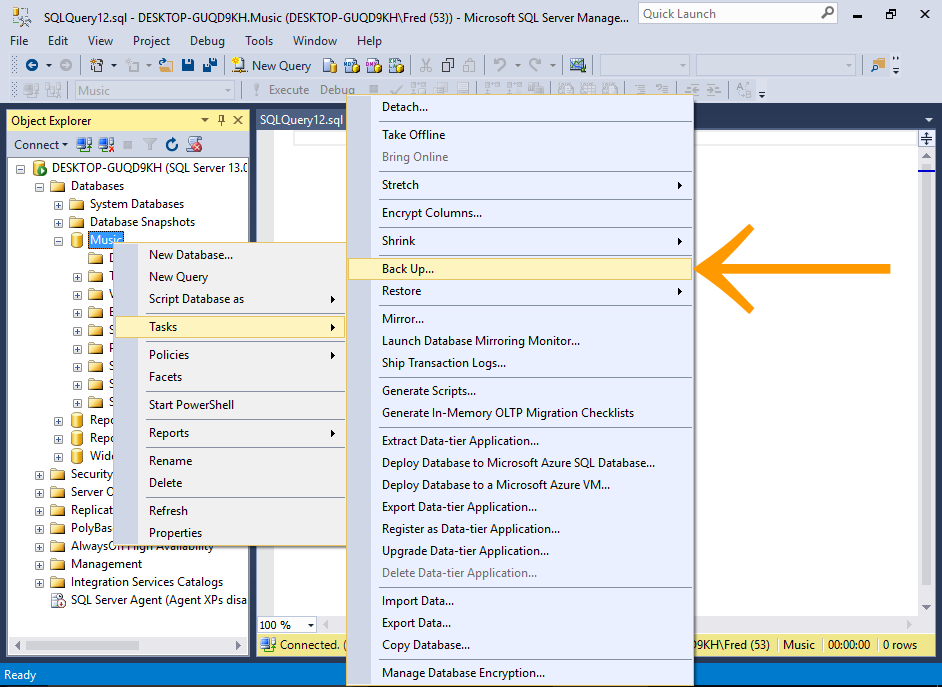
* [Create a Database User](https://www.quackit.com/sql_server/sql_server_2016/tutorial/create_a_database_user_in_sql_server_2016.cfm)
* [Restore a Database](https://www.quackit.com/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016.cfm)

SQL Server provides an easy way to create a backup of a database. Backups can be done either with Transact-SQL, PowerShell, or via the SQL Server Management Studio GUI.

Here, I'll demonstrate how to create a backup using the SQL Server Management System GUI, then using Transact-SQL, then finally, with SQL Server Powershell.

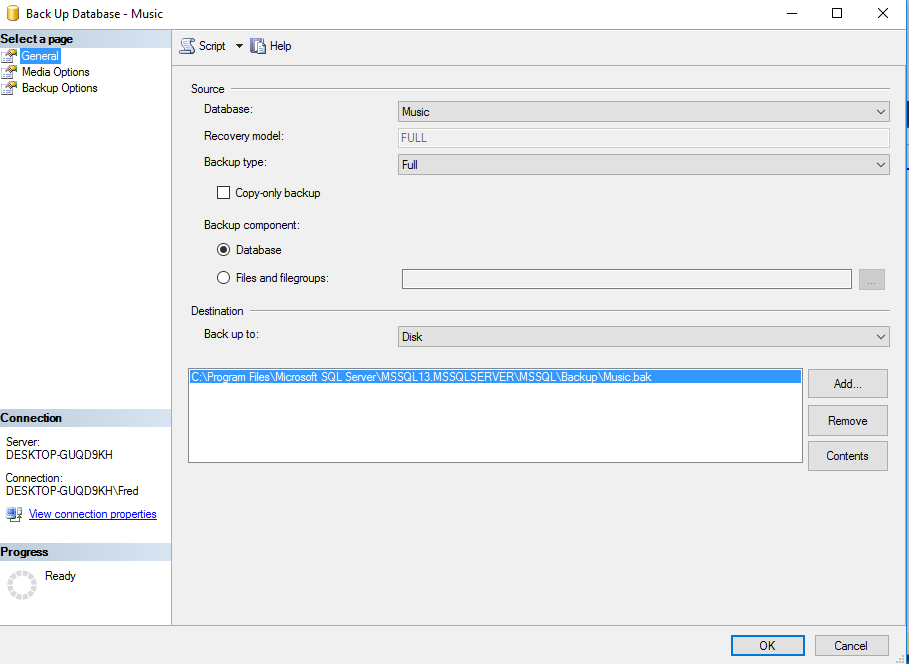
## Create a Backup via the GUI

### Launch the Back Up Database Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016_1.png)

In the Object Explorer, right-click on the database you'd like to back up, and select Tasks > Back Up... from the contextual menu.

### Review the Backup Settings

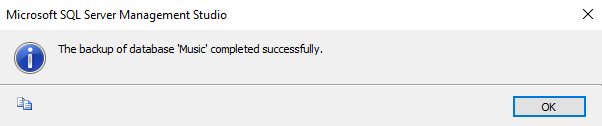
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016_2.png)

This dialog box gives you the opportunity to change any of the settings if required.

For our example, leave it at the default settings and click OK to create the backup.

You can change the database here if you accidentally selected the wrong one at the previous step.

### Backup Complete

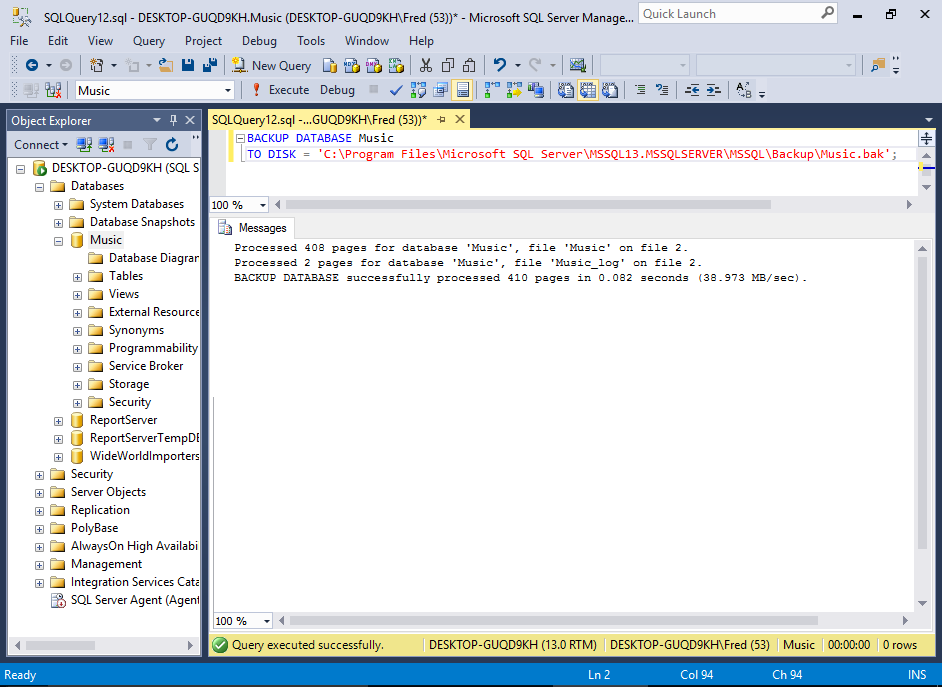
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016_3.png)

You'll receive a message when the backup is complete.

Click OK to close the message and dialog box.

The backup file will now be located at the specified location.

## Backup a Database Using Transact-SQL

* [](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016_4.png)

You can perform the same backup as above using SQL.

To do this, open a new query window and execute a BACKUP statement.

The BACKUP statement accepts various options (just like the GUI option), but you can also run a simple backup with a minimum of code.

### Sample Code

Below is an example of a simple backup script that specifies the database to backup, and the location to back it up to.

After running this code, the backup file will be located at the specified location.

BACKUP DATABASE Music

TO DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Backup\Music.bak';

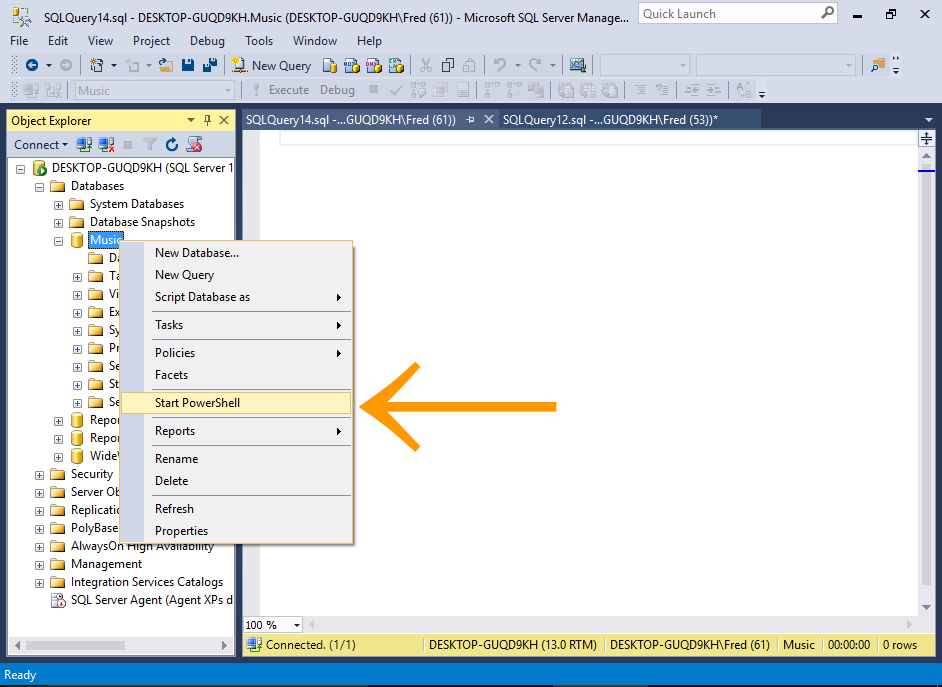
You can see the full syntax for the BACKUP statement at the [Microsoft website](https://msdn.microsoft.com/en-us/library/ms186865.aspx).

## Backup a Database using PowerShell

SQL Server 2016 supports Windows PowerShell, which is a scripting shell, typically used to automate administration and deployment tasks.

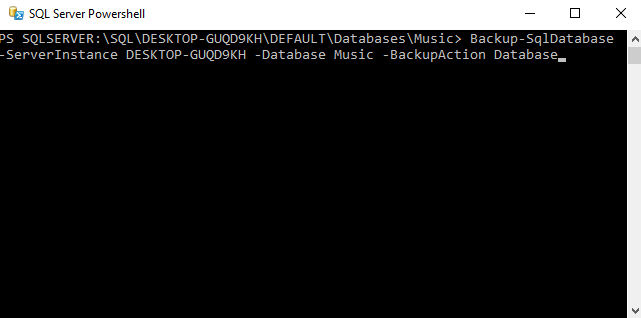
The PowerShell language supports more complex logic than Transact-SQL scripts, which gives you the ability to build more sophisticated scripts for your backups and other tasks.

### Open PowerShell

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016_5.png)

Right-click on the database and select Start Powershell.

### Run the Backup Command

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016_6.png)

Enter the command for creating the backup and press Enter (or Return, depending on your keyboard).

The backup will run immediately.

#### Sample Code

The following code will create a backup just like the previous examples. Just replace MyServer with the name of your server.

After running this code, the backup file will be located at the default location.

Backup-SqlDatabase -ServerInstance MyServer -Database Music

You can also specify a location

Backup-SqlDatabase -ServerInstance MyServer -Database Music -BackupFile 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Backup\Music.bak'

You can also specify -BackupAction Database to explicitly state that it is a full backup. However, this is the default option.

You can see the full documentation for the Backup-SqlDatabase command at the [Microsoft website](https://technet.microsoft.com/en-us/library/mt683378.aspx).

## Overwriting Backup Files

If you ran all of the above examples exactly as they are, you might have noticed that each time you ran it, the file size of the backup file increased.

This is because each successive backup is adding itself to the existing file.

It's doing this because we are using the same file name, and we have not explicitly specified that each backup should overwrite any existing file.

There's an option that allows you to overwrite the existing file.

* Using the **GUI**, click on Media Options in the left menu of the Back Up Database dialog box, and select Overwrite all existing backup sets in the Overwrite Media section.
* Using **SQL** add WITH INIT to the SQL statement.
* Using **Powershell**, add -Initialize to the command.

## Saving Backup Files

However, it's often a good idea to create a full backup with a unique filename (typically including the date in the file name). Having a unique filename will mean each backup will be a separate file.

Also, depending on the size of your database, and how much new data is being entered into it, you may wish to supplement your full backups with differential backups. A differential backup captures only the data that has changed since the most recent full backup.

# SQL Server 2016: Restore a Database

* [Backup a Database](https://www.quackit.com/sql_server/sql_server_2016/tutorial/backup_a_database_in_sql_server_2016.cfm)
* [More Database Tutorials](https://www.quackit.com/database/)

As with database backups, in SQL Server 2016, you can restore a database by using either Transact-SQL, PowerShell, or via the SQL Server Management Studio GUI.

Here, I'll demonstrate how to restore a database using the SQL Server Management System GUI, then using Transact-SQL.

## Restore a Database via the GUI

In this example, we're going to restore a backup of a brand new database called WideWorldImporters.

The WideWorldImporters database is a sample database provided by Microsoft's SQL Server Team to demonstrate how SQL Server's features can be used in a real-world scenario.

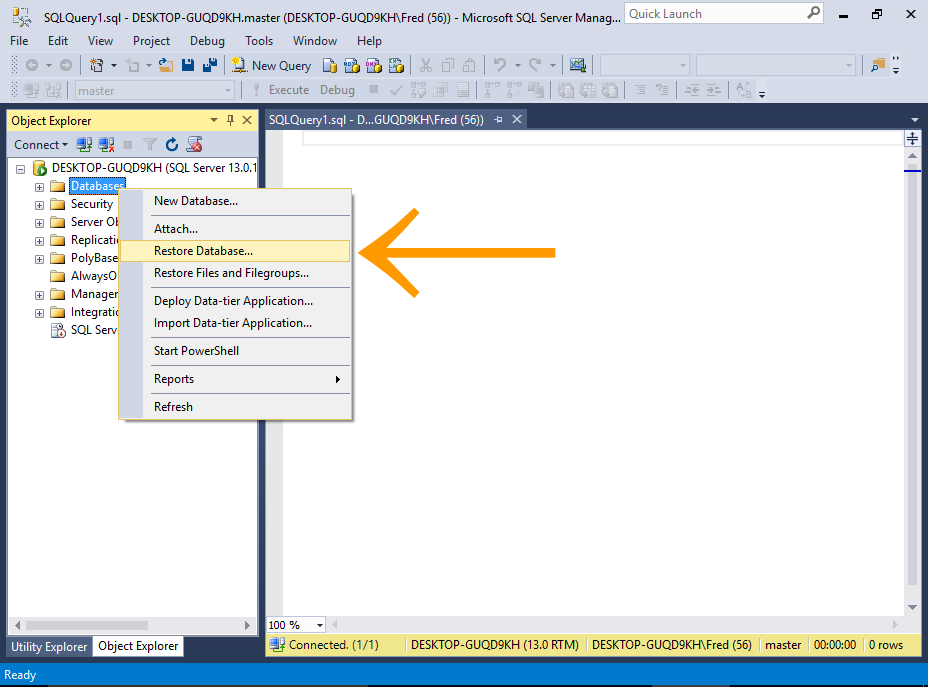
This process will create a new database called WideWorldImporters. The database will contain various objects (such as tables, views, stored procedures, etc). It will also contain sample data, so there's no need to add your own.

### Download the Database Backup File

Before you start, [**download the database**](https://github.com/Microsoft/sql-server-samples/releases/tag/wide-world-importers-v1.0) (from GitHub).

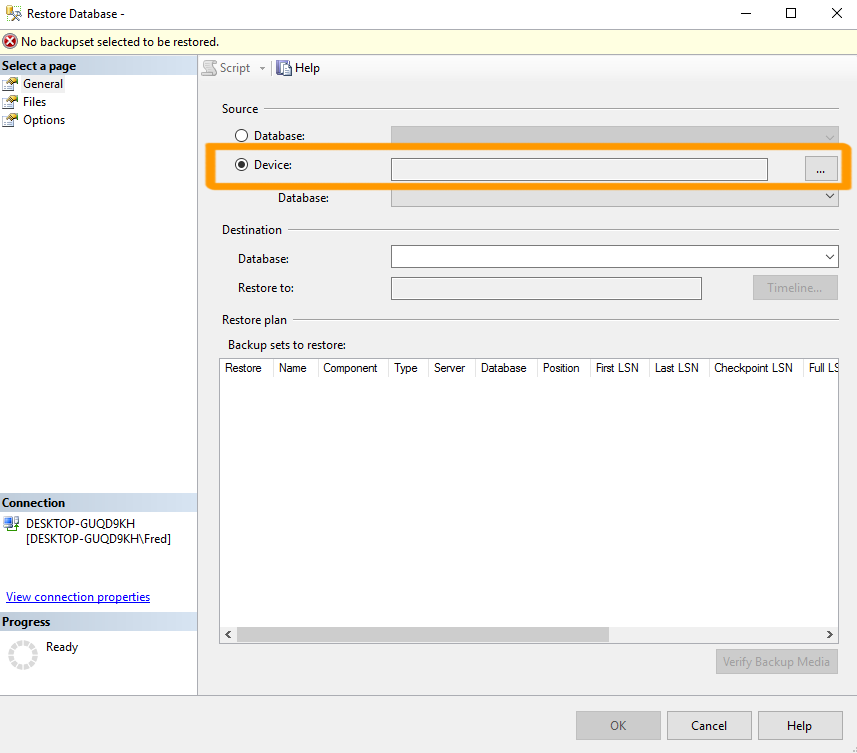
* Download the WideWorldImporters-Full.bak file if you're using the Evaluation, Developer, or the Enterprise Edition of SQL Server.
* Or download the WideWorldImporters-Standard.bak file if you're using SQL Server Standard Edition.

### Launch the Restore Database Dialog Box

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_1.png)

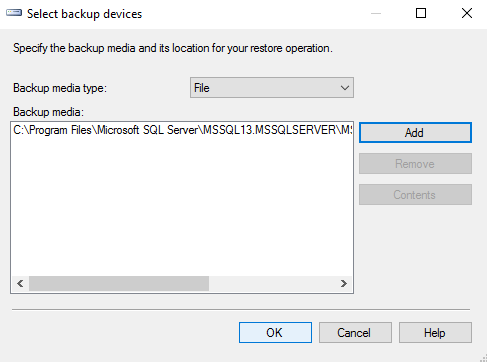
In the Object Explorer, right-click on the Databases node and select Restore Database... from the contextual menu.

### Select the Backup File

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_2.png)

Under the Source heading, select Device and click the ellipses button (...) to launch the Select backup devices dialog box.

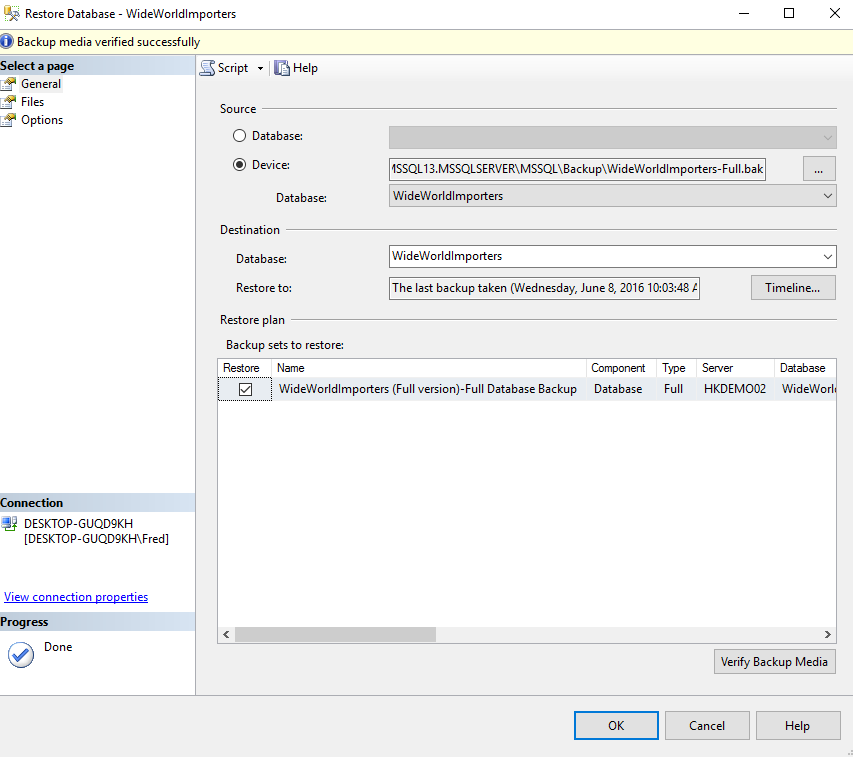
### Select the Backup File

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_3.png)

Ensuring that File is selected, click Add to browse to the database .bak file and add it.

Once you've added the backup file and it's listed under Backup media:, click OK to exit this dialog box.

### Check the Settings

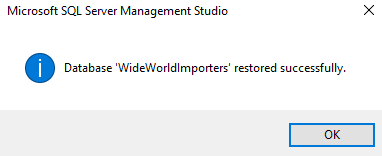
[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_4.png)

Various fields in the Restore Database dialog box will be populated based on the backup file that you selected.

Click OK to restore the database.

You can (optionally) click Verify Backup Media to test that there's no problems with the backup file before running the restore process.

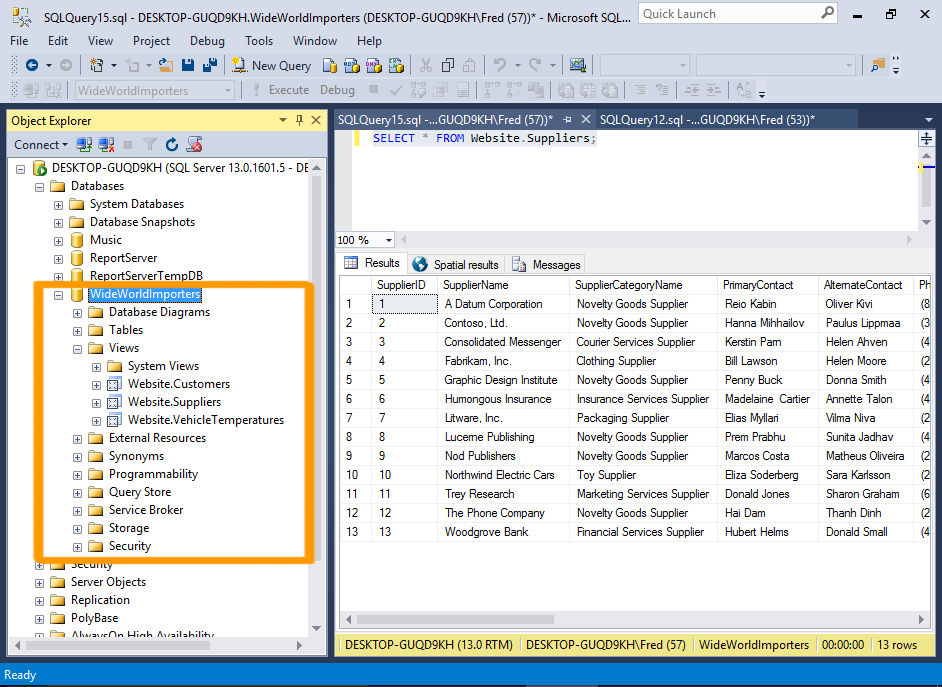
### Success Message

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_5.png)

You will receive a message informing you that the database was successfully restored.

Click OK to finish.

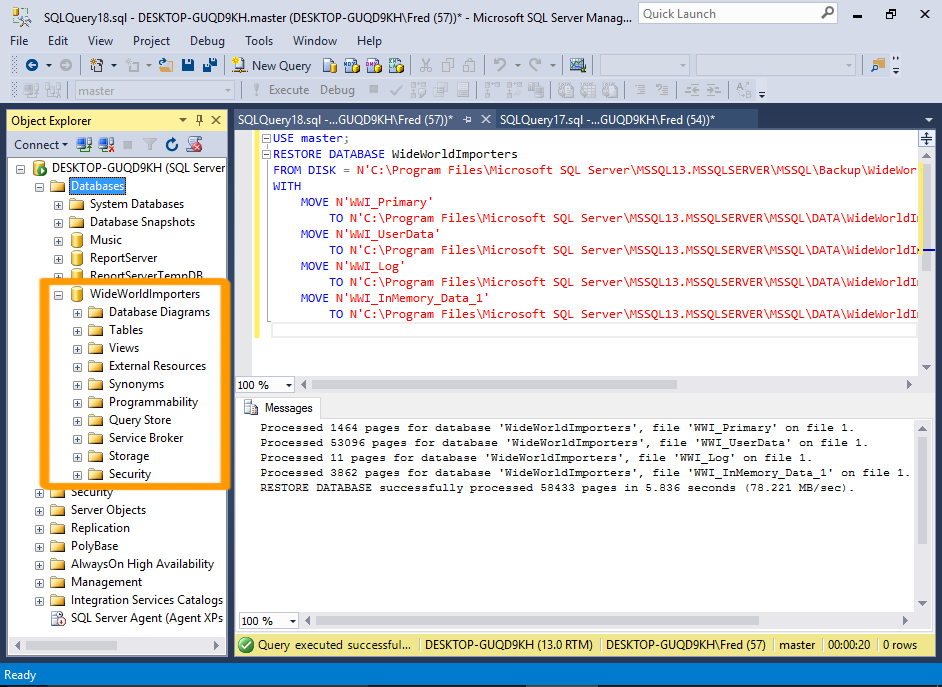
### Check the Database

[](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_6.png)

Navigate to the WideWorldImporters database and expand its nodes to review it's various objects such as tables, views, stored procedures, etc.

You can start working with this database immediately. For example, running SELECT \* FROM Website.Suppliers; will return a list of suppliers using the Website.Suppliers view.

## Restore a Database Using Transact-SQL

* [](https://www.quackit.com/pix/sql_server/sql_server_2016/tutorial/restore_a_database_in_sql_server_2016_7.png)

You can perform the same database restore as above using SQL.

To do this, open a new query window and execute a RESTORE statement.

The RESTORE statement accepts various options (just like the GUI option), but you can also run a simple restore with a minimum of code.

### Sample Code

Below is an example of a simple restore script that specifies the database file to restore, and the destination database.

It also specifies where the data and log files will be located.

After running this code, the WideWorldImporters database will be created.

Delete the existing database first by running the following code:

USE master;

DROP DATABASE WideWorldImporters;

You can also delete a database via Object Explorer by right-clicking on the database name and selecting Delete from the contextual menu.

Now check that the database is no longer listed in the Object Explorer. You might have to right-click on Databases and click Refresh before it disappears.

Once you've confirmed that the database no longer exists on your server, run the following script to bring it back:

USE master;

RESTORE DATABASE WideWorldImporters

FROM DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Backup\WideWorldImporters-Full.bak'

WITH

MOVE N'WWI\_Primary'

TO N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\WideWorldImporters.mdf',

MOVE N'WWI\_UserData'

TO N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\WideWorldImporters\_UserData.ndf',

MOVE N'WWI\_Log'

TO N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\WideWorldImporters.ldf',

MOVE N'WWI\_InMemory\_Data\_1'

TO N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\WideWorldImporters\_InMemory\_Data\_1';

#### Display a Progress Message

You can use the STATS option to receive an update on the progress of the restore process.

For example, STATS=10 will result in 10 percent processed, 20 percent processed, etc as the database is being restored.

You can see the full syntax for the RESTORE statement at the [Microsoft website](https://msdn.microsoft.com/en-us/library/ms186858.aspx).

You can now use this database to try out different things. Feel free to expermiment with it. Query data, delete data, drop objects, anything. After all, if you completely mess it up, you can always restore the database from the backup file.