Practice 01. SQL Server 2012 Introduction

Outline

I. MSSQL Server 2012 Installation

Step 1 − Download the Evaluation Edition from

http://www.microsoft.com/download/en/details.aspx?id=29066

Once the software is downloaded, the following files will be available based on your download (32 or 64 bit) option.

ENU\x86\SQLFULL_x86_ENU_Core.box

ENU\x86\SQLFULL_x86_ENU_Install.exe

ENU\x86\SQLFULL_x86_ENU_Lang.box

OR

ENU\x86\SQLFULL_x64_ENU_Core.box

ENU\x86\SQLFULL_x64_ENU_Install.exe

ENU\x86\SQLFULL_x64_ENU_Lang.box

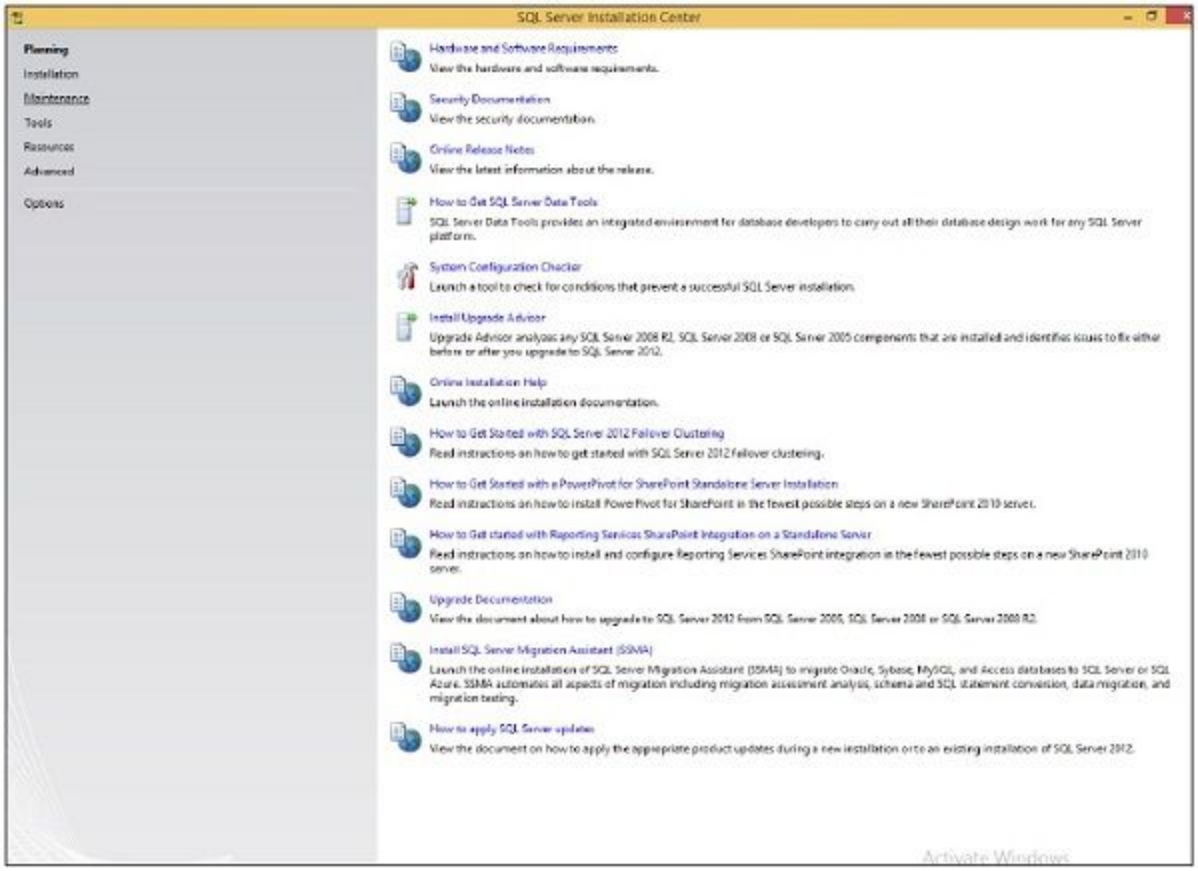
Note − X86 (32 bit) and X64 (64 bit)

Step 2 − Double-click the “SQLFULL_x86_ENU_Install.exe” or “SQLFULL_x64_ENU_Install.exe”, it will extract the required files for installation in the“SQLFULL_x86_ENU” or “SQLFULL_x86_ENU” folder respectively.

Step 3 − Click the “SQLFULL_x86_ENU” or “SQLFULL_x64_ENU_Install.exe” folder and double-click “SETUP” application.

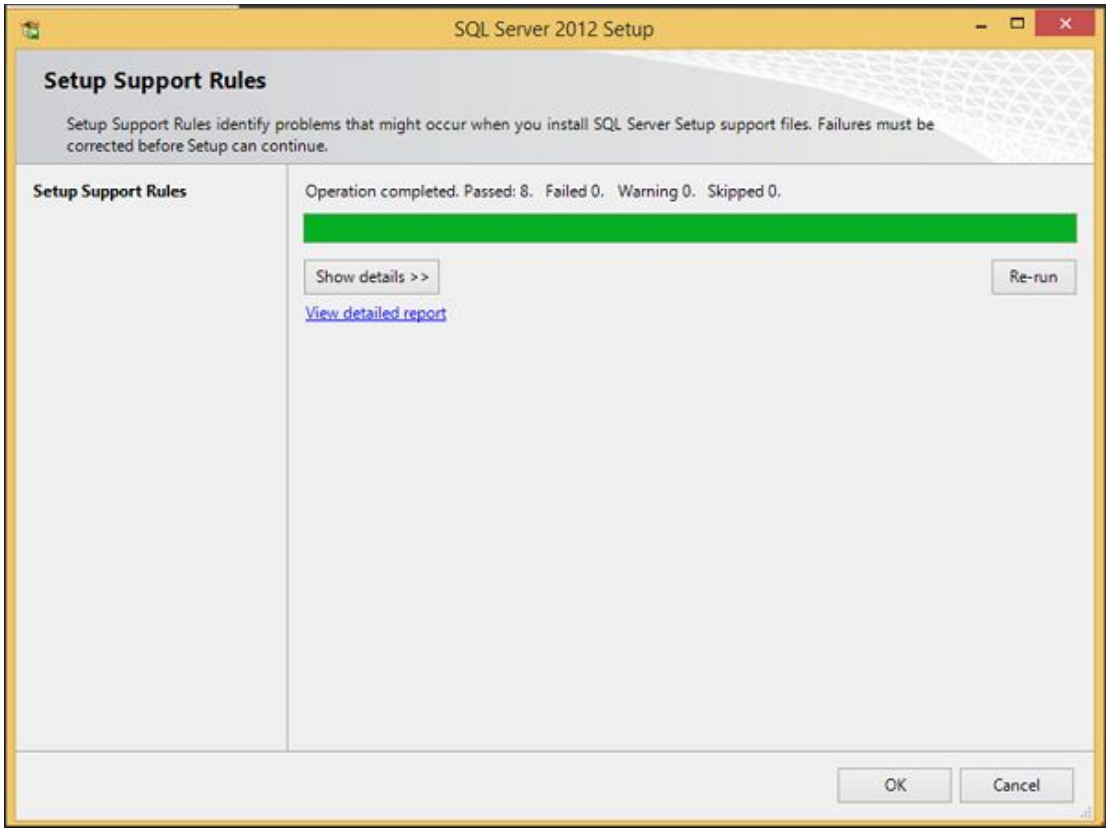
For understanding, here we have used SQLFULL_x64_ENU_Install.exe software.

Step 4 − Once we click on 'setup' application, the following screen will open.

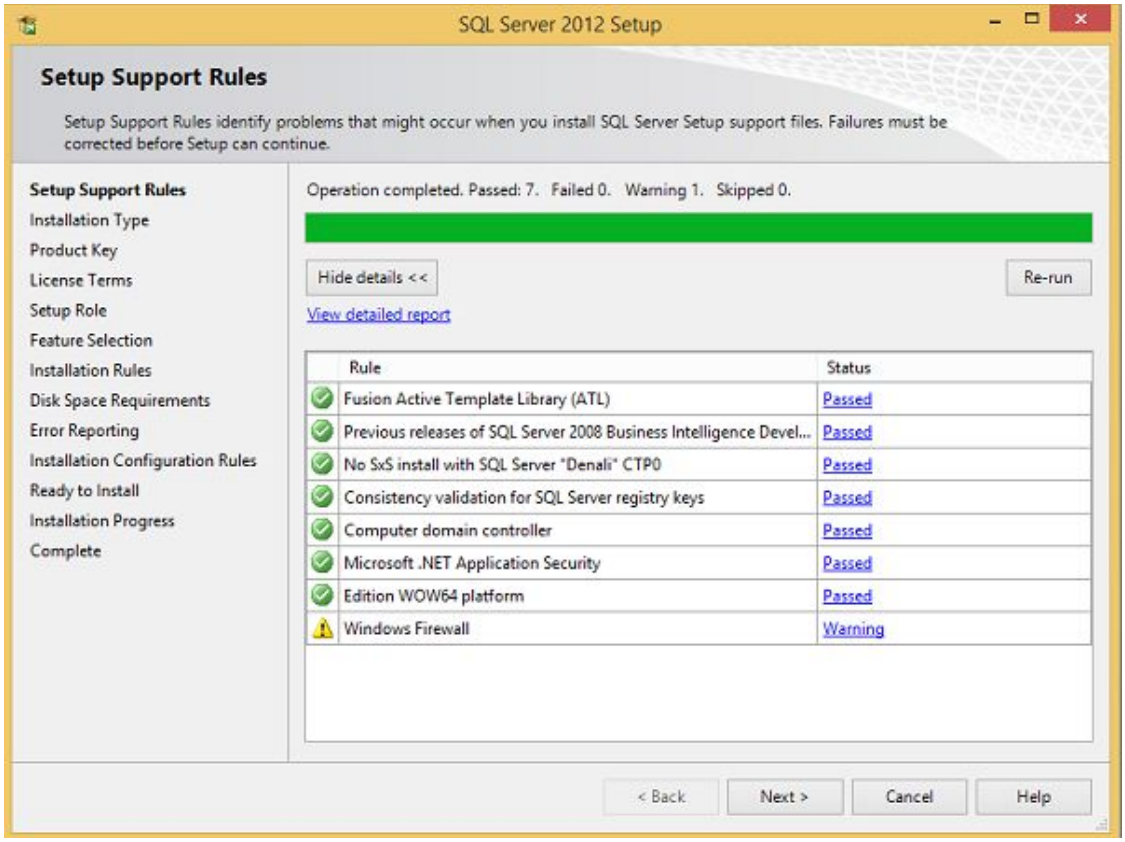


Step 5 − Click Installation which is on the left side of the above screen.

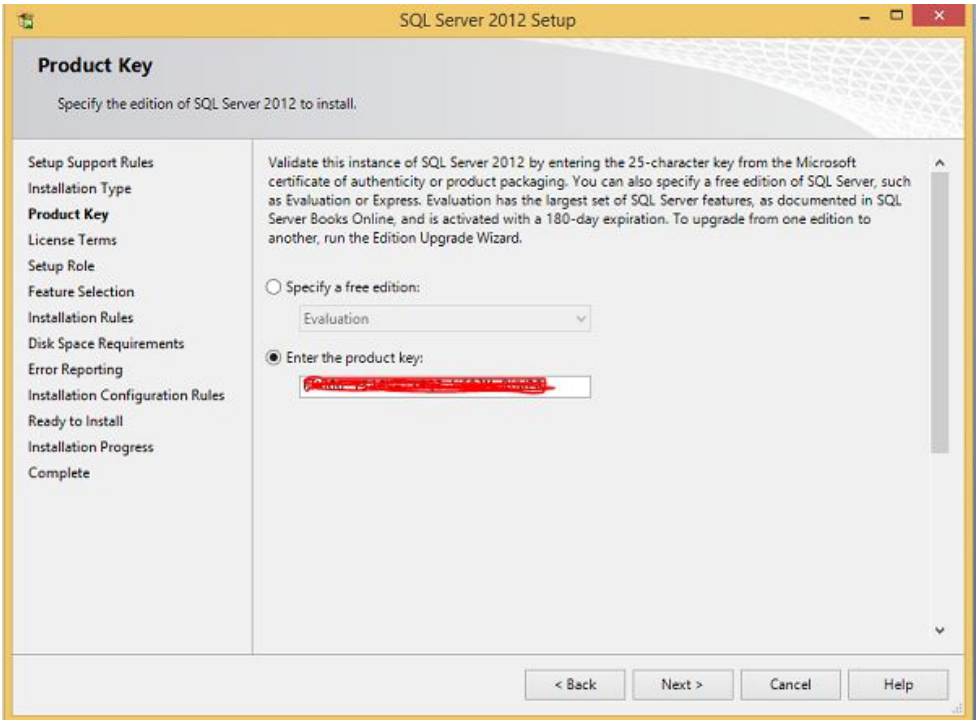
Step 6 − Click the first option of the right side seen on the above screen. The following screen will open.



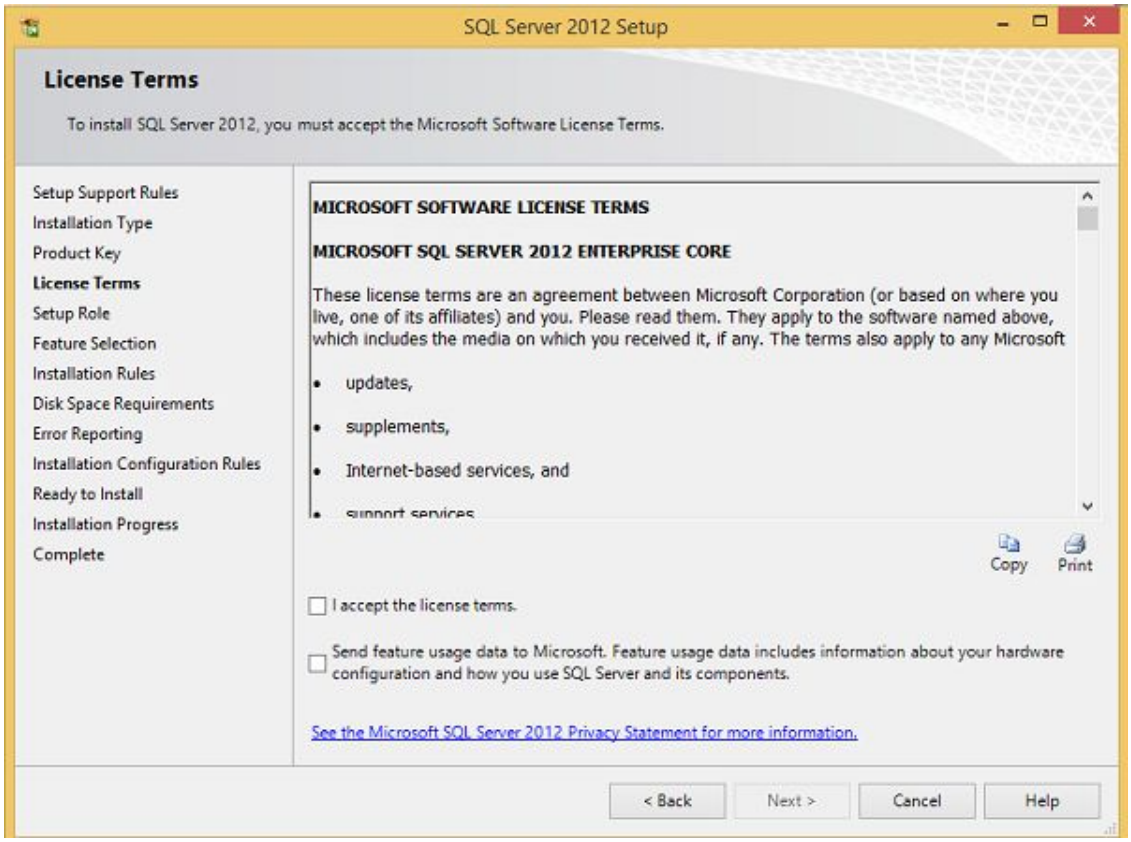
Step 7 − Click OK and the following screen pops up.



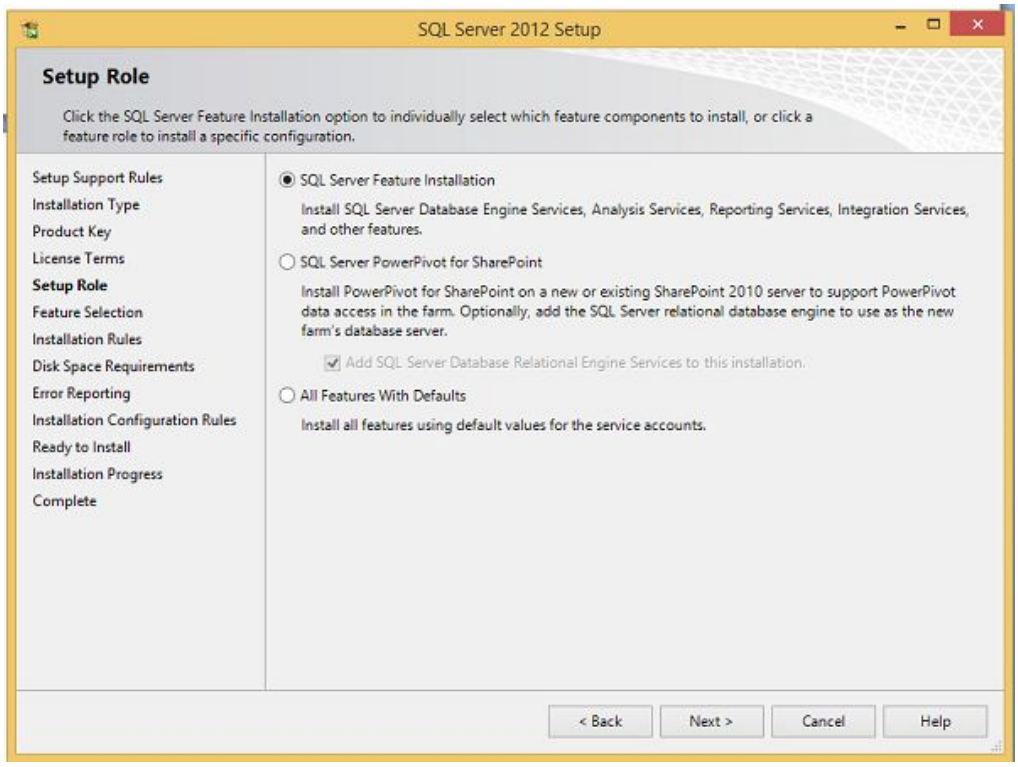
Step 8 − Click Next to get the following screen.



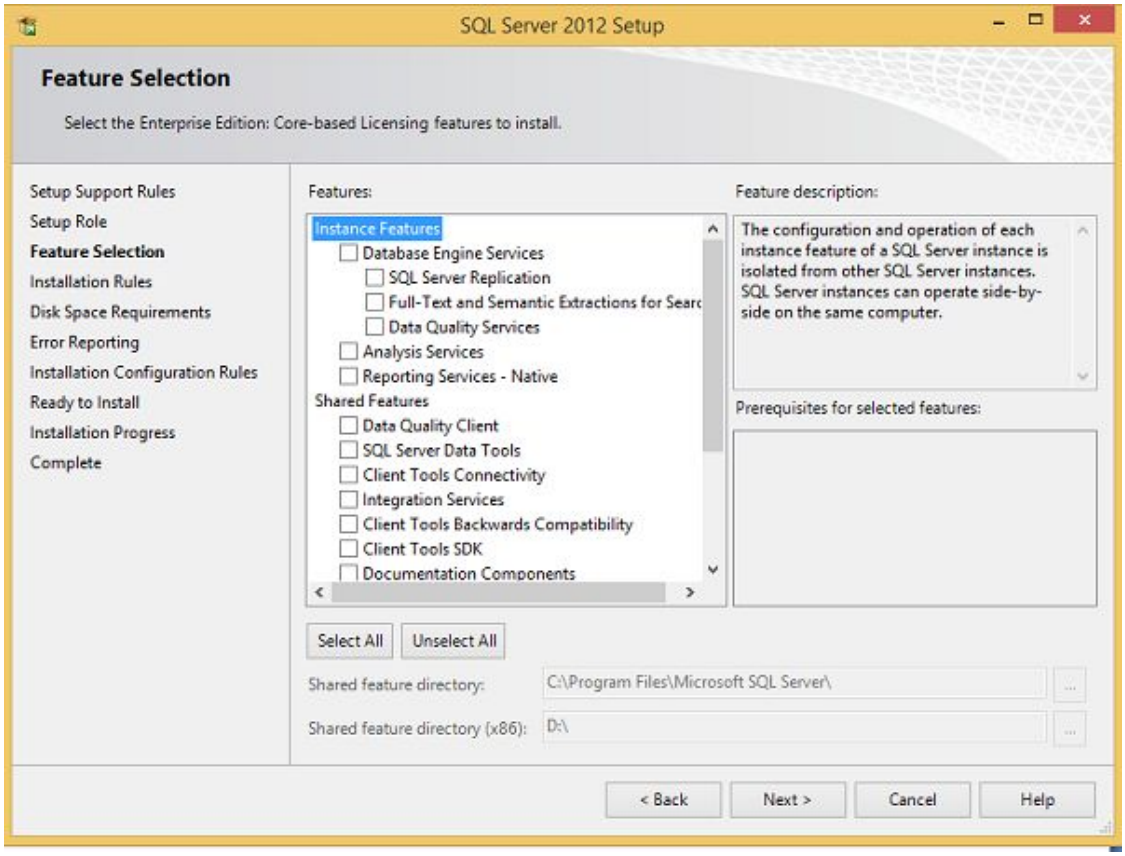
Step 9 − Make sure to check the product key selection and click Next.



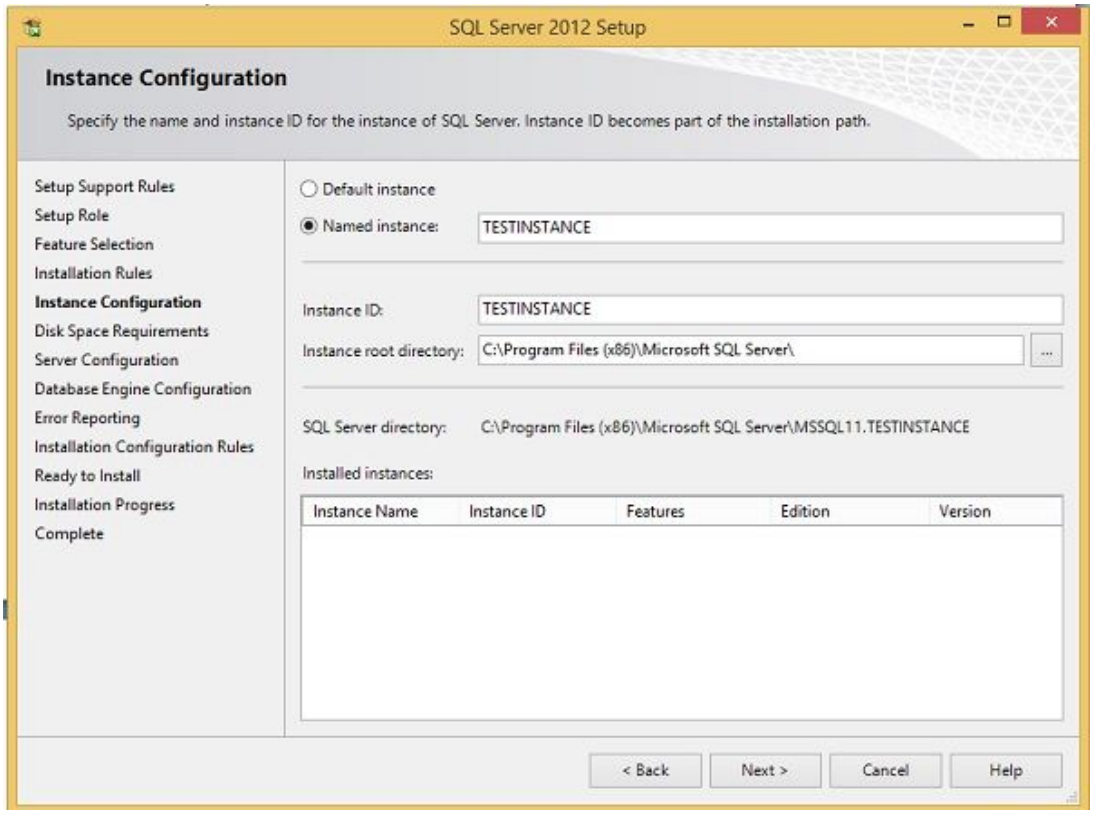
Step 10 − Select the checkbox to accept the license option and click Next.



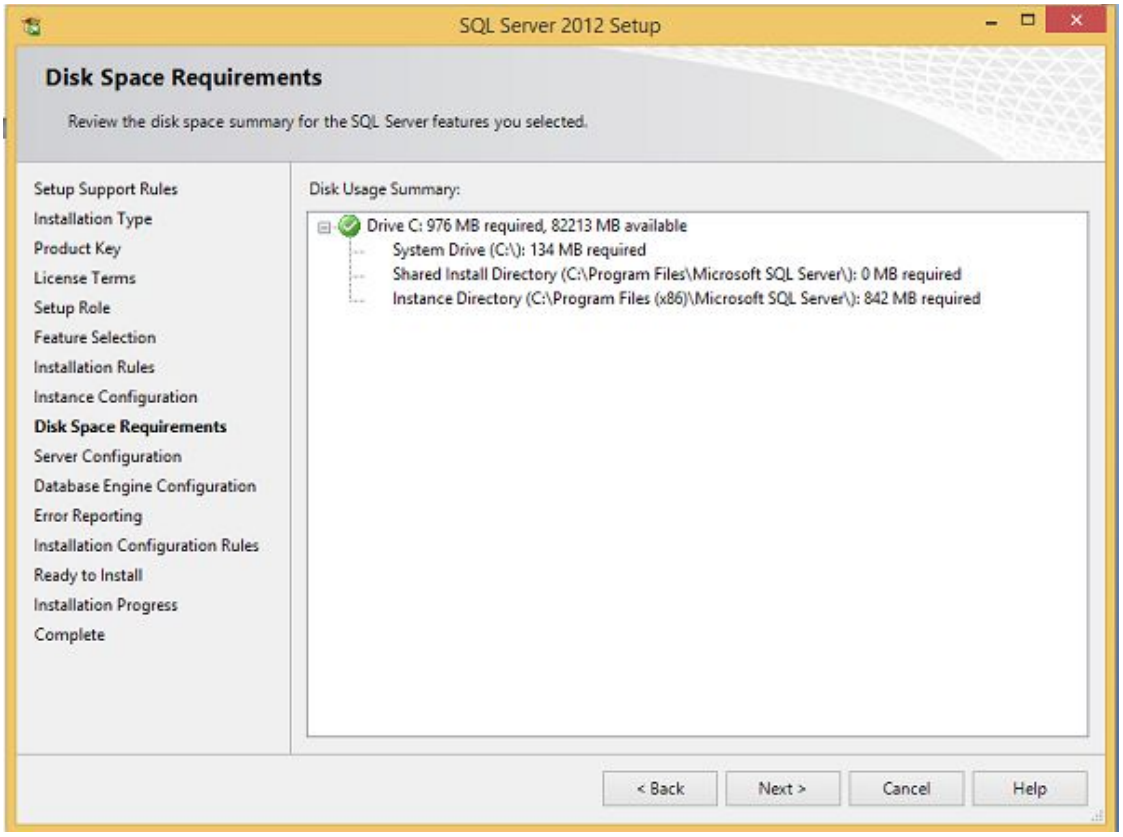
Step 11 − Select SQL Server feature installation option and click Next.



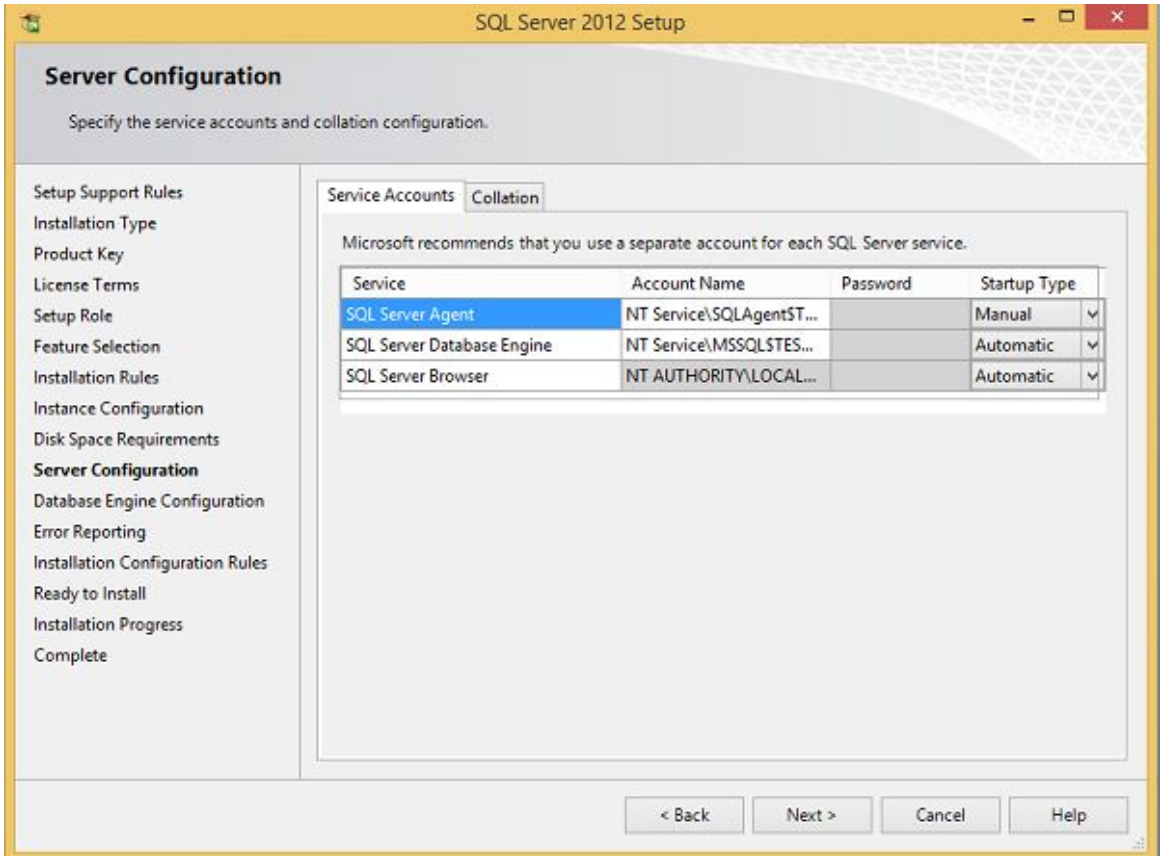
Step 12 − Select Database engine services checkbox and click Next.



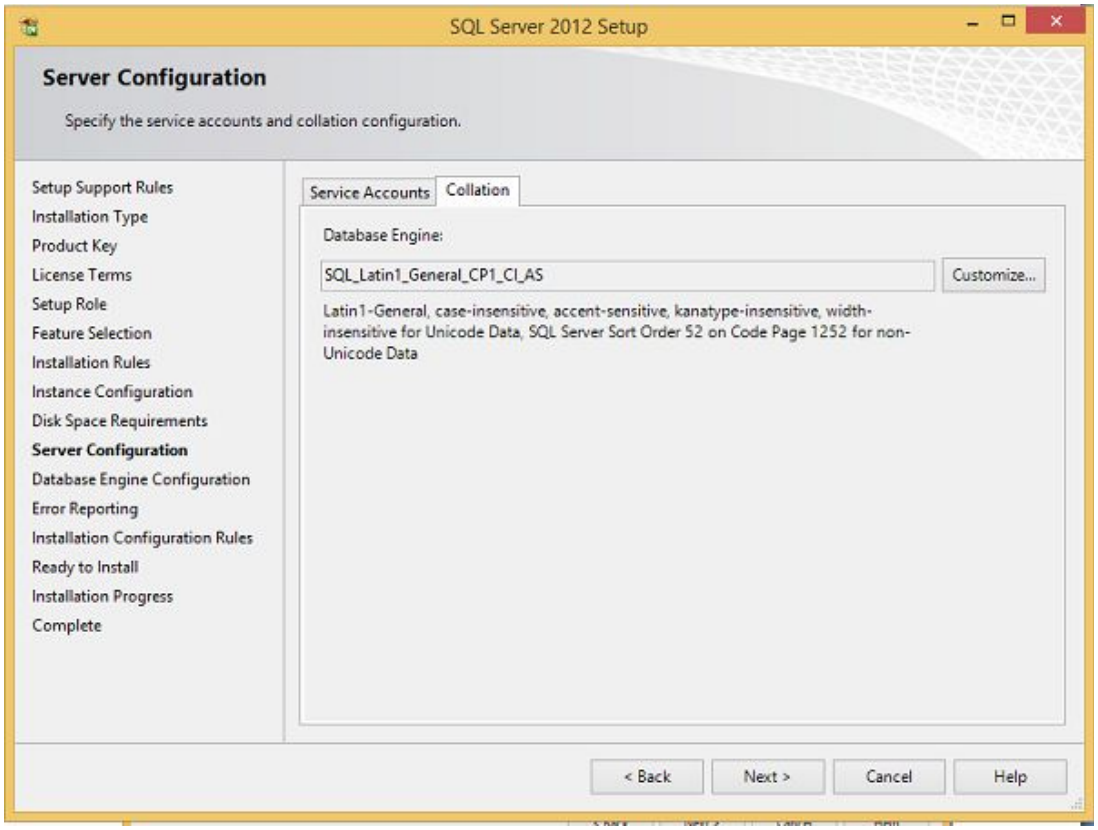
Step 13 − Enter the named instance (here I used TestInstance) and click Next.



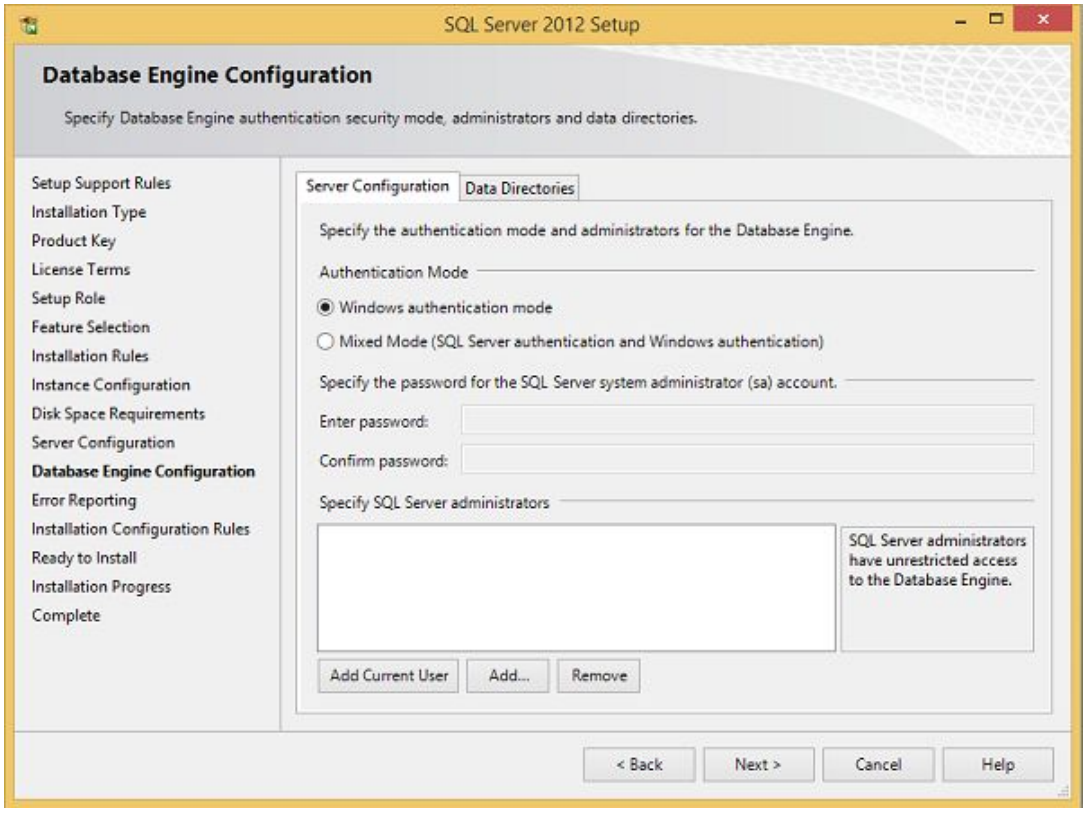
Step 14 − Click Next on the above screen and the following screen appears.



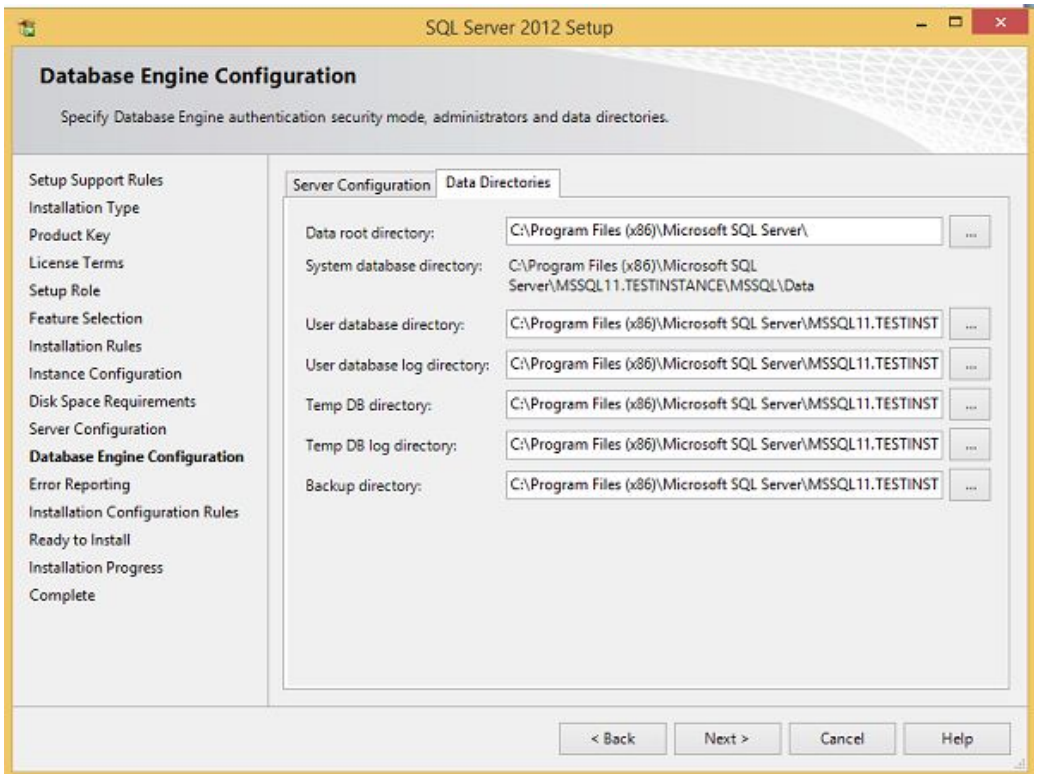
Step 15 − Select service account names and start-up types for the above listed services and click Collation.



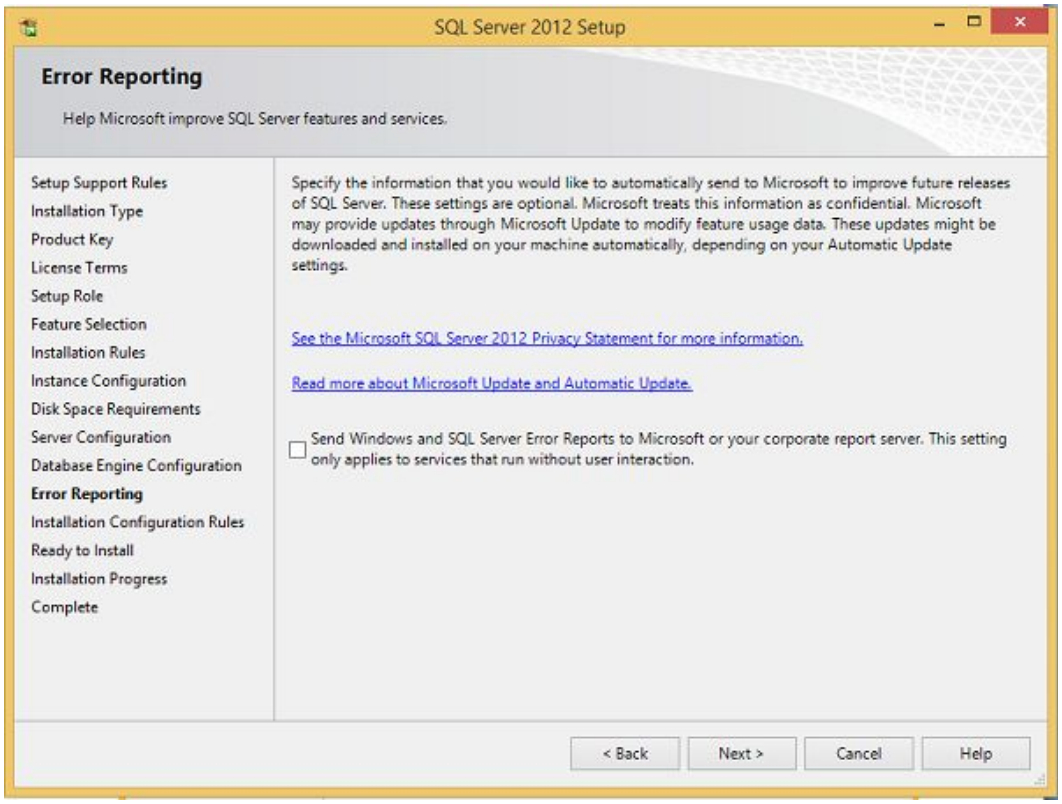
Step 16 − Make sure the correct collation selection is checked and click Next.



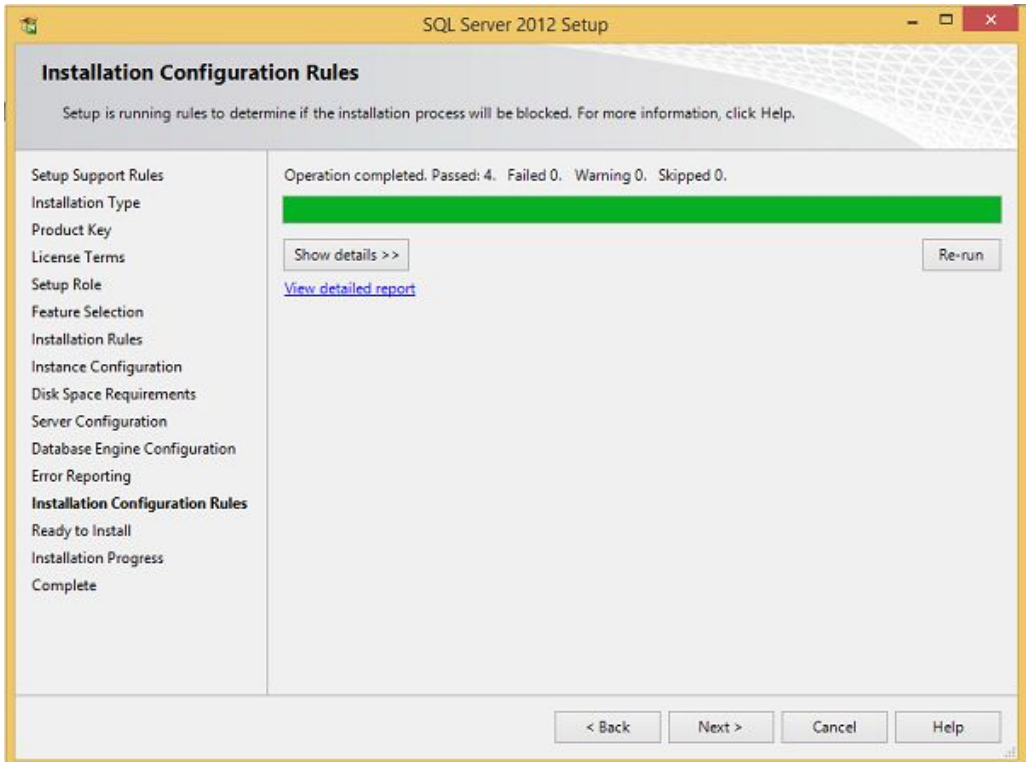
Step 17 − Make sure authentication mode selection and administrators are checked and click Data Directories.



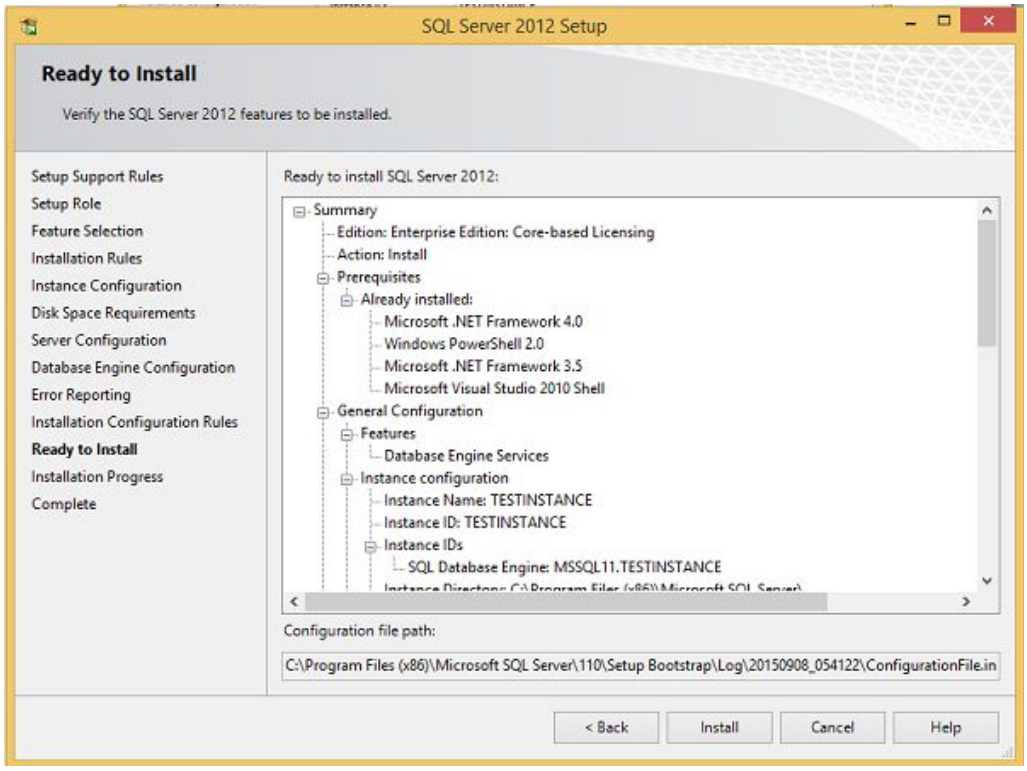
Step 18 − Make sure to select the above directory locations and click Next. The following screen appears.



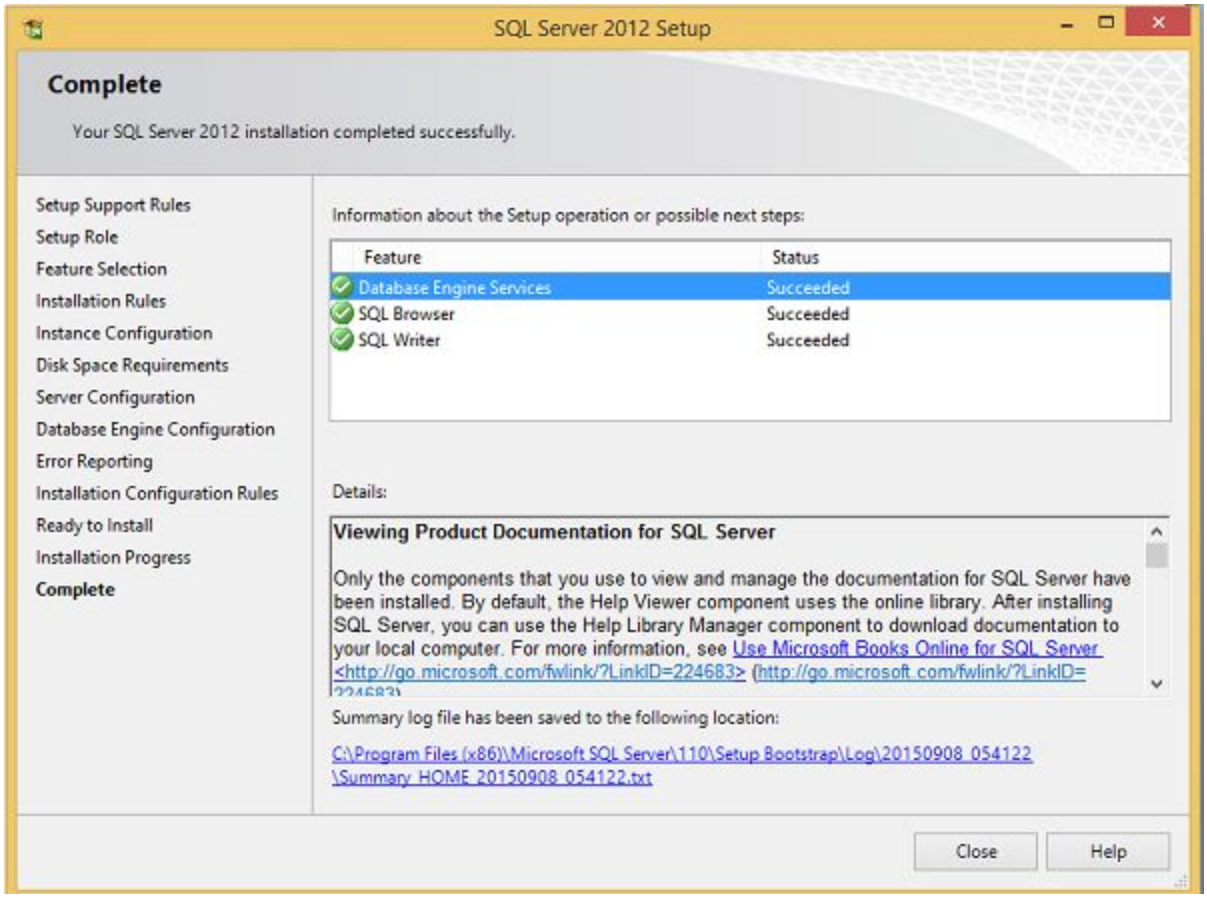
Step 19 − Click Next on the above screen.



Step 20 − Click Next on the above screen to the get the following screen.



Step 21 − Make sure to check the above selection correctly and click Install.



Installation is successful as shown in the above screen. Click Close to finish.

After installation.

Assigning a TCP/IP Port Number to the SQL Server Database Engine First, you may want to change the SQL Servers default TCP port from 1433 to a different port.

This exercise describes how.

1. Open SQL Server Configuration Manager by clicking Start | All Programs | Microsoft SQL Server 2012 | Configuration Tools | SQL Server Configuration Manager.

2. In the left-hand navigation pane, expand SQL Server Network Configuration and click Protocols for MSSQLSERVER. If you are changing the port for a nondefault instance, then you will click Protocols for .

3. Right-click TCP/IP in the left section.

4. You can configure each specific IP address, or you can configure the port for all IP addresses. To do so, click the IP Addresses tab, scroll to the bottom to locate IPALL, and change the port number to your desired port. Don't change the port, as this will require you to include the port number when connecting to this server.
5. Restart the instance of SQL Server that has been changed. Click SQL Server Service in the left navigation pane.
6. Select SQL Server (MSSQLSERVER), right-click, and select Restart.

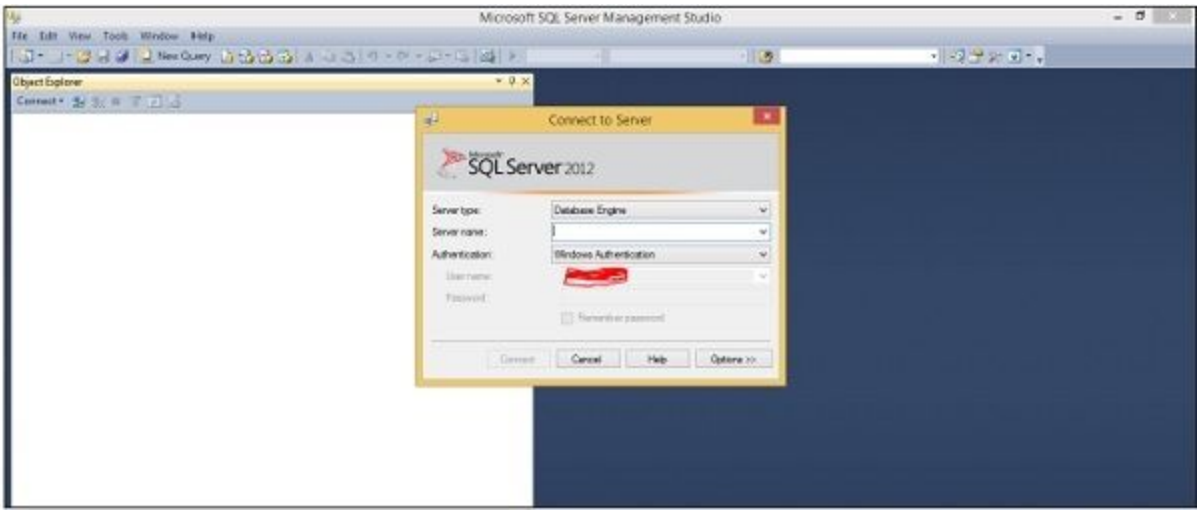
Once this change is made, you are required to specify this port number when you connect to the SQL Server instance.

II. SQL Server Management Studio

SQL Server Management Studio is a workstation component\client tool that will be installed if we select workstation component in installation steps. This allows you to connect to and manage your SQL Server from a graphical interface instead of having to use the command line. In order to connect to a remote instance of an SQL Server, you will need this or similar software. It is used by Administrators, Developers, Testers, etc. The following methods are used to open SQL Server Management Studio.

First Method
Start -> All Programs -> MS SQL Server 2012 -> SQL Server Management Studio

Second Method
Go to Run and type SQLWB (For 2005 Version) SSMS (For 2008 and Later Versions). Then click Enter. SQL Server Management Studio will be open up as shown in the following snapshot in either of the above method.



While out of the box SSMS is configured to provide a full set of functionality to administrators and developers, it also provides you with the ability to make it your own. If you don't like Object Explorer on the left, you can move it, or if you don't like the font of the query editor, you can change it to one of your choice. You have several options available for configuration.

Personalize SQL Server Management Studio

1. Open SSMS if it is not already open.
2. Select Tools | Options.
3. In the Options dialog box, select Fonts and Colors.
4. Select Courier New from the Font drop-down list.
5. Select 16 from the Size drop-down list.
6. Click OK.
7. Open a query window and type SELECT @@SEVERNAME. Click the red exclamation point icon in the menu bar to execute the query.
8. Open Object Explorer if it is not already open by selecting Object Explorer from the View menu or press F8
9. Click the drop-down arrow located to the right of the words Object Explorer. Select Float from the menu.
10. Click and drag Object Explorer onto the left docking option that appears. This docks Object Explorer back in its original position. Explore a little and move it to other docking locations. Find the one that best fits your preference

III. Login database

A login is a simple credential for accessing SQL Server. For example, you provide your username and password when logging on to Windows or even your e-mail account. This username and password builds up the credentials. Therefore, credentials are simply a username and a password.

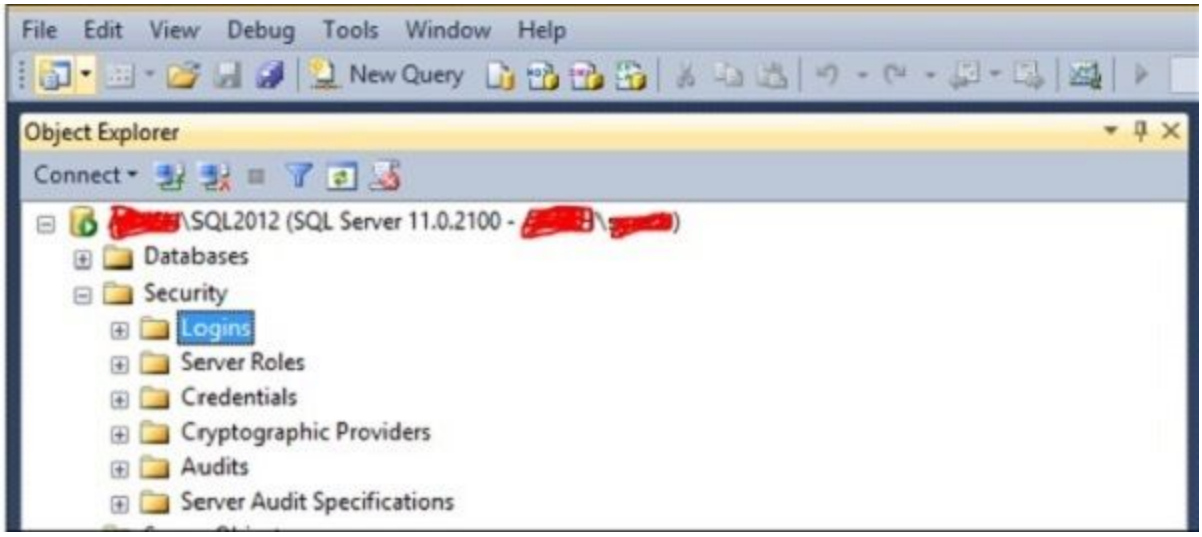
SQL Server allows four types of logins:

- A login based on Windows credentials.
- A login specific to SQL Server.
- A login mapped to a certificate.
- A login mapped to asymmetric key.

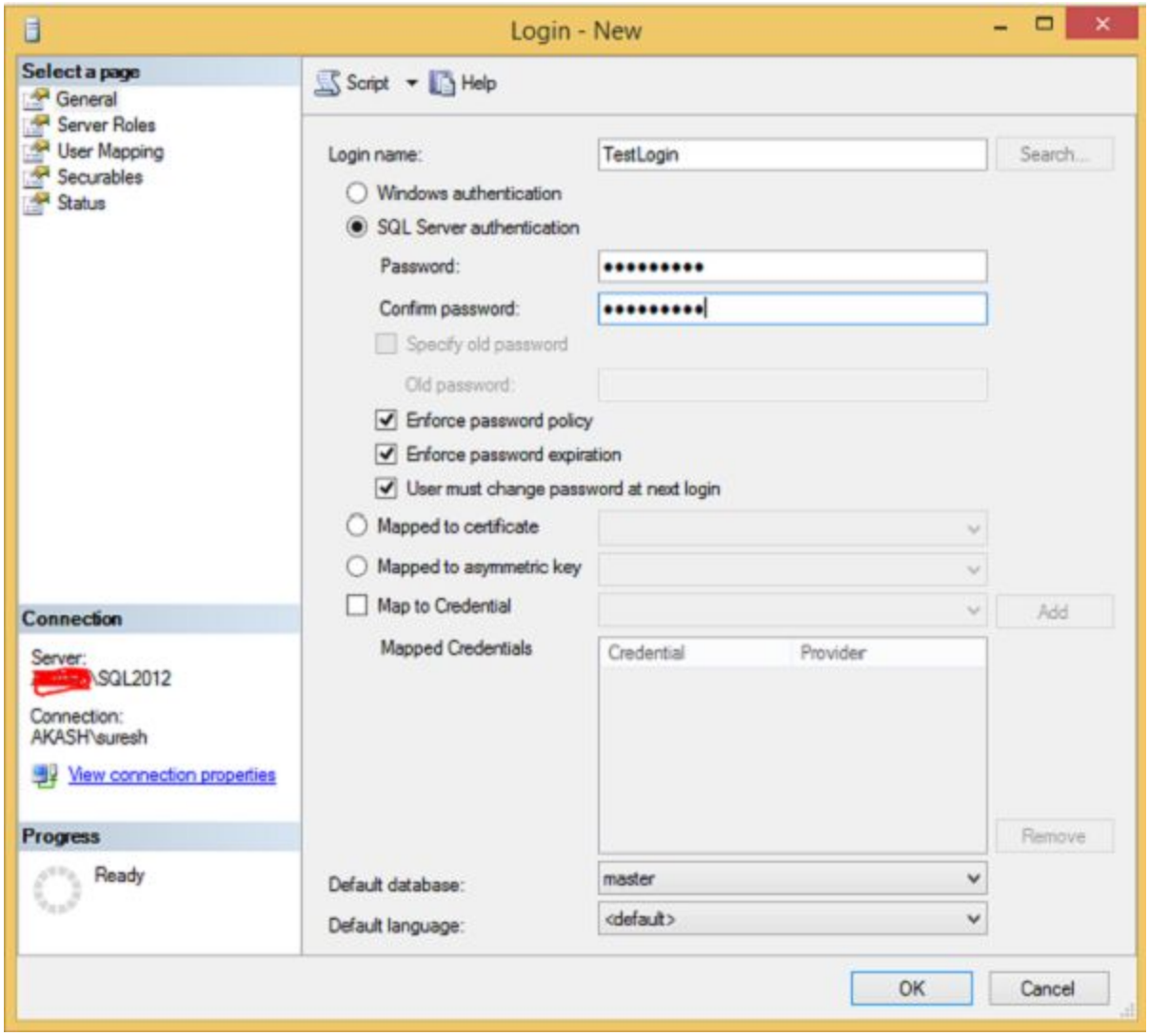
In this tutorial, we are interested in logins based on Windows Credentials and logins specific to SQL Server. Logins based on Windows credentials allow you to log in to SQL Server using a Windows username and password. If you need to create your own credentials (username and password,) you can create a login specific to SQL Server. To create, alter, or remove a SQL Server login, you can take one of two approaches: Using SQL Server Management Studio. Using T-SQL statements. Following methods are used to create Login:

First Method – Using SQL Server Management Studio

Step 1 − After connecting to SQL Server Instance, expand logins folder as shown in the following snapshot.



Step 2 − Right-click on Logins, then click Newlogin and the following screen will open.



Step 3 − Fill the Login name, Password and Confirm password columns as shown in the above screen and then click OK.

Login will be created as shown in the following image.

Second Method – Using T-SQL Script

```
Create login yourloginname with password='yourpassword'
```

To create login name with TestLogin and password ‘P@ssword’ run below the following query.

```
Create login TestLogin with password='P@ssword'
```

IV. Understanding SQL Server Databases

Database is a collection of objects such as table, view, stored procedure, function, trigger, etc.
In MS SQL Server, two types of databases are available.

- System databases
- User Databases

1. System Databases

System databases are created automatically when we install MS SQL Server. Following is a list of system databases:

Master database

The master database, as its name suggests, is the primary system database. Without it, SQL Server cannot start. The master database contains the most important information about objects within the SQL Server instance, such as the following:

■ Databases
■ AlwaysON
■ Database mirroring
■ Configurations
■ Logins
■ Resource Governor
■ Endpoints

For example, if you want to quickly obtain a list of all the databases on an instance of SQL Server, you can execute the following query: //The following code returns a list of all databases on an instance of SQL Server

Select * from sys.master_files

This query returns a list of databases and also additional configuration options that have been specified for each database. This approach is faster than using Microsoft SQL Server Management Studio (SSMS), where you view this information one database at a time.

Tempdb database

The tempdb database is a global playground for temporary objects created by the internal processes that run SQL Server and temporary objects that are created by users or applications. These temporary objects included temporary tables and stored procedures, table variables, global temporary tables, and cursors. In addition to temporary objects, tempdb stores row versions for read-committed or snapshot isolation transactions, online index operations, and AFTER triggers. One important thing to note about tempdb is that it is re-created every time SQL Server is restarted. Although you can create objects in tempdb, you should never use it as a database where persisted information is stored.

Model database

The model database is exactly what its name implies: a model for all databases that are created on an instance of SQL Server. In other words, it's used as a template each time you create a database. For example, if you want a particular table to exist in every database created on an instance of SQL Server, you will create that table in the model database. As a result, each time a database is created, it will include that table.

Msdb database

The server serves primarily as the back-end database for Microsoft SQL Server Agent. Whenever you create and/or schedule a SQL Server Agent job, the metadata for that job is stored in this database. In addition to SQL Server Agent data, msdb stores information for the following components: ■ Service brokers ■ Alerts ■ Log shipping ■ SSIS packages ■ Utility control point (UCP) ■ Database mail ■ Maintenance plans resource database The resource database is a hidden, read-only database that is usually not discussed very often. The resource database's primary purpose is to improve the upgrade process from one version of SQL Server to the next. All system objects for an instance of SQL Server are stored within the resource database. This database cannot be backed up or restored. You should not attempt to change or move this database unless Microsoft Customer Support directs you to do so. distribution database

Resource database

The resource database is a hidden, read-only database that is usually not discussed very often. The resource database's primary purpose is to improve the upgrade process from one version of SQL Server to the next. All system objects for an instance of SQL Server are stored within the resource database. This database cannot be backed up or restored. You should not attempt to change or move this database unless Microsoft Customer Support directs you to do so.

Distribution database

The final system database is the distribution database. This database exists only when you have configured this instance as a distributor for replication. Prior to configuring replication, you must perform this configuration. All metadata and history for the various types of replication are stored within this database.

➔ **View system databases**

1. Open SQL Server Management Studio (SSMS) and connect to a server.
2. Object Explorer should be open. If it is not, press F8 to open it.
3. In Object Explorer, expand Databases.
4. You will see a folder labeled System Databases. Expand it.

2. User Databases:

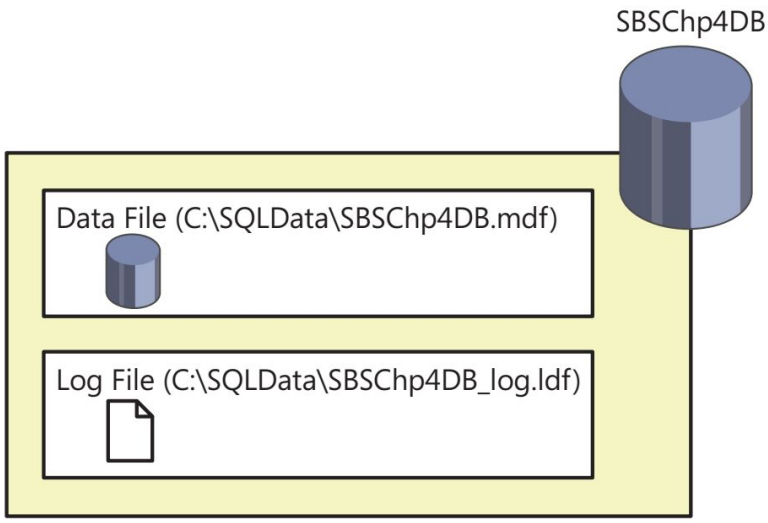
User databases are created by users (Administrators, developers, and testers who have access to create databases).

❖ Understanding the SQL Server database structure

As mentioned previously, databases are the primary data storage objects within SQL Server. The database creation process, while very simple, always requires careful thought relating to the structure. Databases can be created using many different technologies and techniques. In this chapter, you will focus on using T-SQL and SSMS. By default, every SQL Server database consists of two files:

■ The data file contains data and database objects such as tables, views, and stored procedures.

■ The log file contains information that assists in the recoverability of transactions in the database.



❖ Creating a database

There are two types of data files: primary and secondary.

When a database is initially created, the primary data file is created.

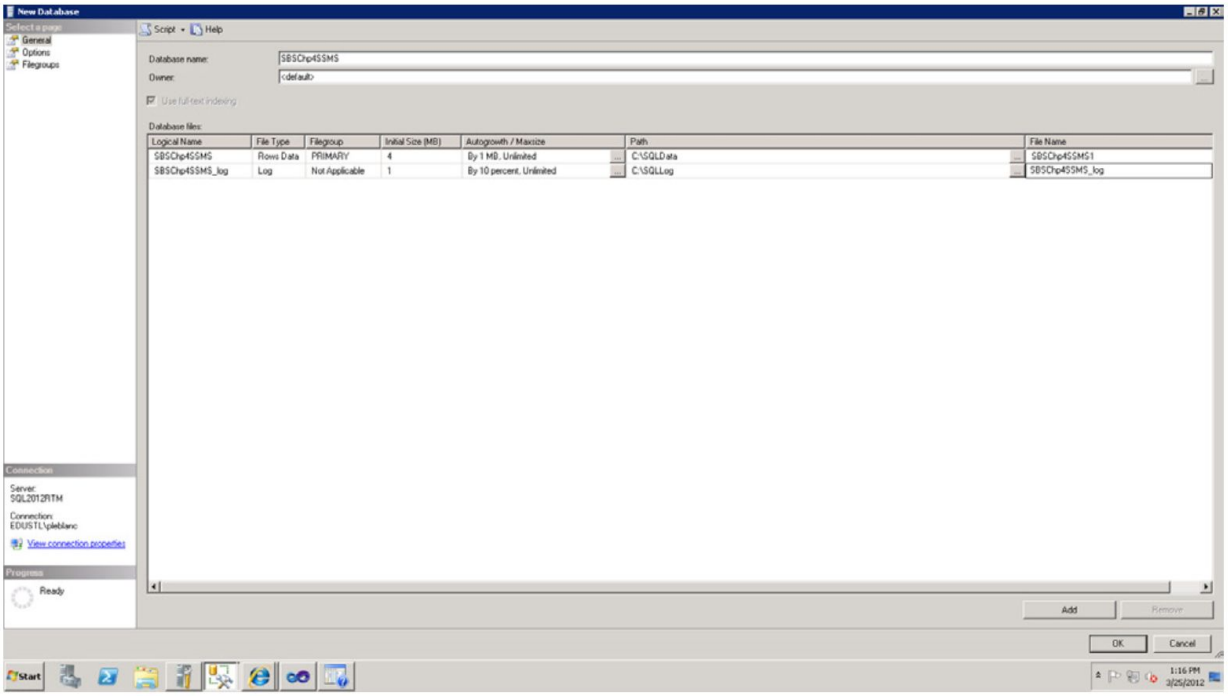
By default, it contains all the startup information for the database. As user-defined objects are created, they may also be stored in the primary data file. However, you may implement certain architectural strategies to improve the performance, scalability, and maintainability of your database.

Prior to running the script, create two folders on the root of your C drive: SQLData and SQLLog.

Create your first database with SSMS

1. Open SSMS.

2. Open Object Explorer, if it is not already opened.
3. Click the arrow next to your server.
4. Right-click the Databases folder.
5. In the context menu, select New Database.
6. The New Database dialog box opens. Ensure that General is selected in the Select a Page section on the left.
7. In the Database Name text box, type SBSChp4SSMS.
8. In the Database Files section, locate the Path column. On the first row under the Path column, click the ellipsis button. Browse to C:\SQLData.
9. On the same row, under the File Name column, type SBSChp4SSMS.
10. On the second row, under the Path column, click the ellipsis button. Browse to C:\SQLLog.
11. On the same row, under the File Name column, type SBSChp4SSMS_log.



12. Click OK
13. In the left section labeled Select a Page, select Filegroup

Create your frst database with T-SQL

1. Open the query editor in SSMS
2. In the query editor, enter the following T-SQL code:

```
--Use this script to create a database using T-SQL
USE master;
CREATE DATABASE SBSChp4TSQL
```

```
ON PRIMARY
(NAME='SBSChp4TSQL1', FILENAME = 'C:\SQLDATA\SBSTSQL1.mdf', SIZE=10MB,
MAXSIZE=20,
FILEGROWTH=10%)
LOG ON
(NAME='SBSChp4TSQL_log', FILENAME = 'C:\SQLLog\SBSTSQL_log.ldf',
SIZE=10MB, MAXSIZE=200, FILEGROWTH=20%);
```

❖ **Understanding arguments**

In the previous script, several arguments are used so that the database is placed in a specifc directory and it grows at a certain rate SQL Server provides a long list of arguments that can further extend how a database is created and where it resides The previous script uses the following commonly used arguments:

■ database_name is the name of the database, which must be unique to any of the databases that exist at the time of creation

■ ON specifies the filegroup and begins the section where the data file is defined.

■LOG ON begins the section where the log is defined.

■ Name is the logical file name used by SQL Server when referencing the fle. As with database_ name, it must be unique

■ FileName is the operating system path and file name, including the file extension.

■ Size specifies the initial size of the file in megabytes (MB) by default. Kilobytes (KB), gigabytes(GB), and terabytes (TB) can also be specified.

■Maxsize specifies the maximum size to which the fle can grow (shown in megabytes by default)

■ Filegrowth specifies the growth increment of the fle. It is also shown in megabytes by default, but it can be specified as a percentage

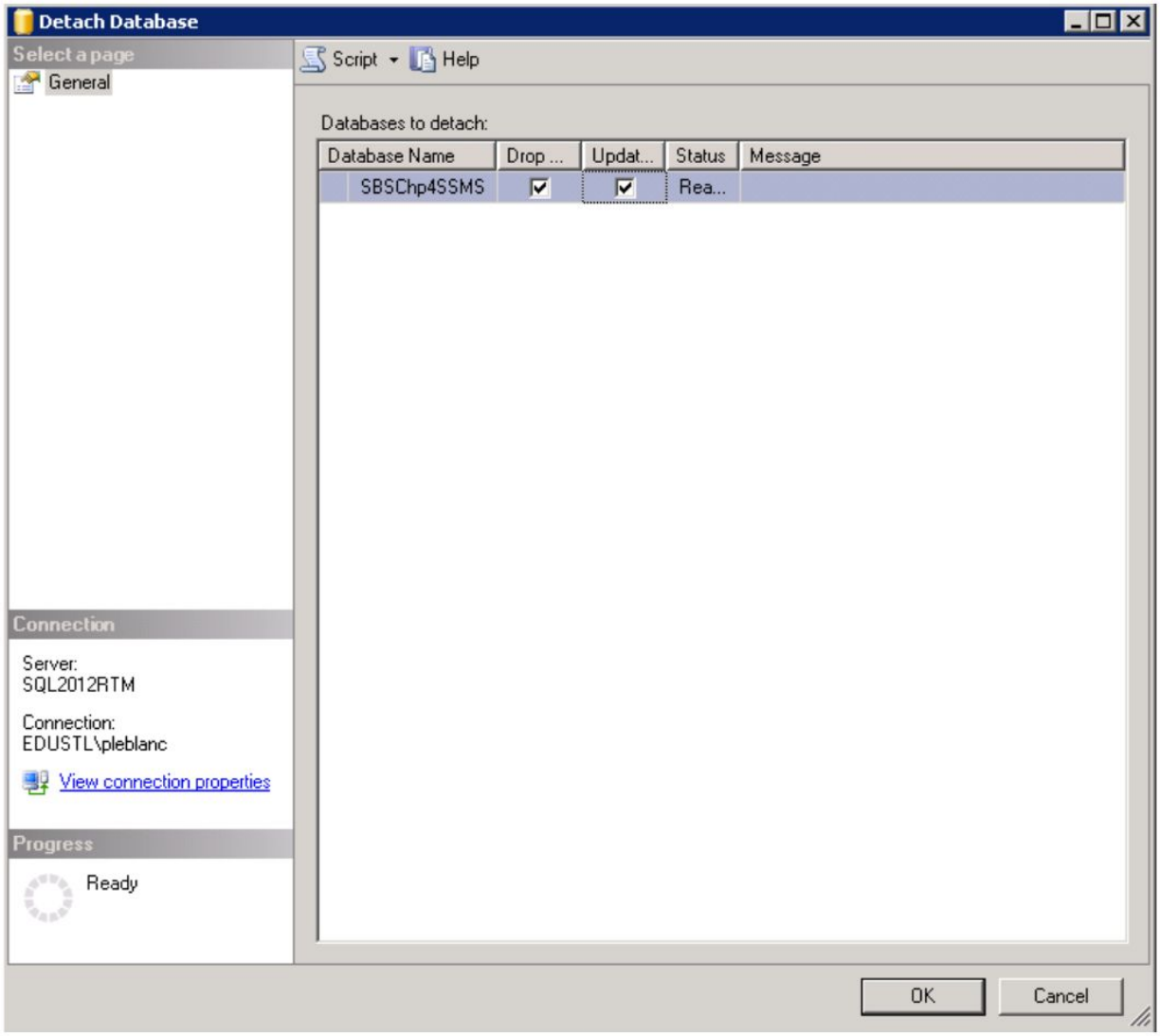
V. Detaching and attaching SQL Server databases

Now that you've created your database, what happens if you need to move it to another instance of SQL Server? For example, assume that you want to redistribute the free space on a server or decommission a server, which would require you to detach a database from one instance of SQL Server and then attach the database to a new instance of SQL Server.

To accomplish this, you can use either T-SQL or SSMS. There are currently two ways to attach a database to and one way to detach a database from an instance of SQL Server To attach a database, you use sp_attach or CREATE DATABASE specifying the FOR ATTACH argument. As a result, it is recommended that you use only the CREATE DATABASE option when attaching databases.

Detach a SQL Server database using SSMS

1. Open SSMS
2. Open Object Explorer, if it is not already open.
3. Expand the server node
4. Expand the Databases folder
5. Right-click the SBSChp4SSMS database.
6. Select Tasks | Detach.
7. In the Detach Database dialog box, check the boxes in the Drop Connections and Update Statistics columns



8. Click OK

Now that the database is detached, you can copy the files to the new storage location and attach the database to a new instance of SQL Server

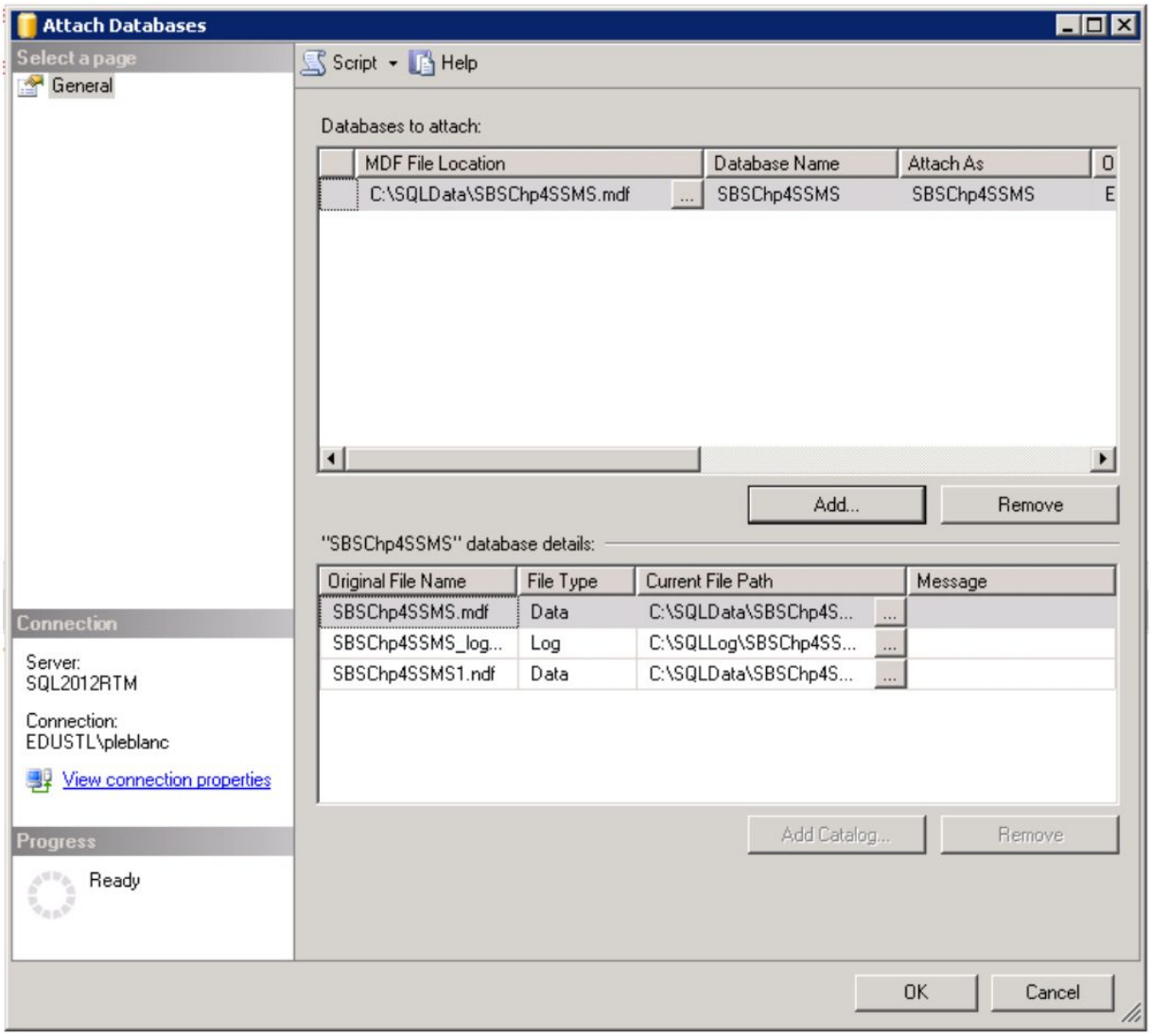
Detach a SQL Server database using T-SQL

1. Open SSMS, and then open a new query window
2. Enter and execute the following script:

```
USE Master;
EXEC sp_detach_db @dbname = 'SBSChp4TSQL';
```

Attach a SQL Server database using SSMS

1. Open SSMS
2. Open Object Explorer, if it is not already open.
3. Expand the server node
4. Right-click the Databases folder



5. Click Attach.

6. Click the Add button.
7. In the Locate Database Files dialog box, expand the folder labeled C.
8. Locate and expand the SQLData folder, and then select the SBSChp4SSMS.mdf file.
9. Click OK.

Attach a SQL Server database using T-SQL

1. Open SSMS, and then open a new query window
2. Enter and execute the following script:

```
USE master;
CREATE DATABASE SBSChp4TSQL ON
(FILENAME = 'C:\SQLData\SBSTSQL1.mdf'),
(FILENAME = 'C:\SQLData\SBSTSQL2.ndf'),
(FILENAME = 'C:\SQLLog\SBSTSQL_Log.ldf')
FOR ATTACH;
```

VI. DROP databases

To remove your database from MS SQL Server, use drop database command. Following two methods can be used for this purpose.

Method 1 – Using T-SQL Script

Following is the basic syntax for removing database from MS SQL Server.

```
Drop database <your database name>
```

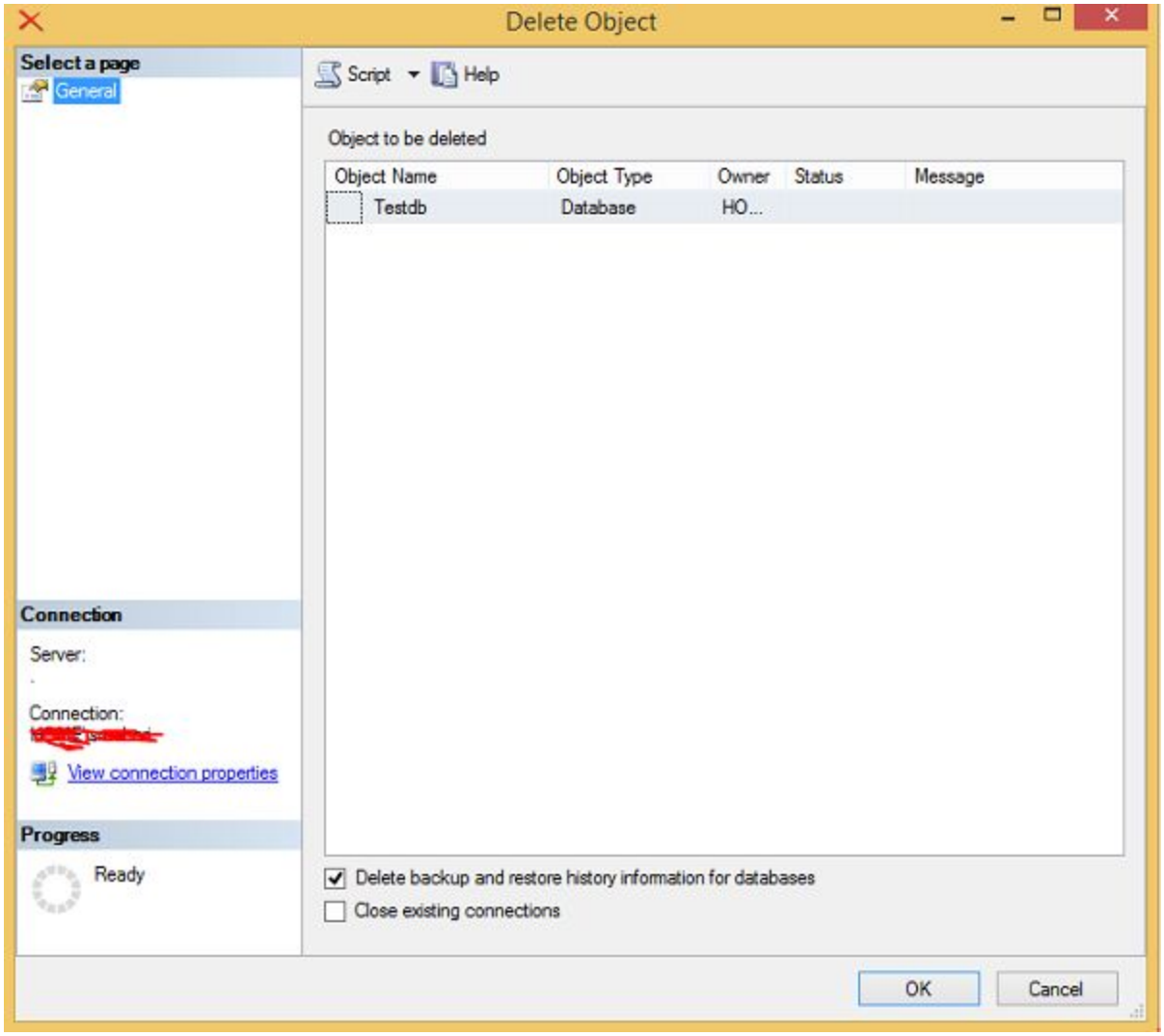
Example

To remove database name ‘Testdb’, run the following query.

```
Drop database Testdb
```

Method 2 – Using MS SQL Server Management Studio

Connect to SQL Server and right-click the database you want to remove. Click delete command and the following screen will appear.



Click OK to remove the database (in this example, the name is Testdb as shown in the above screen) from MS SQL Server.

VII. CREATE Backups

Backup is a copy of data/database, etc. Backing up MS SQL Server database is essential for protecting data. MS SQL Server backups are mainly three types − Full or Database, Differential or Incremental, and Transactional Log or Log.

Backup database can be done using either of the following two methods.

Method 1 – Using T-SQL

Full Type

```
Backup database <Your database name> to disk = '<Backup file location + file name>'
```

Differential Type

```
Backup database <Your database name> to
  disk = '<Backup file location + file name>' with differential
```

Log Type

```
Backup log <Your database name> to disk = '<Backup file location + file name>'
```

Example

The following command is used for full backup database called 'TestDB' to the location 'D:\' with backup file name 'TestDB_Full.bak'

```
Backup database TestDB to disk = 'D:\TestDB_Full.bak'
```

The following command is used for differential backup database called 'TestDB' to the location 'D:\' with backup file name 'TestDB_diff.bak'

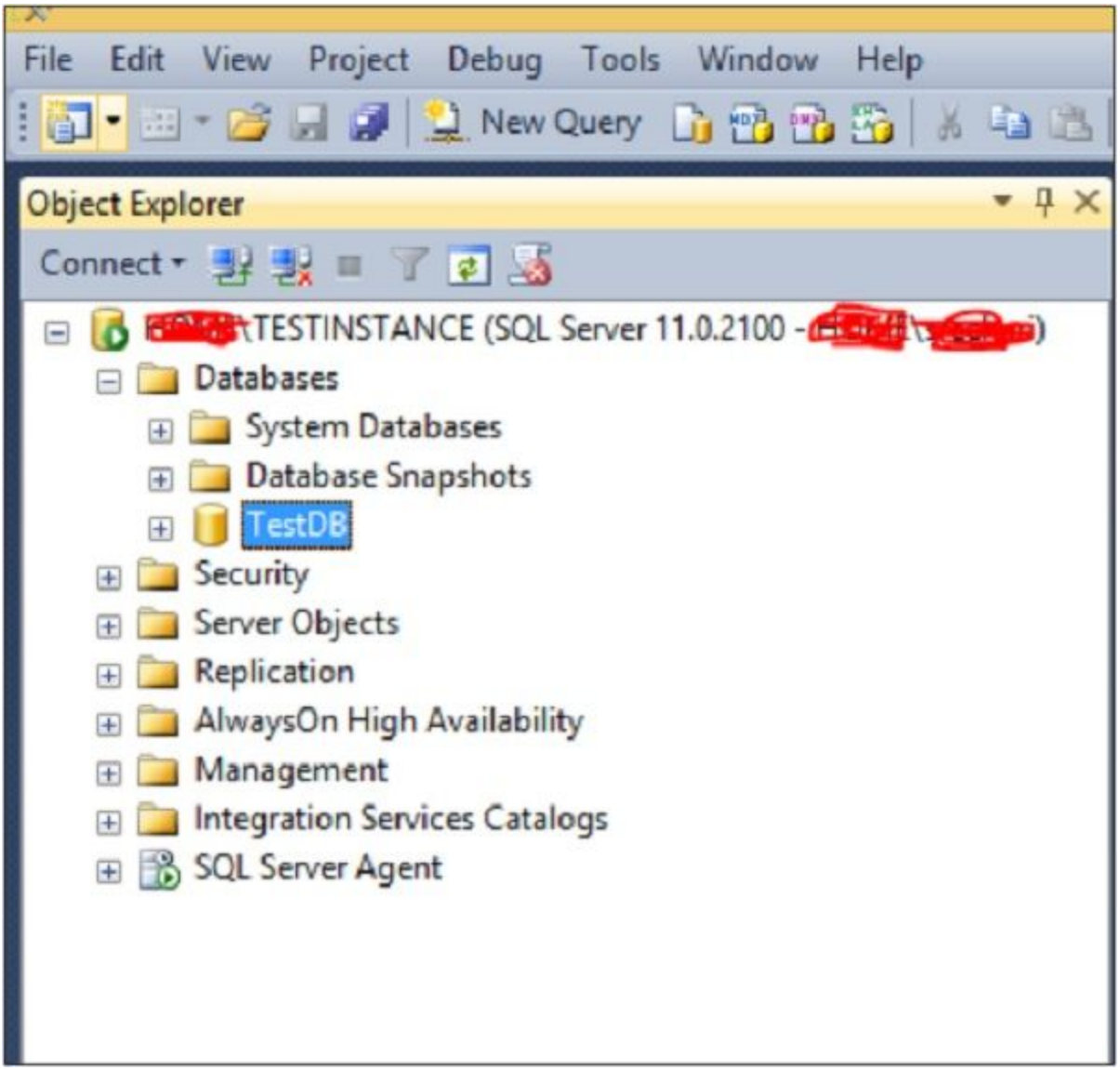
```
Backup database TestDB to disk = 'D:\TestDB_diff.bak' with differential
```

The following command is used for Log backup database called 'TestDB' to the location 'D:\' with backup file name 'TestDB_log.trn'

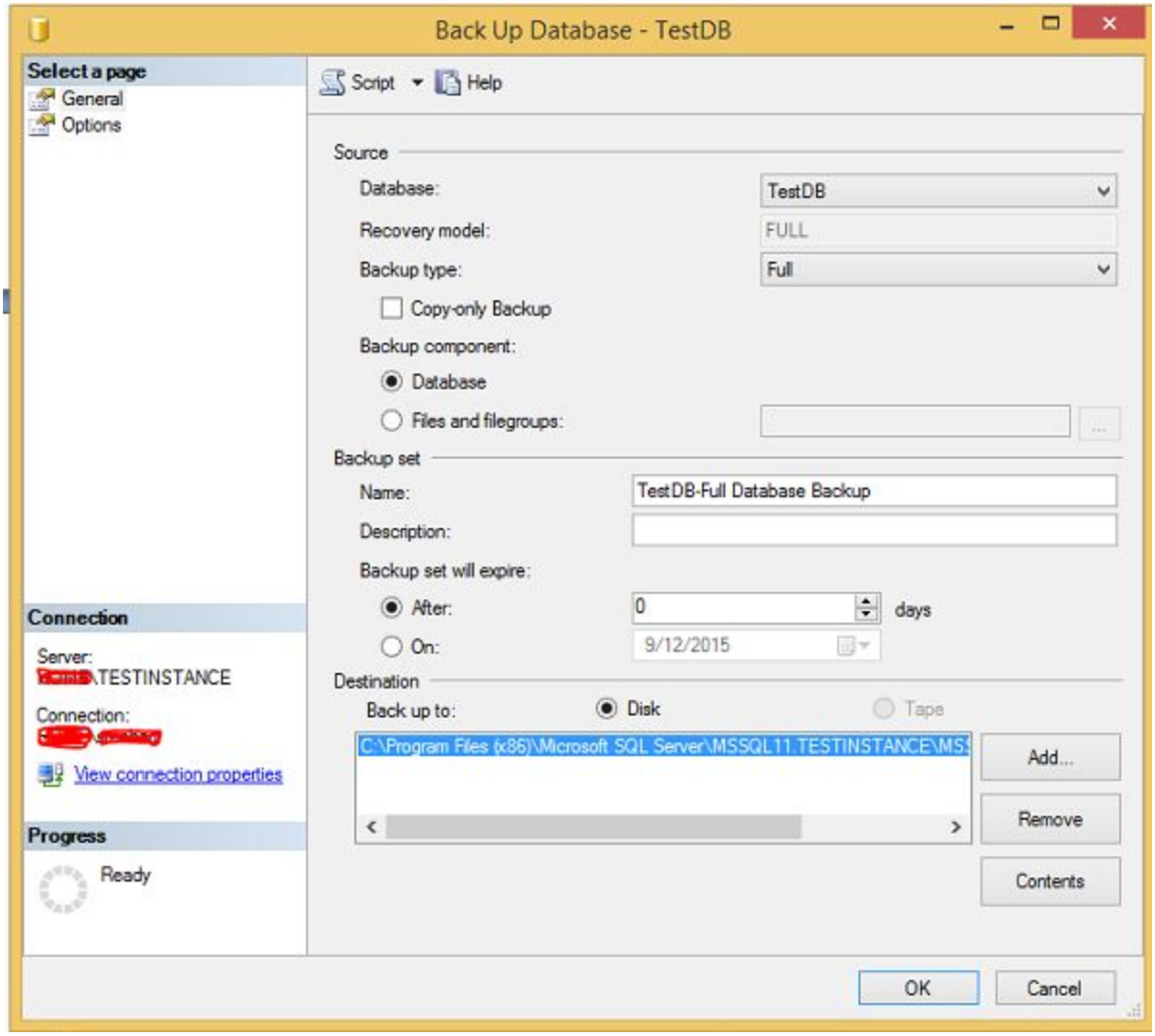
```
Backup log TestDB to disk = 'D:\TestDB_log.trn'
```

Method 2 – Using SSMS (SQL SERVER Management Studio)

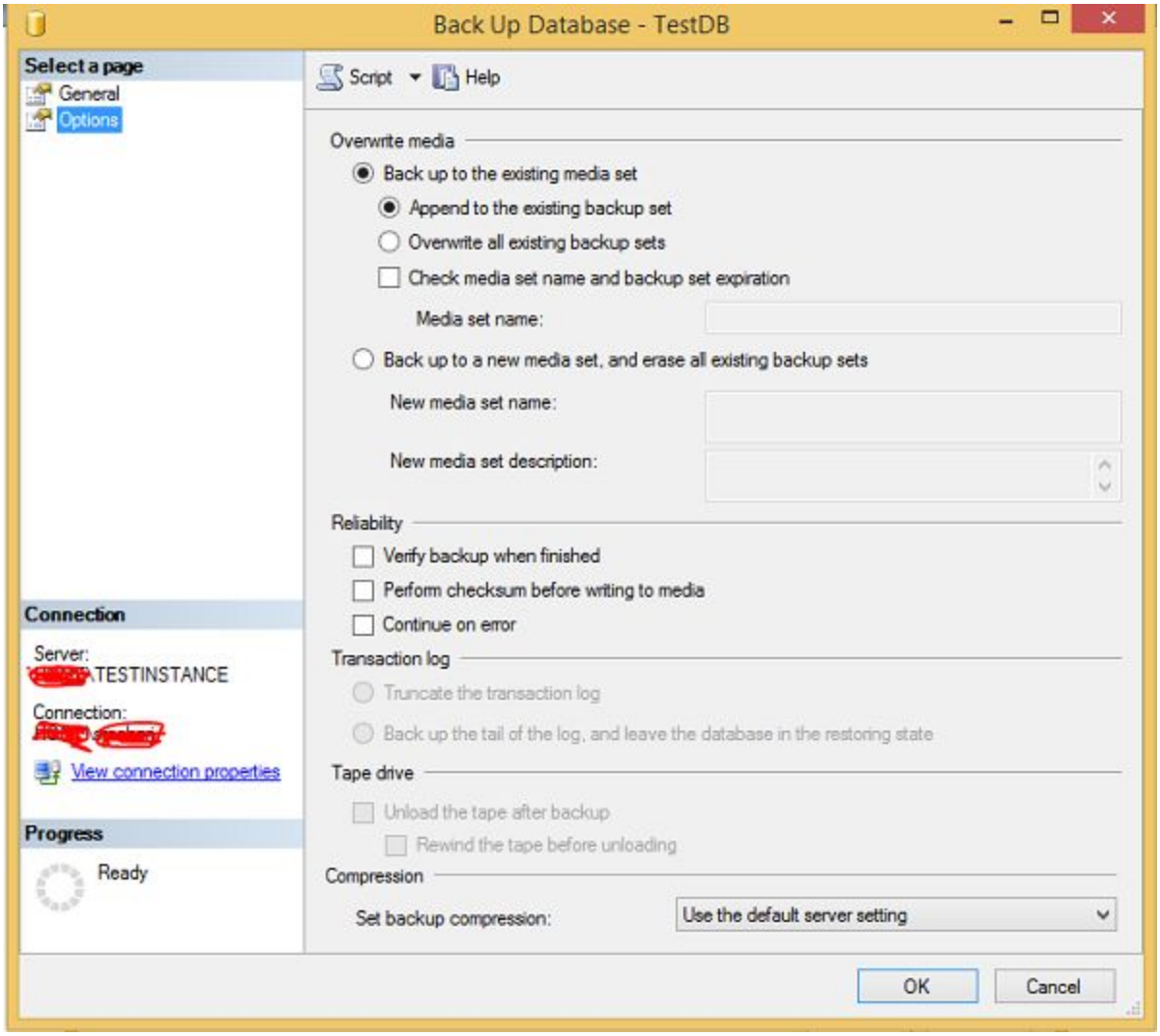
Step 1 − Connect to database instance named 'TESTINSTANCE' and expand databases folder as shown in the following snapshot.



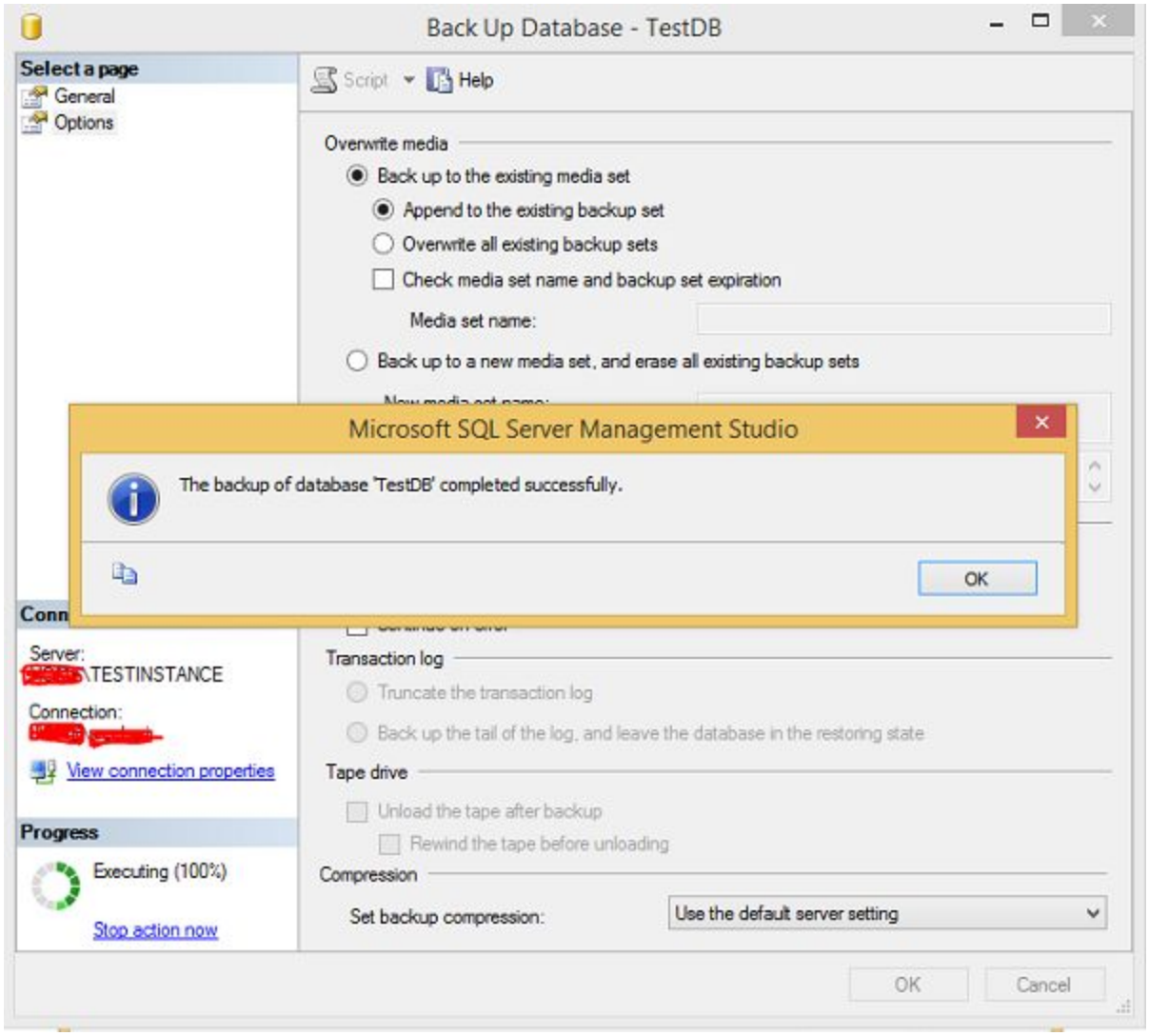
Step 2 − Right-click on 'TestDB' database and select tasks. Click Backup and the following screen will appear.

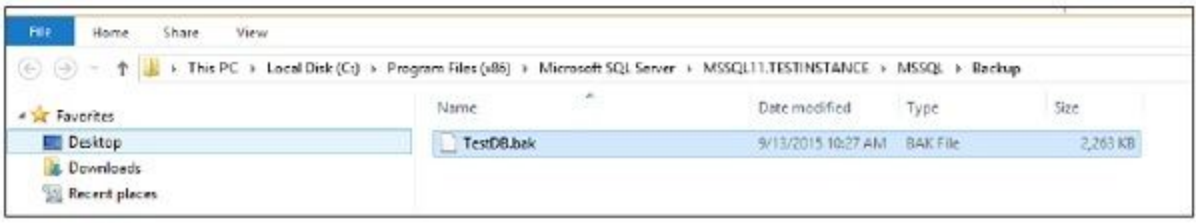


Step 3 − Select backup type (Full\diff\log) and make sure to check destination path which is where the backup file will be created. Select options at the top left corner to see the following screen.



Step 4 − Click OK to create 'TestDB' database full backup as shown in the following snapshot.





VIII. RESTORE databases

Restoring is the process of copying data from a backup and applying logged transactions to the data. Restore is what you do with backups. Take the backup file and turn it back into a database.

The Restore database option can be done using either of the following two methods.

Method 1 – T-SQL

Syntax

```
Restore database <Your database name> from disk = '<Backup file location + file name>'
```

Example

The following command is used to restore database called 'TestDB' with backup file name 'TestDB_Full.bak' which is available in 'D:\' location if you are overwriting the existed database.

```
Restore database TestDB from disk = ' D:\TestDB_Full.bak' with replace
```

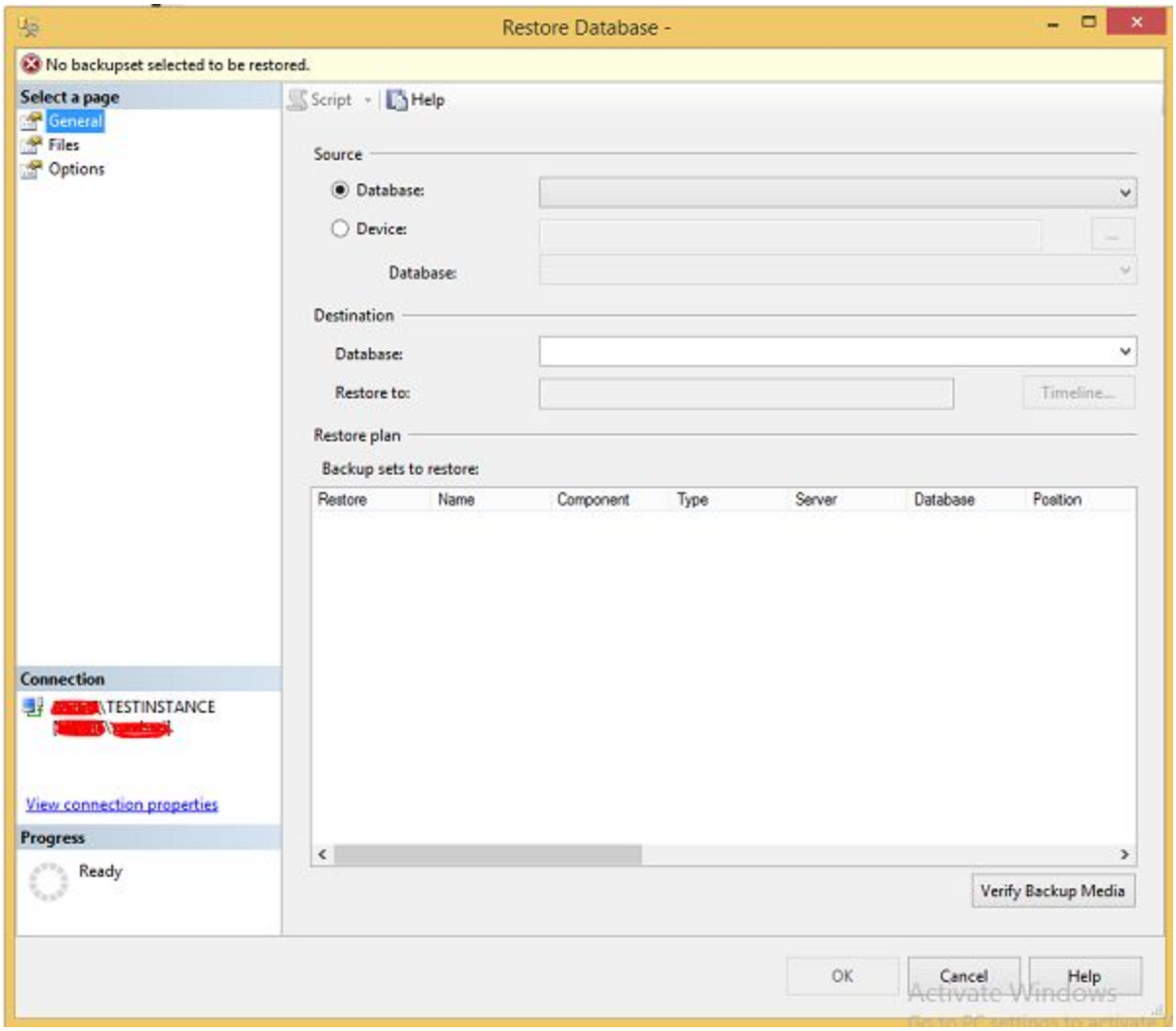
If you are creating a new database with this restore command and there is no similar path of data, log files in target server, then use move option like the following command.

Make sure the D:\Data path exists as used in the following command for data and log files.

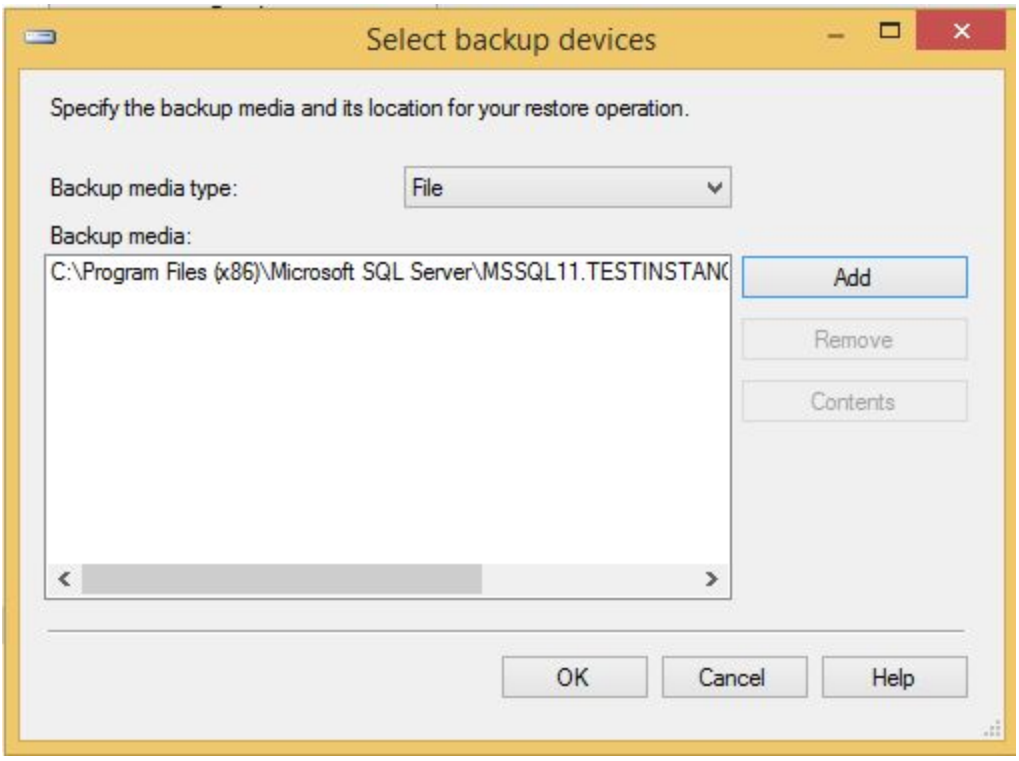
```
RESTORE DATABASE TestDB FROM DISK = 'D:\ TestDB_Full.bak' WITH MOVE 'TestDB'
TO
  'D:\Data\TestDB.mdf', MOVE 'TestDB_Log' TO 'D:\Data\TestDB_Log.ldf'
```

Method 2 – SSMS (SQL SERVER Management Studio)

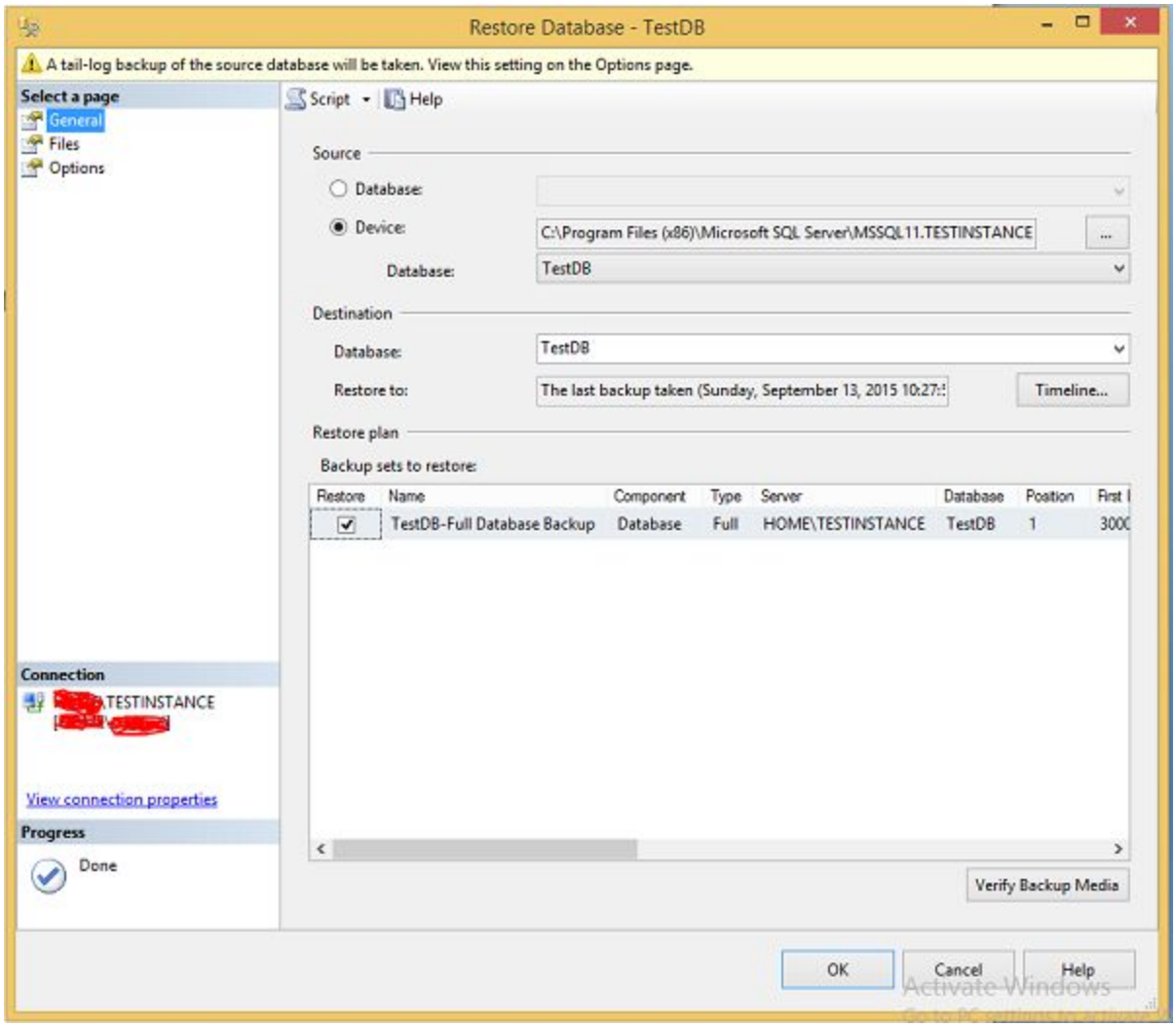
Step 1 − Connect to database instance named 'TESTINSTANCE' and right-click on databases folder. Click Restore database as shown in the following snapshot.



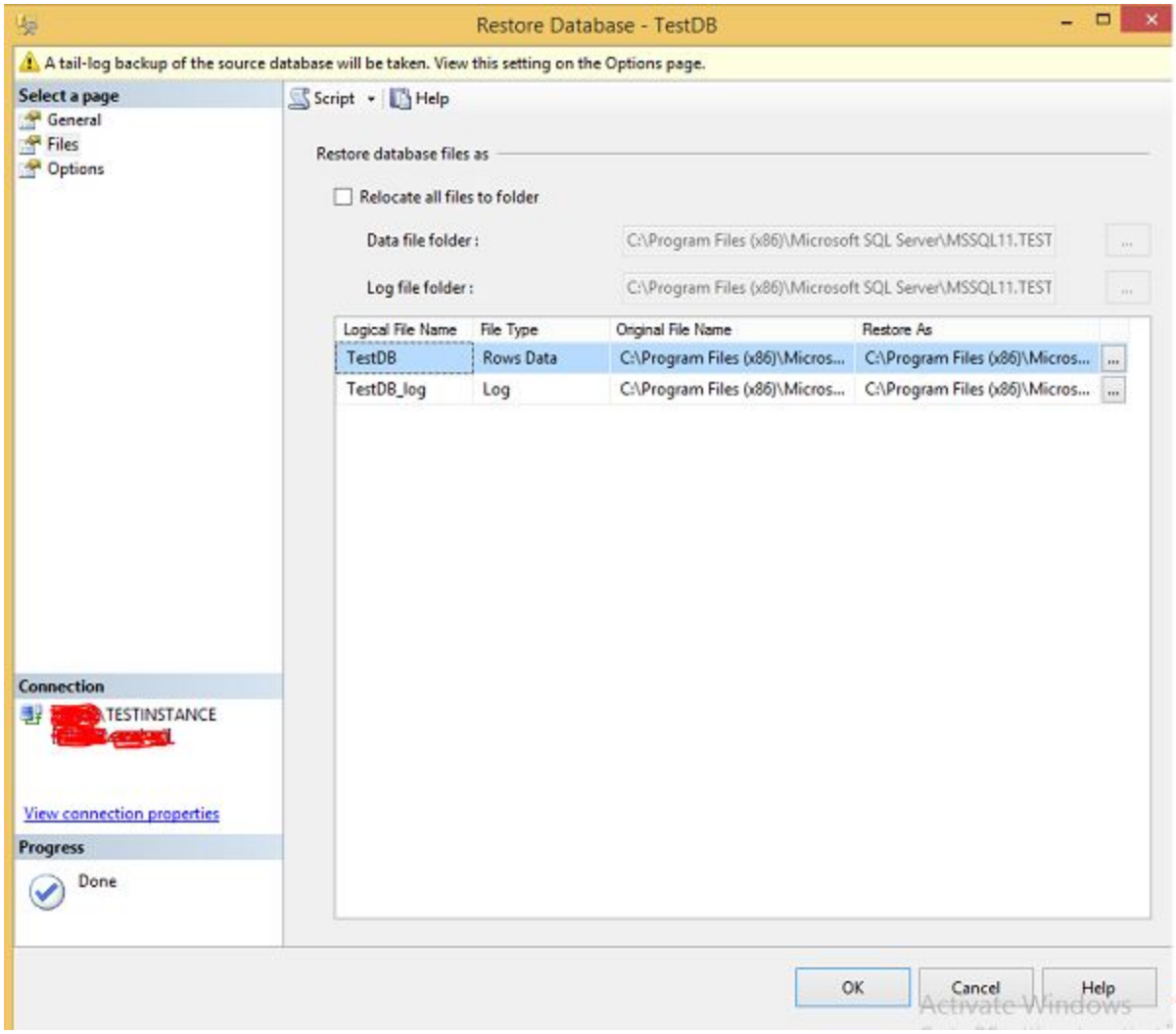
Step 2 − Select device radio button and click on ellipse to select the backup file as shown in the following snapshot.



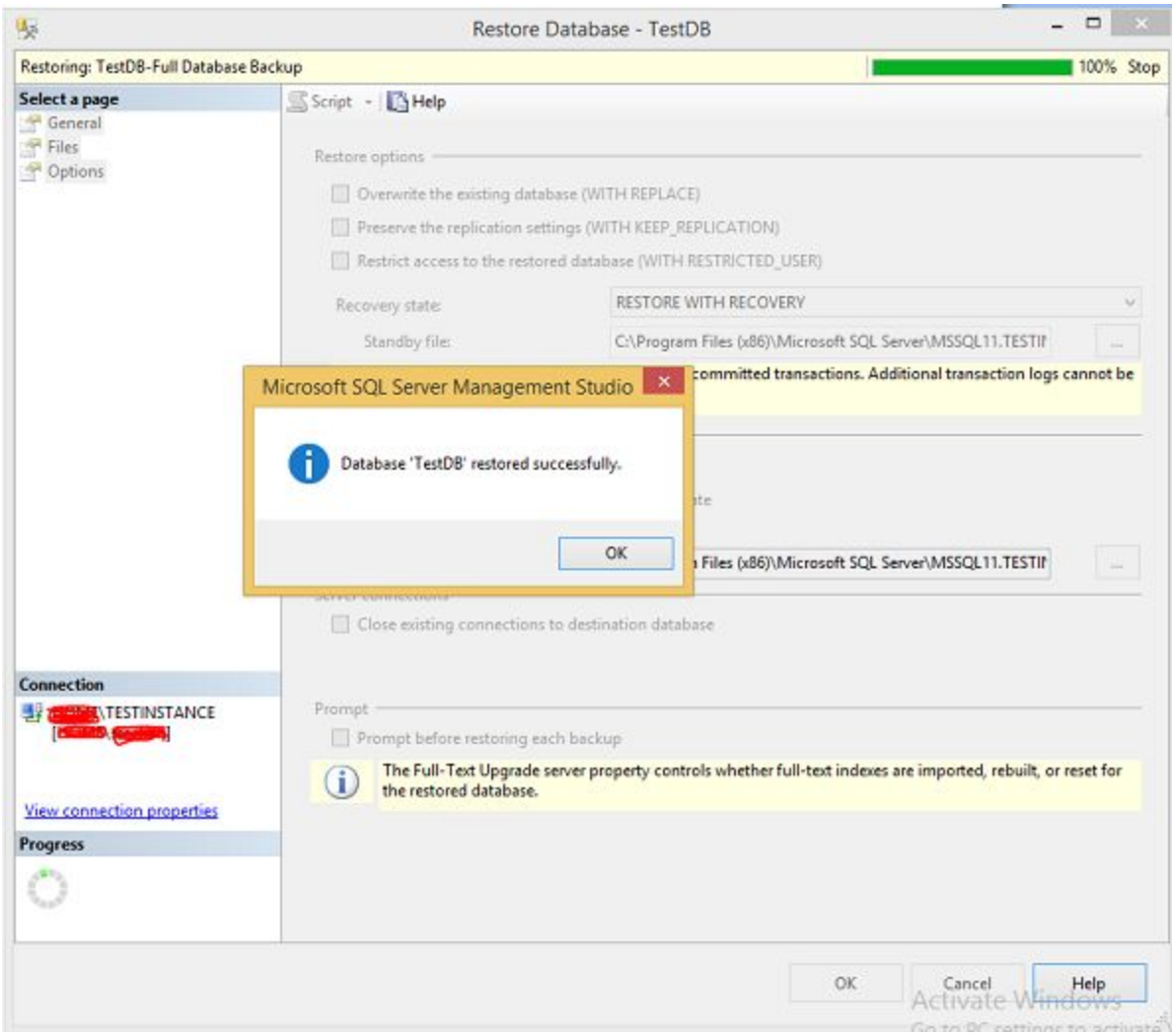
Step 3 − Click OK and the following screen pops up.



Step 4 − Select Files option which is on the top left corner as shown in the following snapshot.



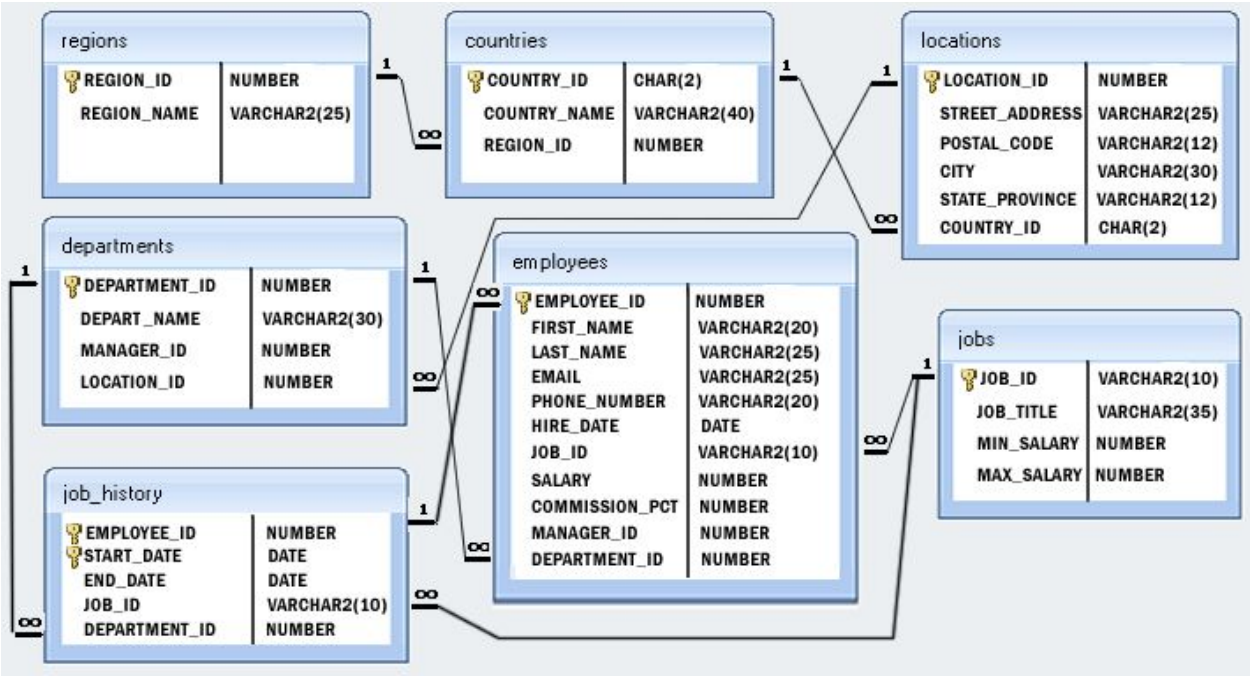
Step 5 − Select Options which is on the top left corner and click OK to restore 'TestDB' database as shown in the following snapshot.



==========

SECTION 02. CREATE YOUR DATABASE

1. Create your database with the name hr
2. Backup
3. Detach and store into USB


regions
REGION_ID NUMBER
REGION_NAME VARCHAR2(25)
countries
COUNTRY_ID CHAR(2)
COUNTRY_NAME VARCHAR2(40)
REGION_ID NUMBER
locations
LOCATION_ID NUMBER
STREET_ADDRESS VARCHAR2(25)
POSTAL_CODE VARCHAR2(12)
CITY VARCHAR2(30)
STATE_PROVINCE VARCHAR2(12)
COUNTRY_ID CHAR(2)
departments
DEPARTMENT_ID NUMBER
DEPART_NAME VARCHAR2(30)
MANAGER_ID NUMBER
LOCATION_ID NUMBER
employees
EMPLOYEE_ID NUMBER
FIRST_NAME VARCHAR2(20)
LAST_NAME VARCHAR2(25)
EMAIL VARCHAR2(25)
PHONE_NUMBER VARCHAR2(20)
HIRE_DATE DATE
JOB_ID VARCHAR2(10)
SALARY NUMBER
COMMISSION_PCT NUMBER
MANAGER_ID NUMBER
DEPARTMENT_ID NUMBER
jobs
JOB_ID VARCHAR2(10)
JOB_TITLE VARCHAR2(35)
MIN_SALARY NUMBER
MAX_SALARY NUMBER
job_history
EMPLOYEE_ID NUMBER
START_DATE DATE
END_DATE DATE
JOB_ID VARCHAR2(10)
DEPARTMENT_ID NUMBER