Most frequently used and real time problem solutions in SQL Server

SQL Server How to Tips & Tricks

Sheo Narayan

SN ITFunda Services LLP, India

V 1.0

Table of Contents

About Author

Mr. Sheo Narayan is a software professional since 2000 (15+ years) and working in .NET Technologies since its first release. He is also the founder of a popular .NET Community website http://www.DotNetFunda.com. He has been awarded many awards like Microsoft® Most Valuable Professional (MVP), Star Entrepreneur Award. He has served thousands of professionals worldwide in solving their technical issues and teaching them in their professional careers. His corporate training clients includes many MNCs.

Copyright notice

SN ITFunda Services LLP

- Website: http://sn.itfunda.com.com
- Email: support@itfunda.com
- Phone: +91-40-4222-2291, +91-768-088-9888

Warning and Disclaimer

Every effort has been taken to make this book and related materials as accurate as possible, but no warranty is implied. The information provided in this book and related materials are on “as is” basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damage arising from the information contained in this book or related items.

Sales information

This book and related items are available on sale on www.itfunda.com website.
For *bulk sale or corporate (commercial) license*, please contact at support@itfunda.com or on above mentioned phone numbers.

- Phone: +91-40-4222-2291

- Mobile: +91-768-088-9888
- Email: support@itfunda.com
- Website: http://www.itfunda.com

Acknowledgements

I would like to take this opportunity to thank my wife Dr. Sunita for all her support and encouragement and of course to my daughter Sindhuja and my son Shreeharsh for being so patient to get my time.

A lot of thanks to my website www.dotnetfunda.com that always inspires me to do more and more for the .NET Technology and its community.

I would also like to thank all my colleagues at DotNetFunda.Com. To name a few Sainath Sherigar, Shivprasad Koirala, Vakul Kumar More, Vuyiswa Maseko for their continuous support and contribution of countless number of hours in maintaining www.dotnetfunda.com.

Of course moderators like Radha Srikanth, are also doing pretty decent job and I would like to thank them as well for their support.

Last but not least, I would like to thank my staffs at SN ITFunda Services LLP for their continuous support in doing whatever I decide to do.

We want to hear from you

As always, we value any kind of feedback, comment on the book. Your feedback is highly appreciated and will help us to improve our ability to serve you. We believe a product never reaches its final stage as continuous improvements makes it better and more useful every day.

If you have any comments on this book or you want us to add any How to's related with .NET Technologies, please feel free to let us know and we shall try to add them in our future release.

Please contact us at below contact details

- For any feedback or support or to contact us, write to support@itfunda.com
  Phone us at +91-40-4222-2291, +91-768-088-9888
- We are also available at Instant Messenger - Skype: FundaSupport | Gmail: FundaHelpLine

References

Writing books without getting knowledge from different sources are impossible. Thanks to the authors of books, internet articles and other blogs for sharing their knowledge.

Introductions

1. How to install SQL Server database?

To install SQL Server 2014 express edition and Management studio, visit - http://www.microsoft.com/en-in/download/details.aspx?id=42299), the URL may change based on the version release of Microsoft SQL Server in future so please search in search engines like Google or Bing and find the correct url. The current home page of SQL Server at Microsoft is http://www.microsoft.com/en-us/server-cloud/products/sql-server/. The SQL Server 2014 database details and versions can be found at http://www.microsoft.com/en-us/server-cloud/products/sql-server/.

The Express edition of the SQL Server is completely free and most suitable for the developers.

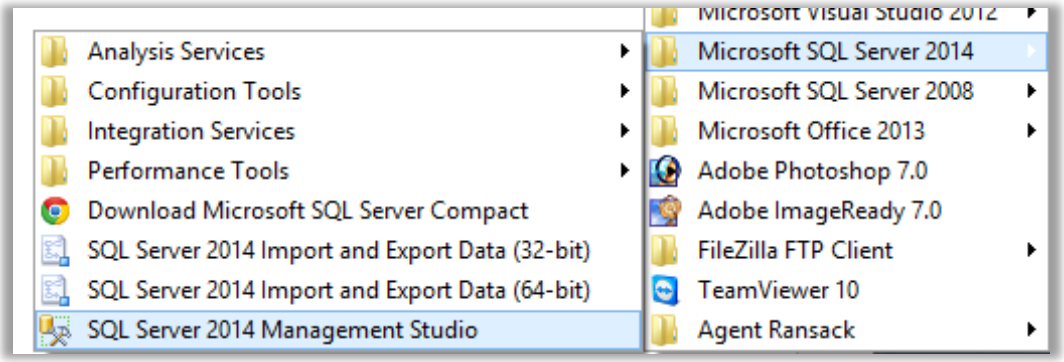
More details on different versions of SQL Server can be found at http://www.microsoft.com/en-us/server-cloud/products/sql-server/Purchasing.aspx

Please also note that by default SQL Server database doesn't have any user interface through which we can interact with the database so we use Microsoft SQL Server Management studio (Integrated Development Environment - IDE) to work with the SQL Server database.

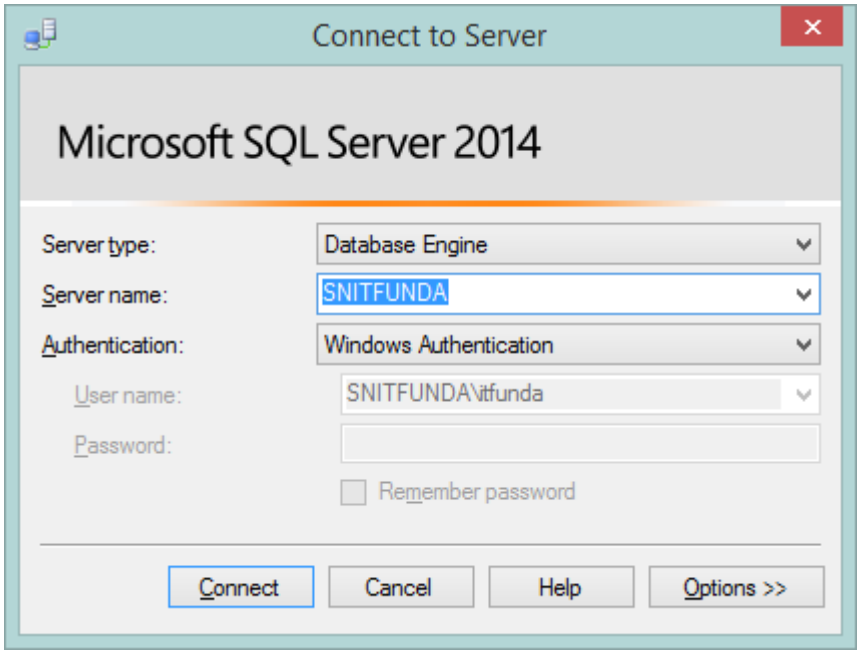
Please ensure that when you are installing the database, you also choose to install Microsoft Management Studio or SQL Server (version) Management studio.

2. How to connect database server in Microsoft Management Studio?

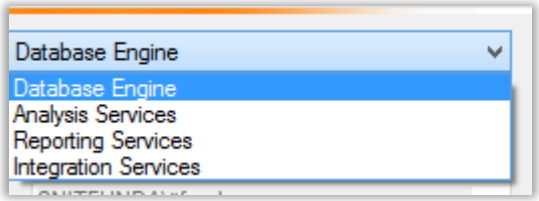
To connect to database using Microsoft Management Studio, first open it from the Programs or click the SQL Server xxxx Management Studio icon like below.



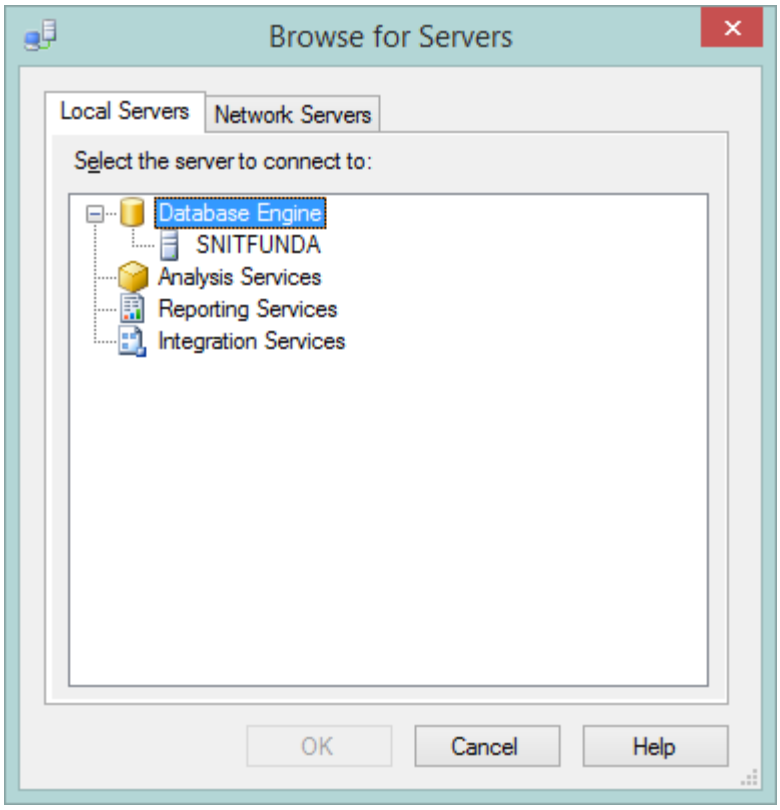
The first screen it gives us a dialog box named Connect to Server screen like below



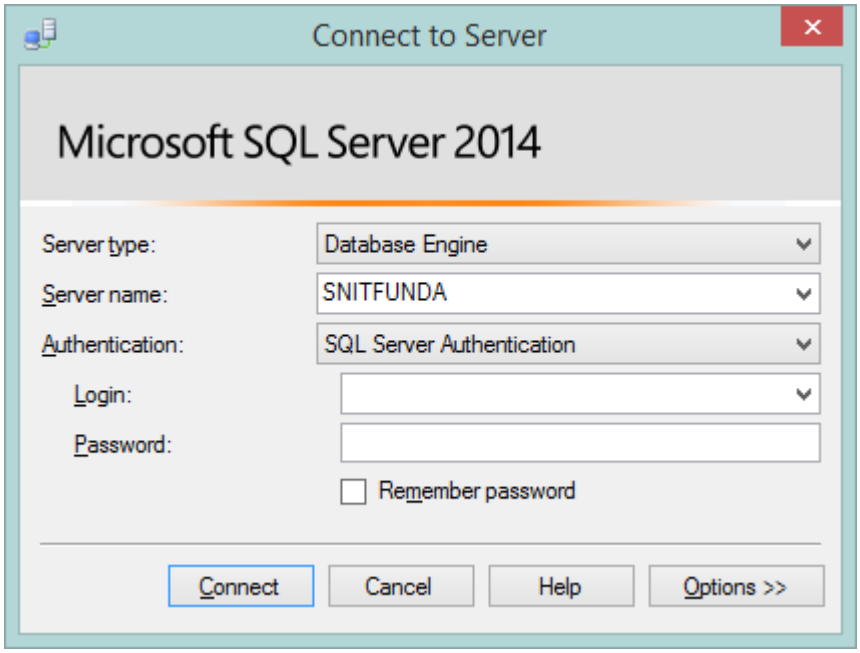
This allows us to choose the Server type, SQL Server Management Studio can connect any of four server type as shown below.



The next box allows us to choose the Server to connect to. In general, this is the name of computer on which the database is installed that is by default written in the TextBox, if not the dropdown icon can be clicked and <Browse for more …> option can be selected that gives option to select other server that is either of locally available or on the Network computers.

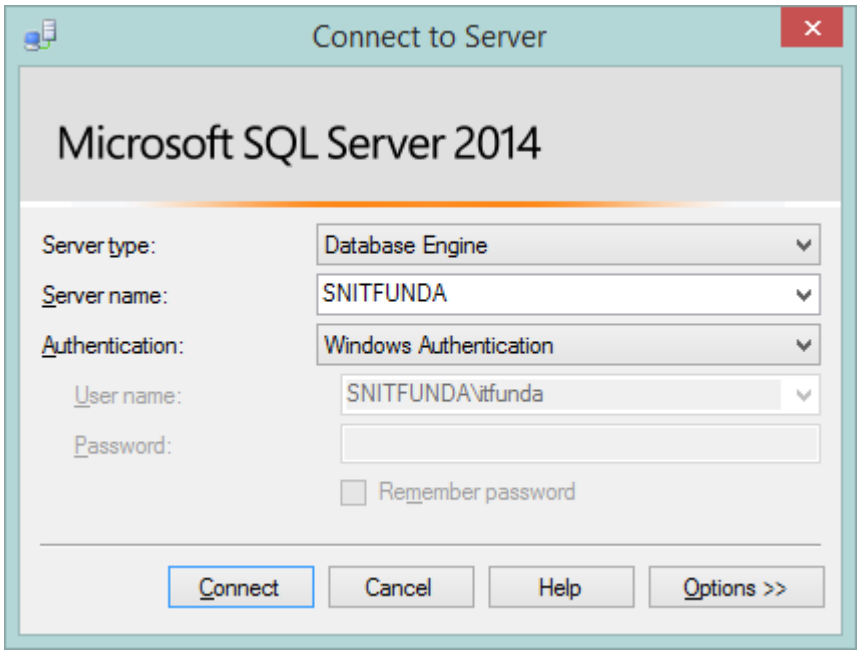


The third dropdown allows us to select Authentication type to use to connect to the server, it can be Windows Authentication (the system credentials are used to connect) or SQL Server Authentication like below

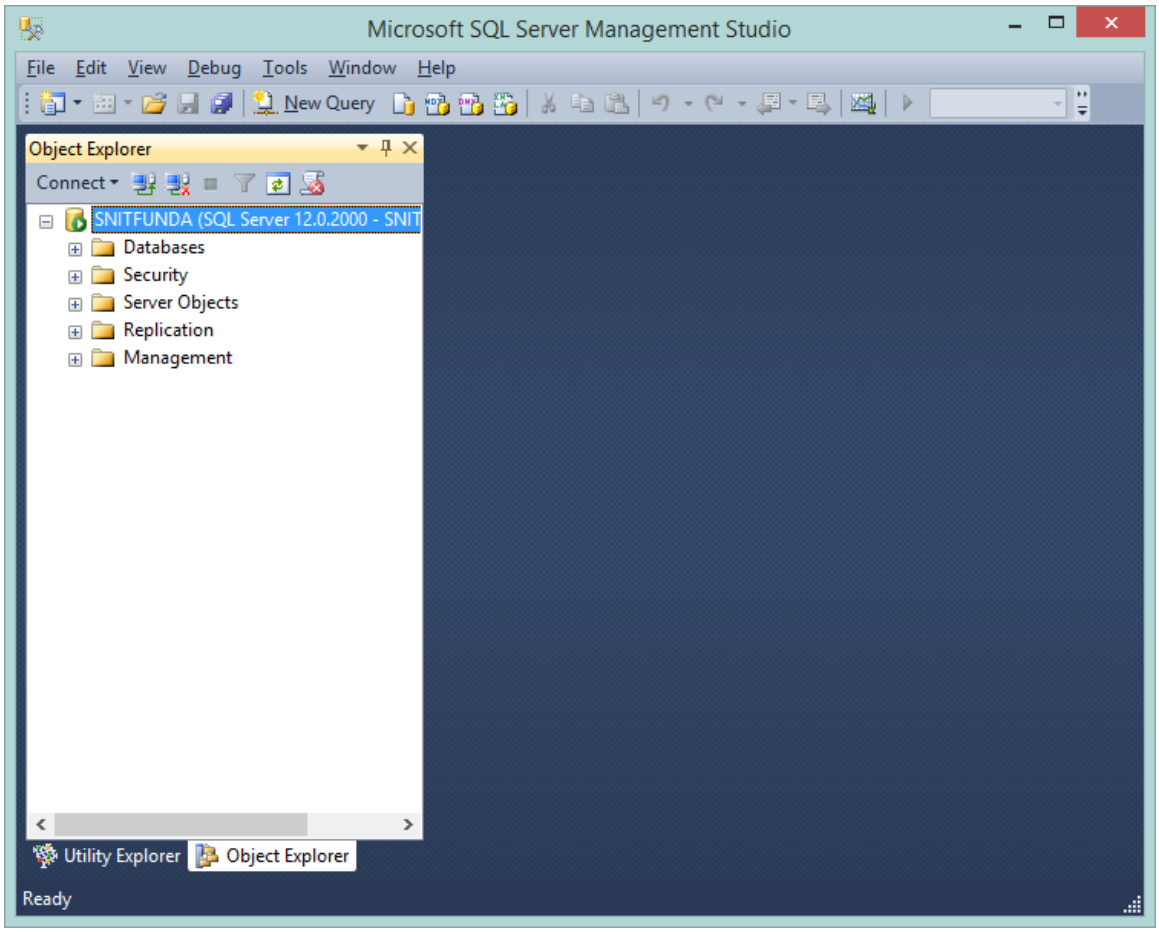


Choosing SQL Server Authentication enables Login and Password textbox to enter proper credentials to connect to the database.

In general, we use Windows Authentication to connect to the local server.



So click on the Connect buttons that gives us the complete screen of SQL Server Management Studio.



Notice the left side panel that has many folders where appropriate objects related with database are kept. The green Play icon at the top-left panel shows the server name that it is connected to.

Description about left side folder – Most of the developers works mainly on two folders of the left panel and they are

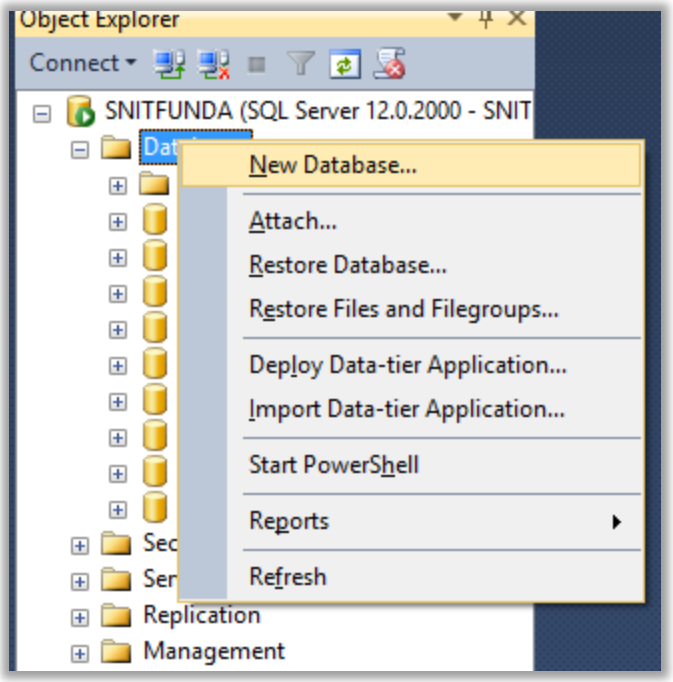
- Databases – this allows us to create new database, re-store backed up database and all database of this server is kept under this folder
- Security – allows us to create new Logins to the database and this server

Remaining folders are either for the SQL Server administrators or to perform some advance operations. Few of them we will learn as and when we will progress.

Database

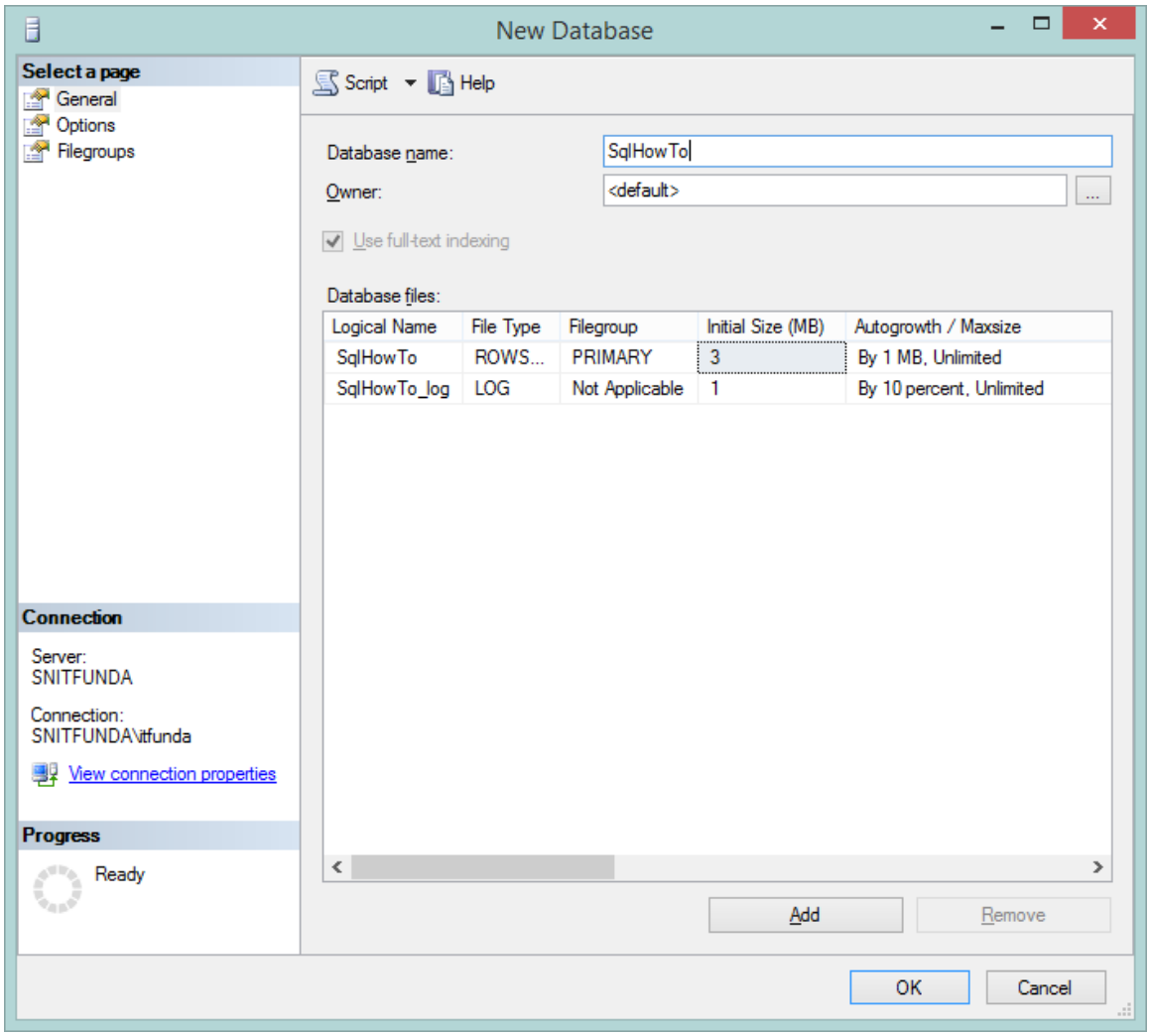
3. How to create a new database in SQL Server?

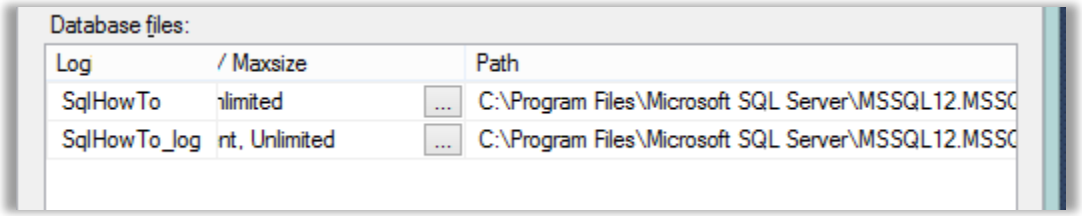
To create a new database in SQL Server, open the SQL Server Management Studio and right click the Databases folder.



Select New Database… that opens up New Database dialog box.

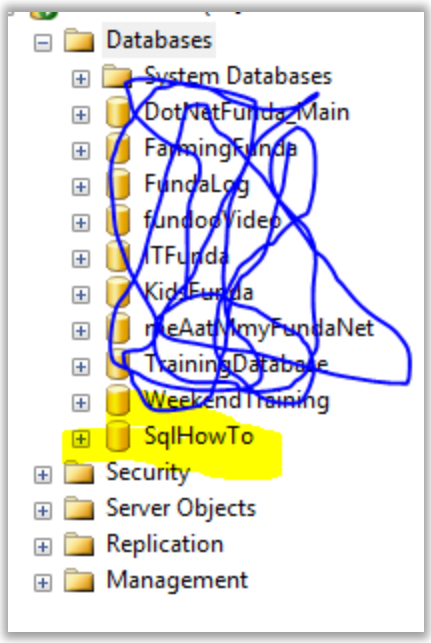
This dialog box allows us to write the name of the database, choose the owner (optional). Select the path where the database file will be stored (scroll to the right to see the Path column, click on the … button to select the folder where we want to save the database file.





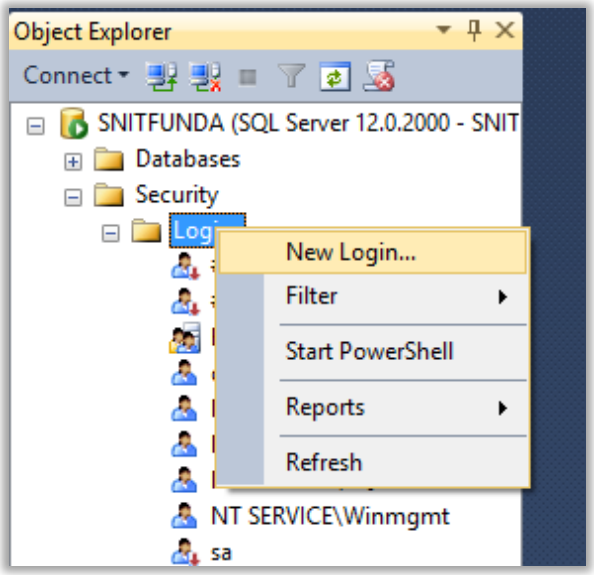
Select Options form the left panel on this dialog box that gives us options to select Collation (languages, culture), compatibility level etc. In general, we do not need to change anything into these options.

Clicking OK button creates a database.



4. How to create a new database user for SQL Server database?

To create a new database user, go to the Security folder and explore Logins sub folder and click New Login …



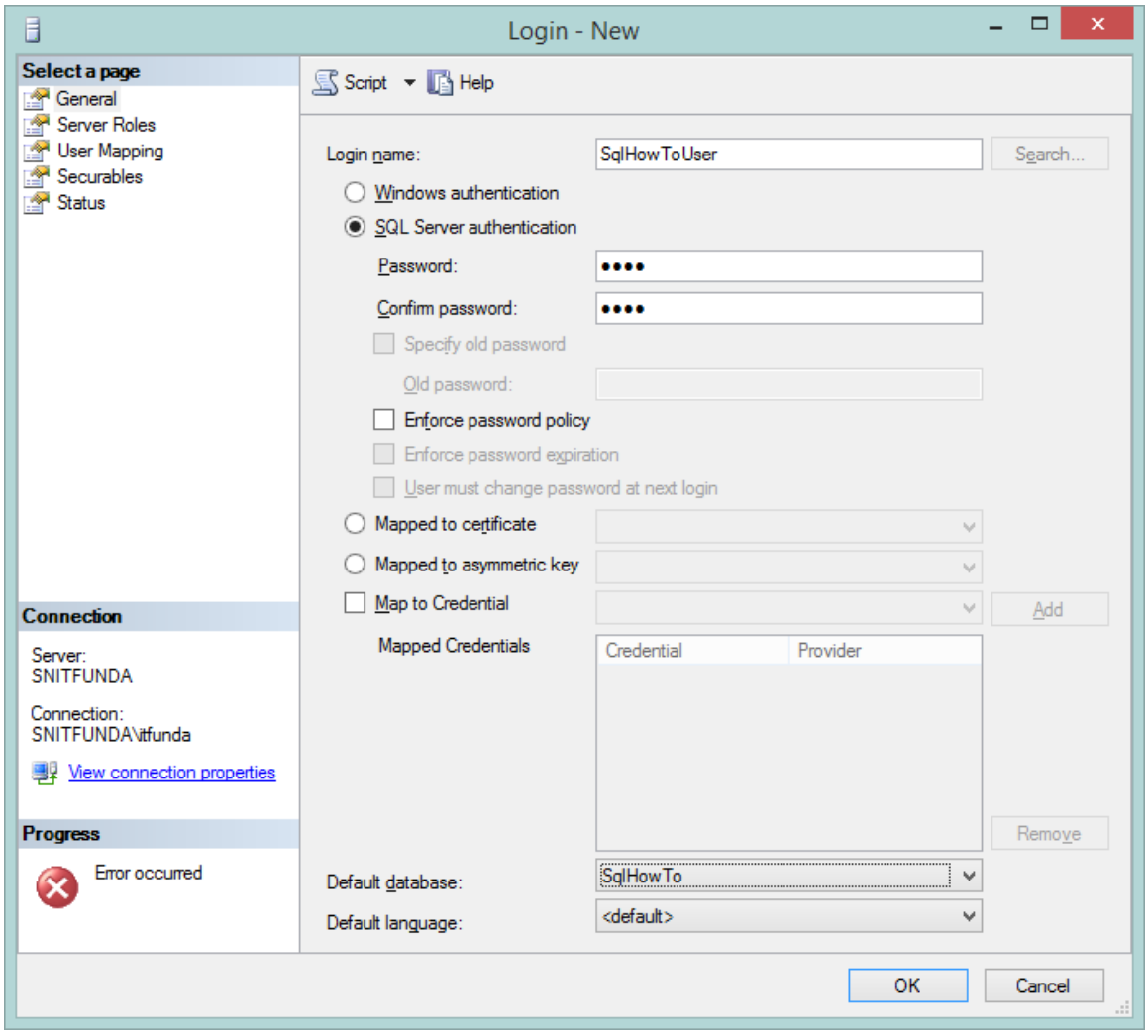
Clicking on the New Login – option gives us "Login – New" dialog box. Here we have ability to create user based on

I. Windows Authentication – that is nothing but the user of this machine or network machine
II. SQL Server authentication – a independent user that will be allowed to work on this database.

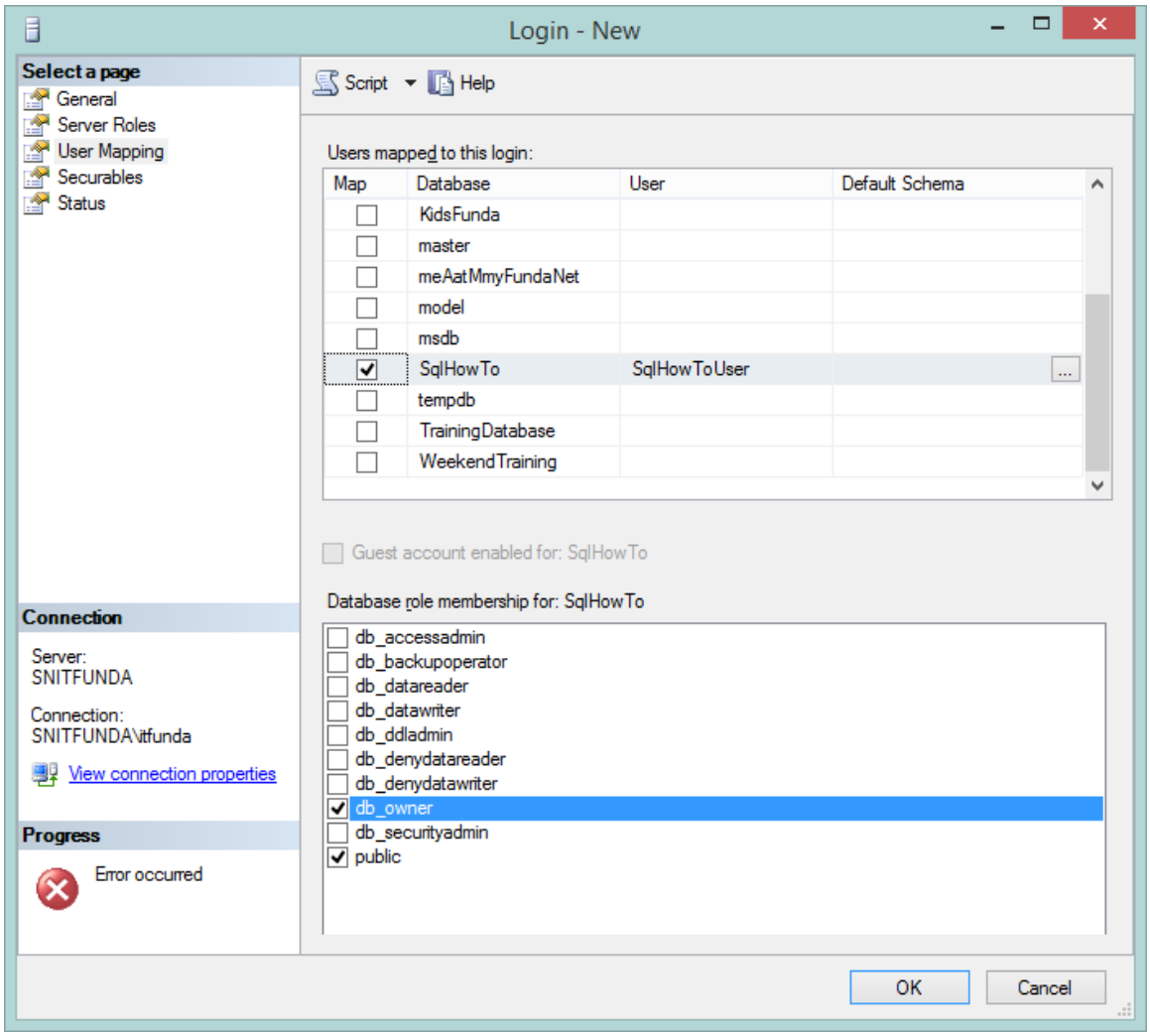
In this case, we will create a new user based on SQL Server authentication, write the Login name and then password.

In case we want to follow the standard policy of creating database password (to ensure that it is strong password that is combination of alphanumeric and special characters), keep the "Enforce password policy" check box checked otherwise uncheck it and write any password you want.

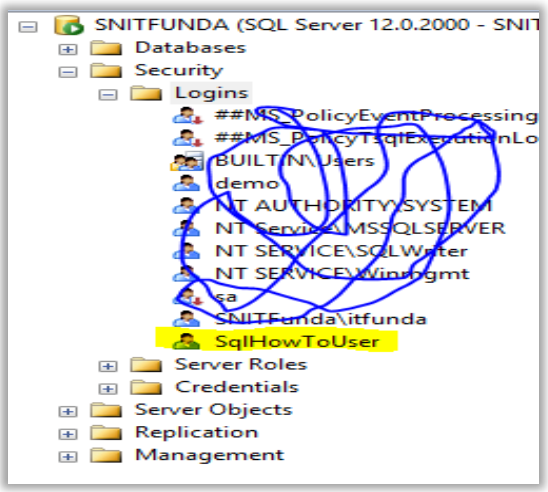
Now, in the "Default database" option, choose the database we want to create user for.



To allow this user to have complete access of selected database, go to "User Mapping" option from left panel and select the database for which we are creating user from the top box (check the checkbox against the database to use) and check "db_owner" checkbox from role membership list (check the db_owner checkbox) as shown below.



Clicking OK creates a new user under Logins folder as shown below.



5. How to specify permissions (Read/Write) for a specific user in the database?

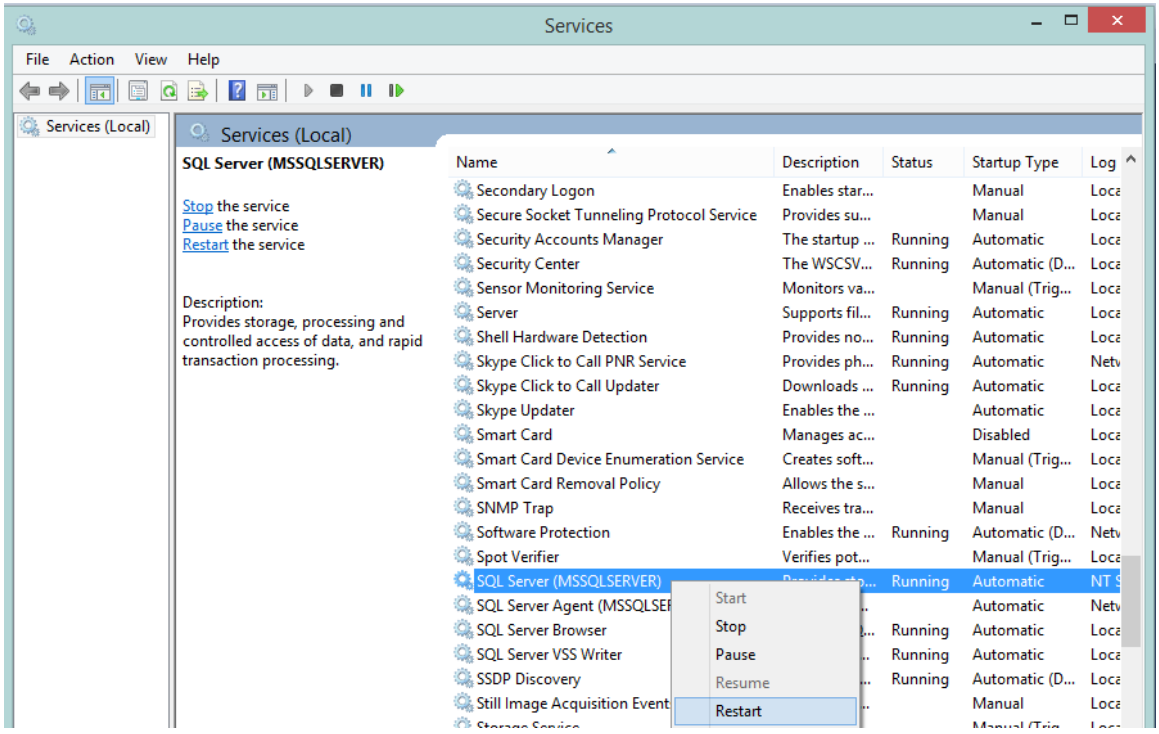
To specify the permission for a specific user in the database, right click the user in the Security > Login folder and select Properties. Now from the left panel, select Securables and select appropriate permission from the Permissions list and check appropriate checkbox against the Permission type. Apart from that we can also select database role by clicking the User Mapping options from the left panel. Follow the same approach as in the previous point of creating new user.

6. How to rename the database in SQL Server?

To rename a database from Management studio, right click the database from the Databases folder and select Rename. The name comes in editable mode, change the name and click Enter.

Please note that the database can't be renamed if the database is already being used in any application or some operations is being performed in the database.

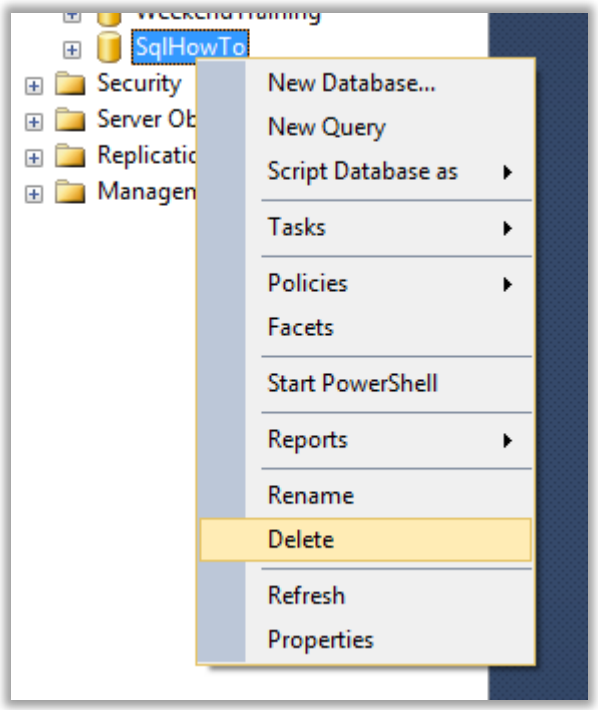
To force rename, we can stop the SQL Server service running on the machine by going to the Services from control panel and restarting it or stopping it.



This ensures that any users of the database blocked and all process going on have been aborted. Then try to rename the database and it should be renamed.

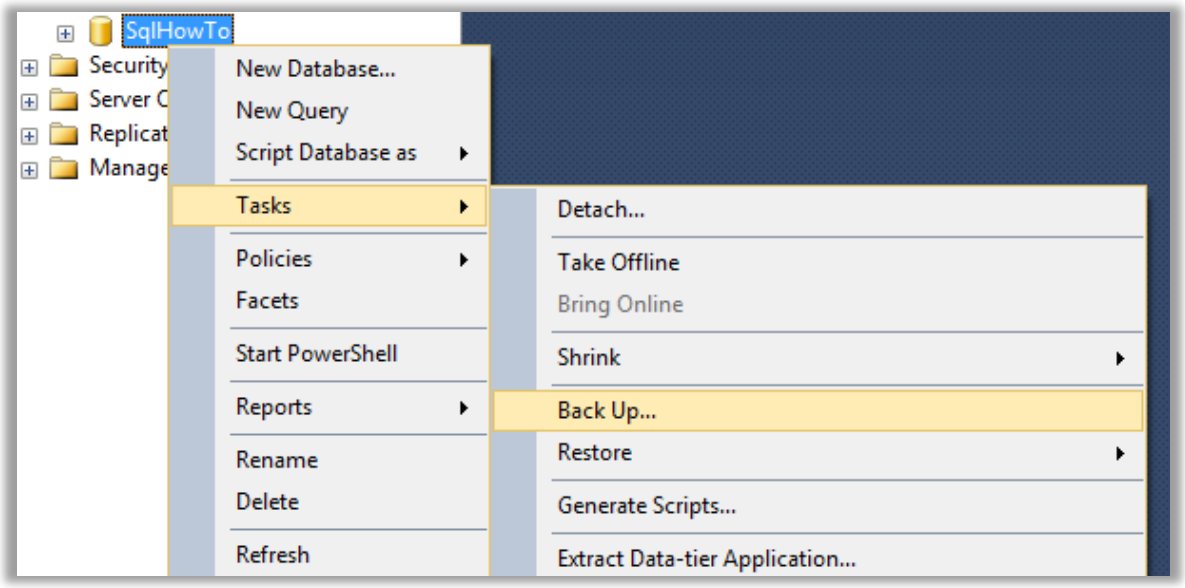
7. How to drop the database from SQL Server?

In order to Drop / Delete the database for SQL Server, right click the database and click Delete. In this case also we need to ensure that the database is not being used currently. To ensure that database is not being used, re-start the SQL Server process from Services as described in above points and try to delete it.

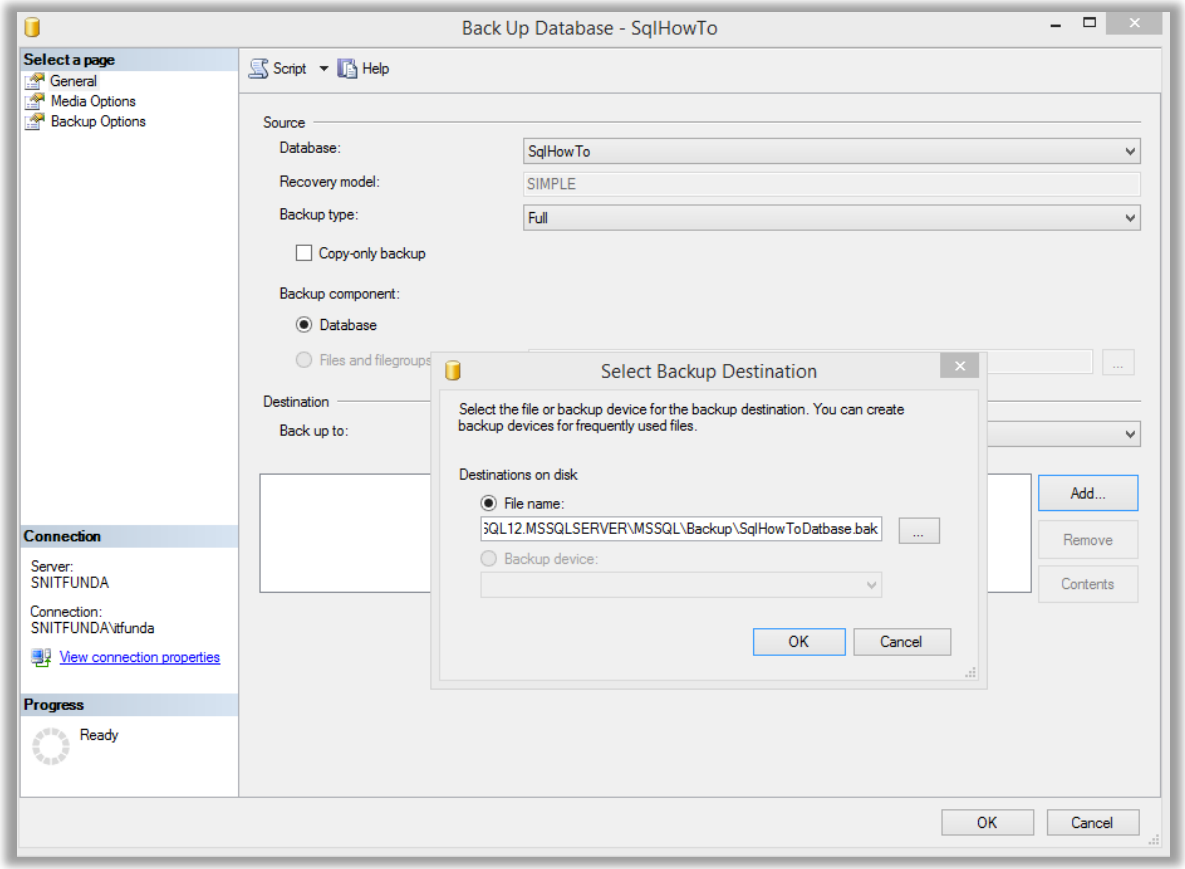


8. How to take backup of the database in SQL Server?

To take back up of the database in SQL Server, right click the database and select Taks > Back Up…



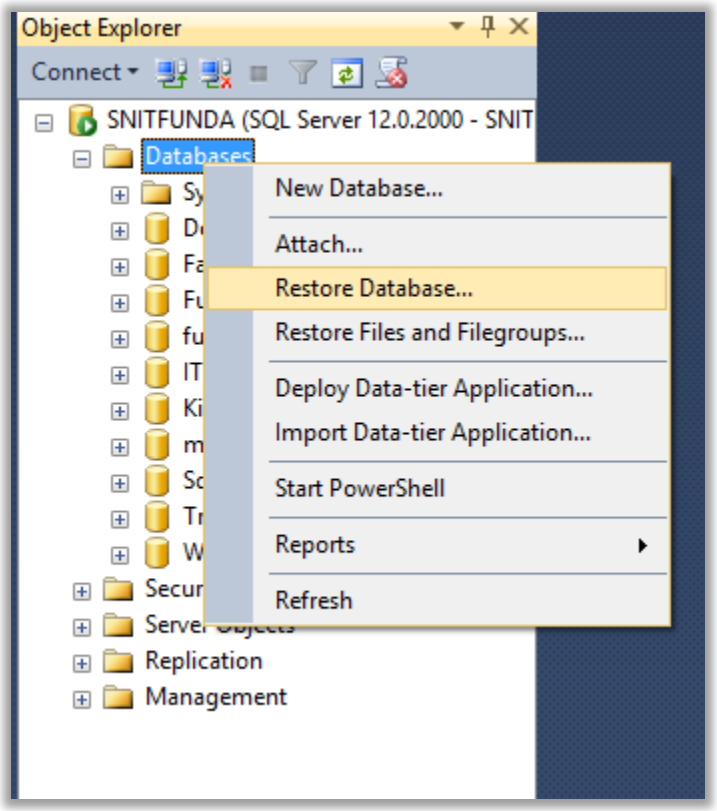
This shows the Back Up Database dialog box as shown below. Now click on Add.. button to add the Destination folder and file name where we want to take the backup of the database.



Click OK that starts the backup and creates a .bak file at the selected location (generally backup file extension is .bak).

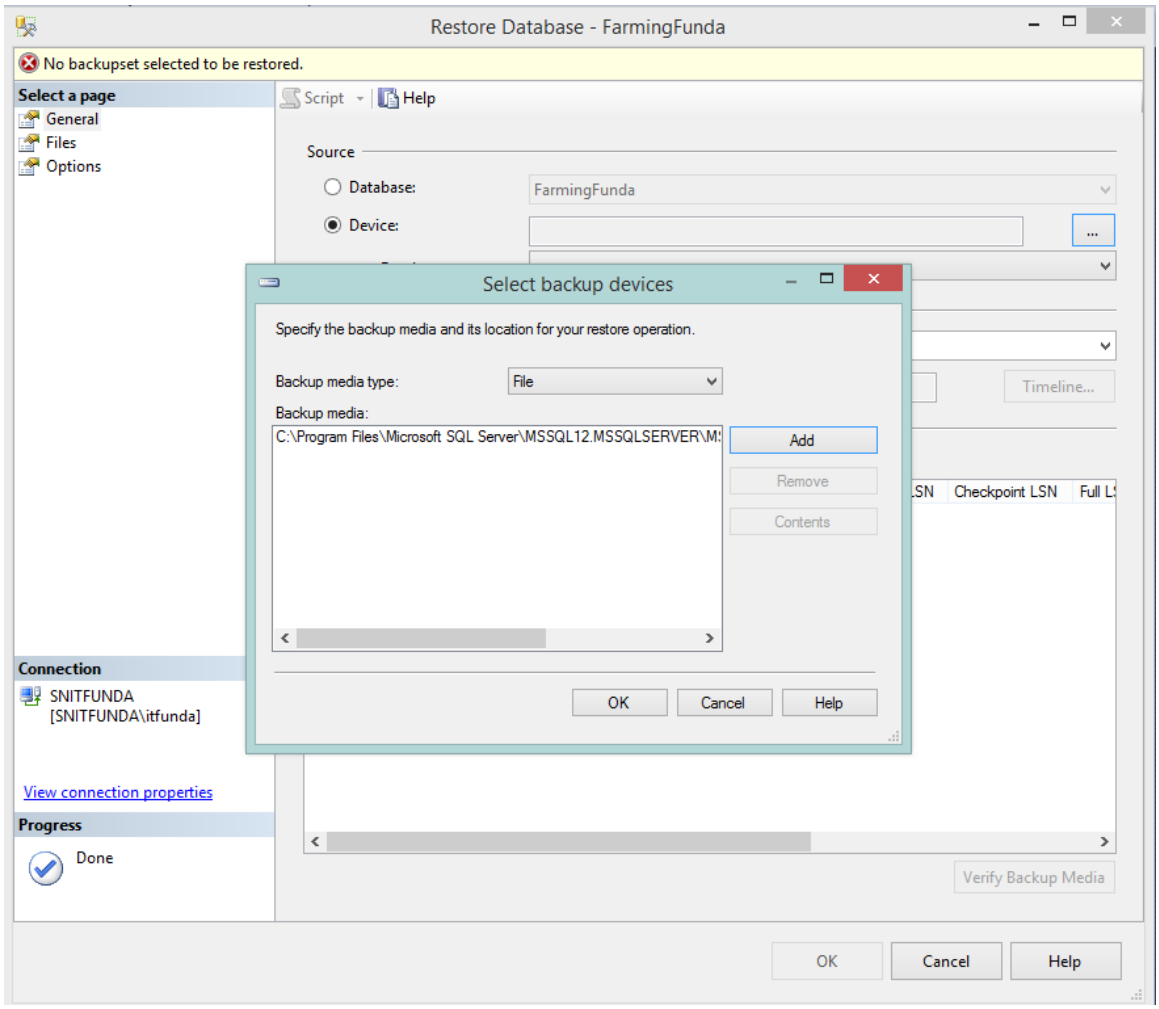
9. How to restore a database in SQL Server?

To restore a database in SQL Server from the backup done using previous points (.bak file), right click Databases folder form the left side panel (Object Explorer) and select Restore Database…

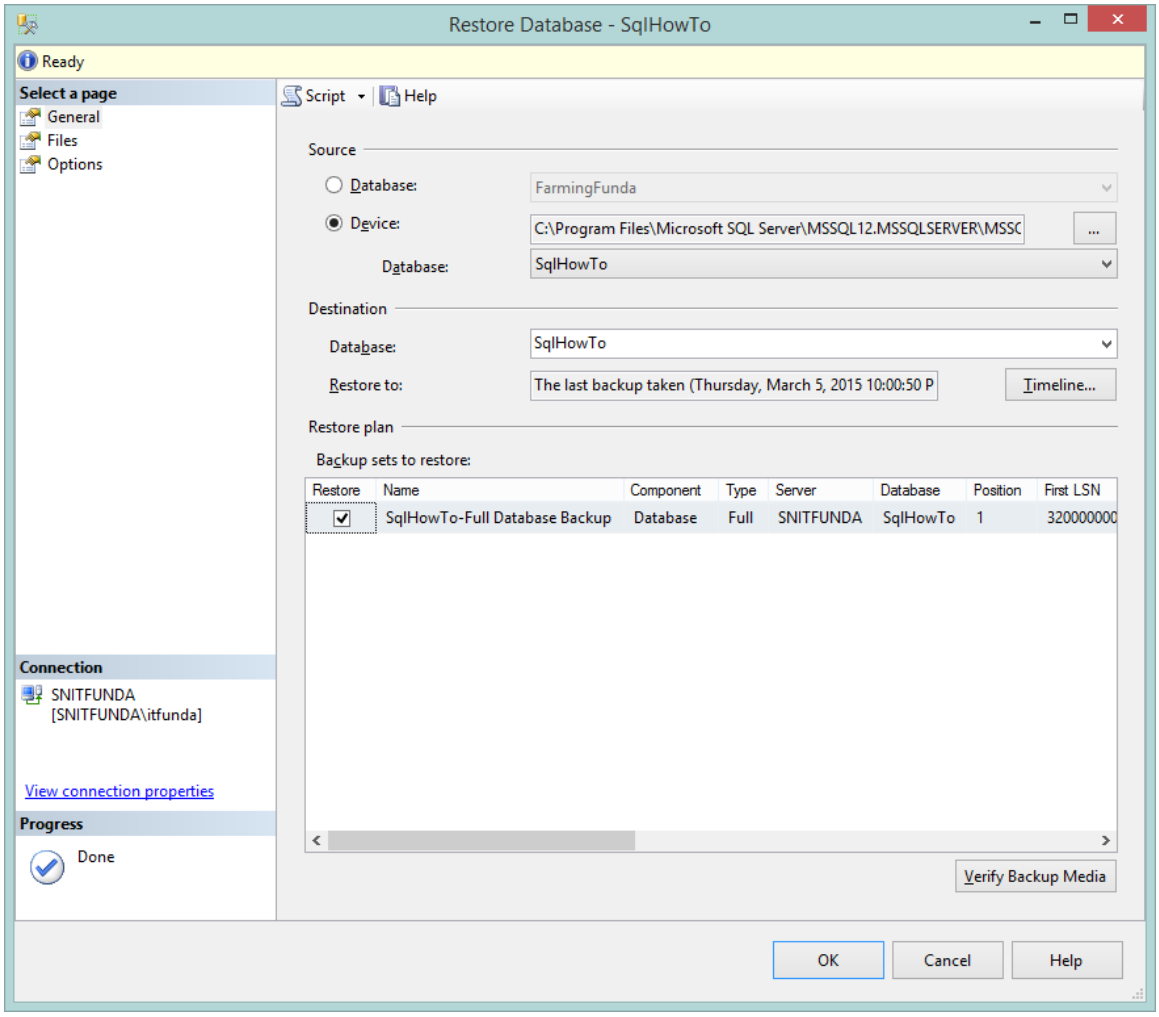


This opens up Restore Database dialog box, now select the database to restore by selecting the 2nd radio button (Device) under Source heading and click on … button against it as shown in the picture. That opens up another dialog box to Select backup device. Now, select the media type (File – in case we have .bak file at the local or network machine and URL – in case we have the .bak file on any webserver)

Assuming, our .bak file is in the local machine, click on Add button and select the .bak file of the database to restore.



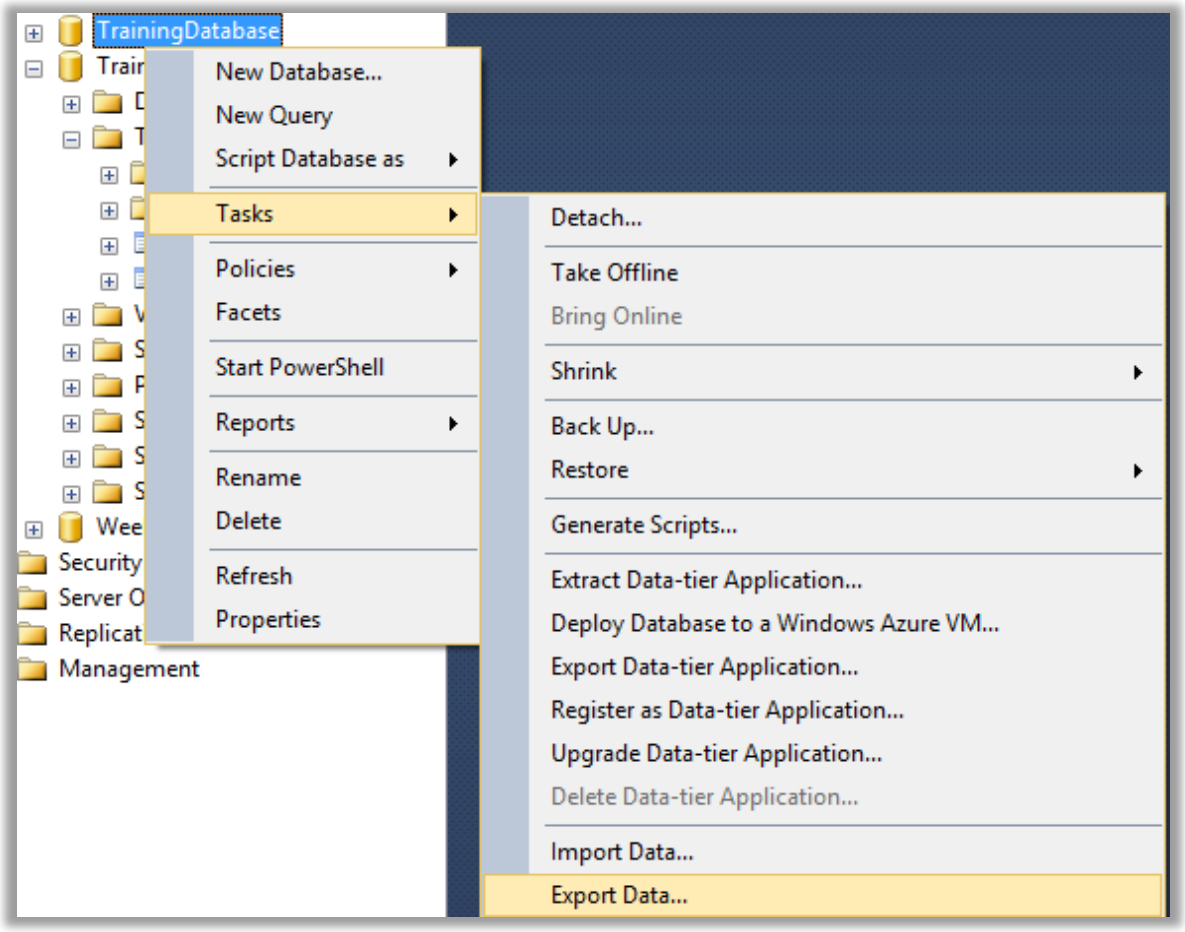
Now, select the Database on which to restore from the existing database list (to override the previous database) under Destination or write the name of the database in the DropDown to create a fresh database using this .bak file.



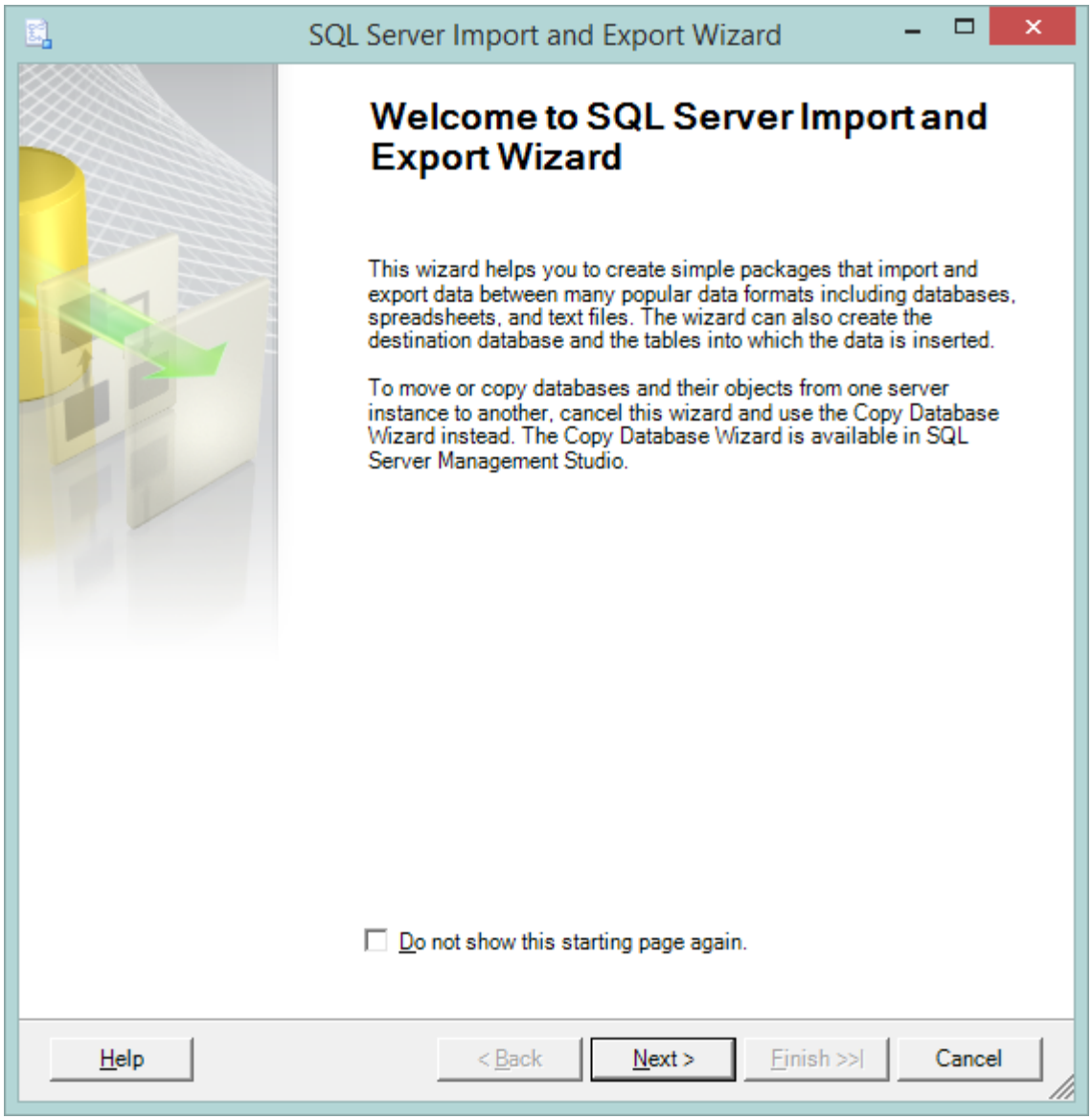
Now, click OK that restores the database depending on whether new name is given to existing database is selected. Note that in order to restore on the existing database (override), the database must not be in use.

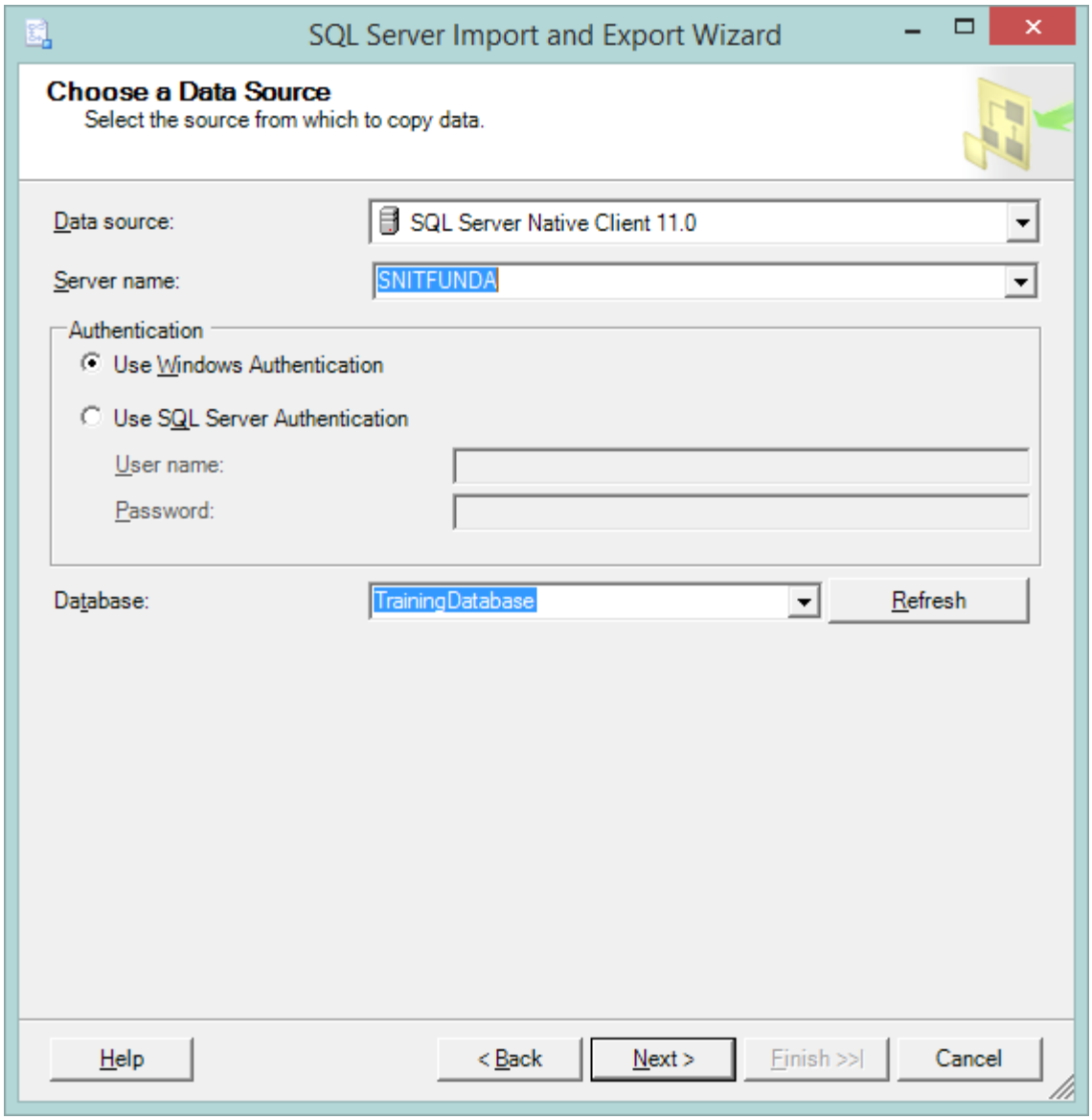
10. How to export the database from SQL Server?

To export database, select Export Data … option by right clicking the database to export and going to Task > Export Data … option.



Select the Data source to export the data from, to export from the SQL Server, select SQL Server Native Client xx.x. Now, select the Server name. Choose the authentication type, depending on what type of authentication we have for the selected server. Finally, select the Database to export.

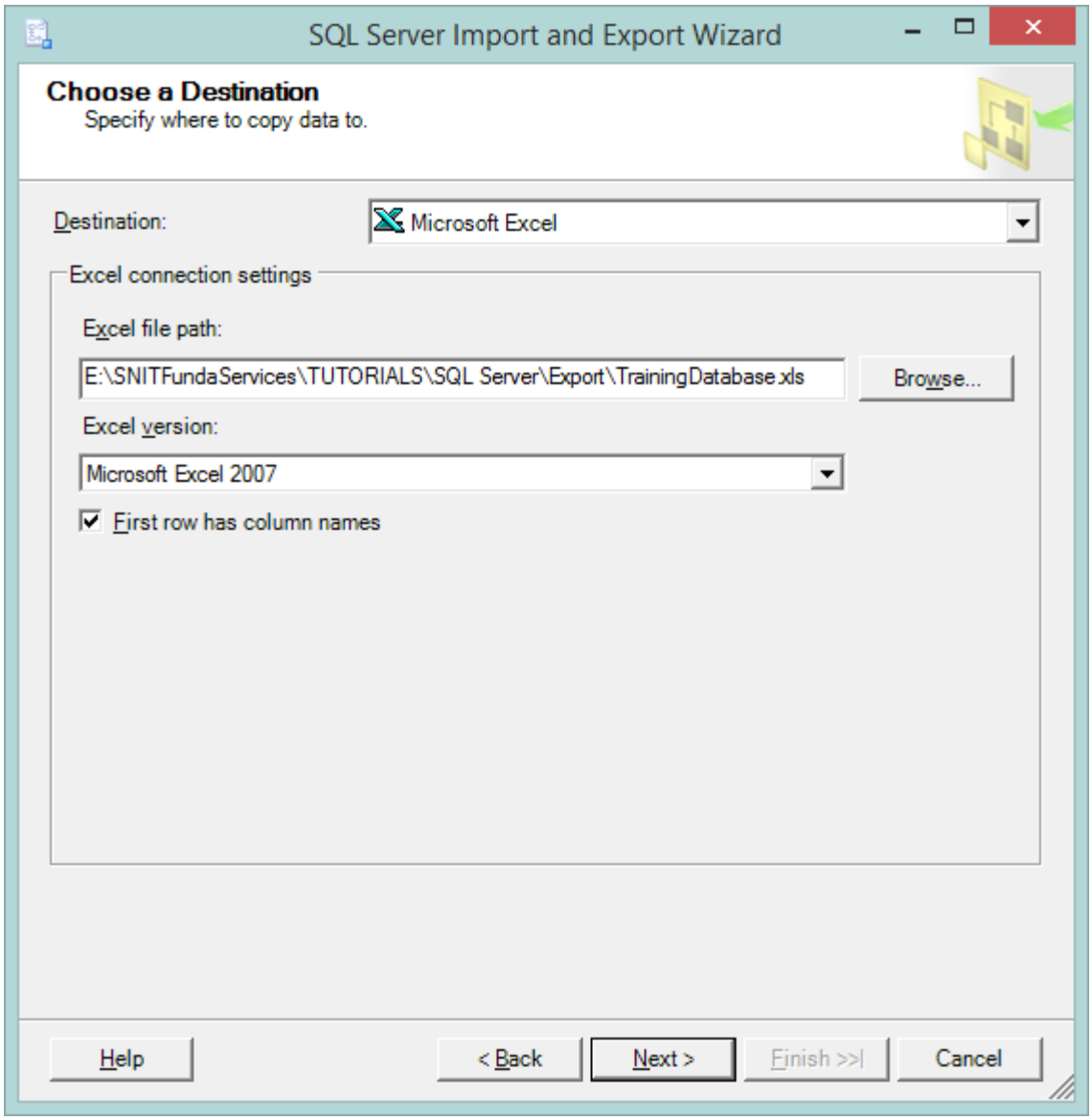

SQL Server Import and Export Wizard
Welcome to SQL Server Import and Export Wizard
This wizard helps you to create simple packages that import and export data between many popular data formats including databases, spreadsheets, and text files. The wizard can also create the destination database and the tables into which the data is inserted.
To move or copy databases and their objects from one server instance to another, cancel this wizard and use the Copy Database Wizard instead. The Copy Database Wizard is available in SQL Server Management Studio.
Do not show this starting page again.
Help
< Back
Next >
Finish >>|
Cancel



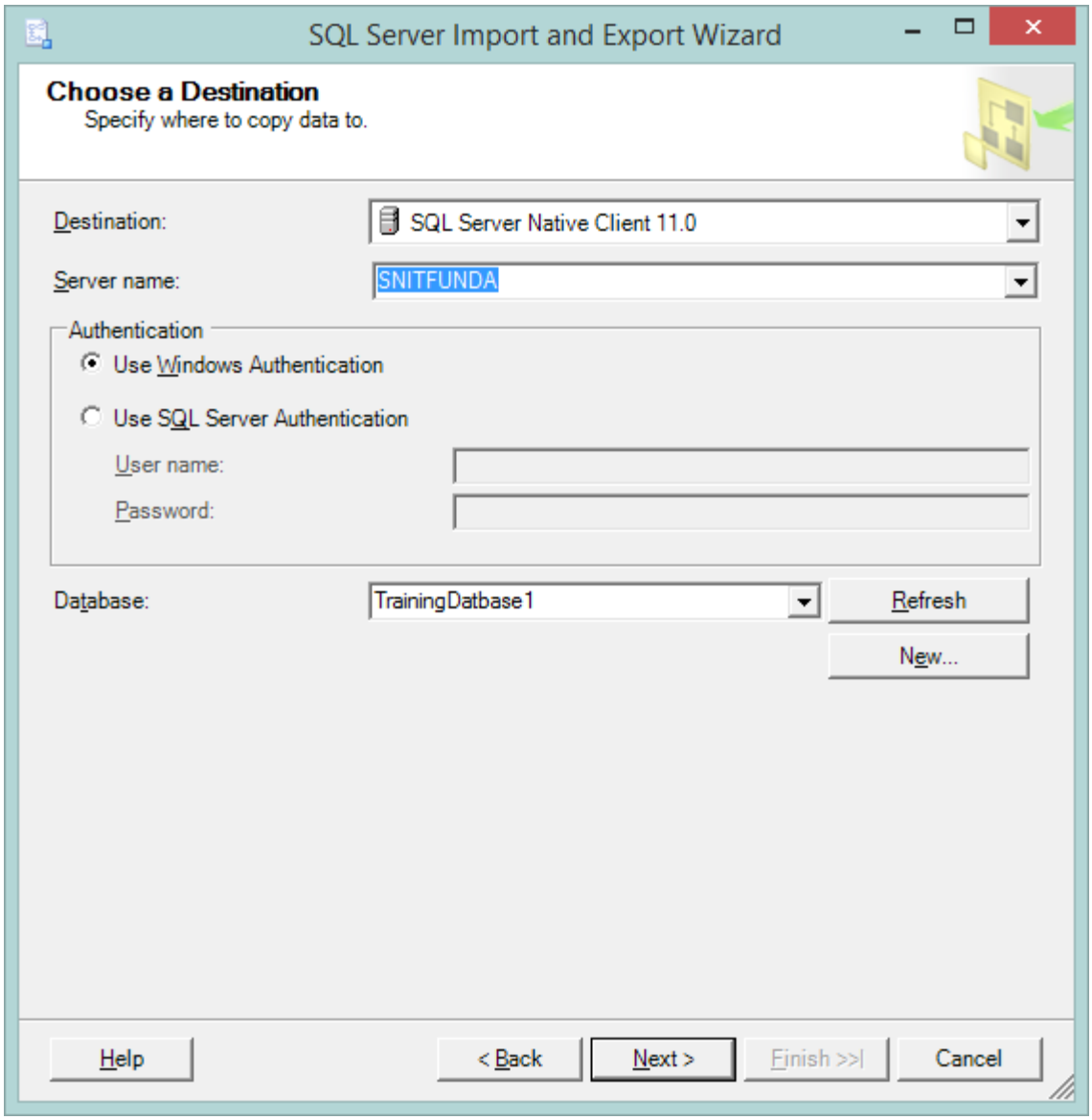
Click Next > that opens up another dialog box that allows us to select the destination. Depending on where we want to Export, select it from the Destination dropdown. In this case, we have selected Microsoft Excel.

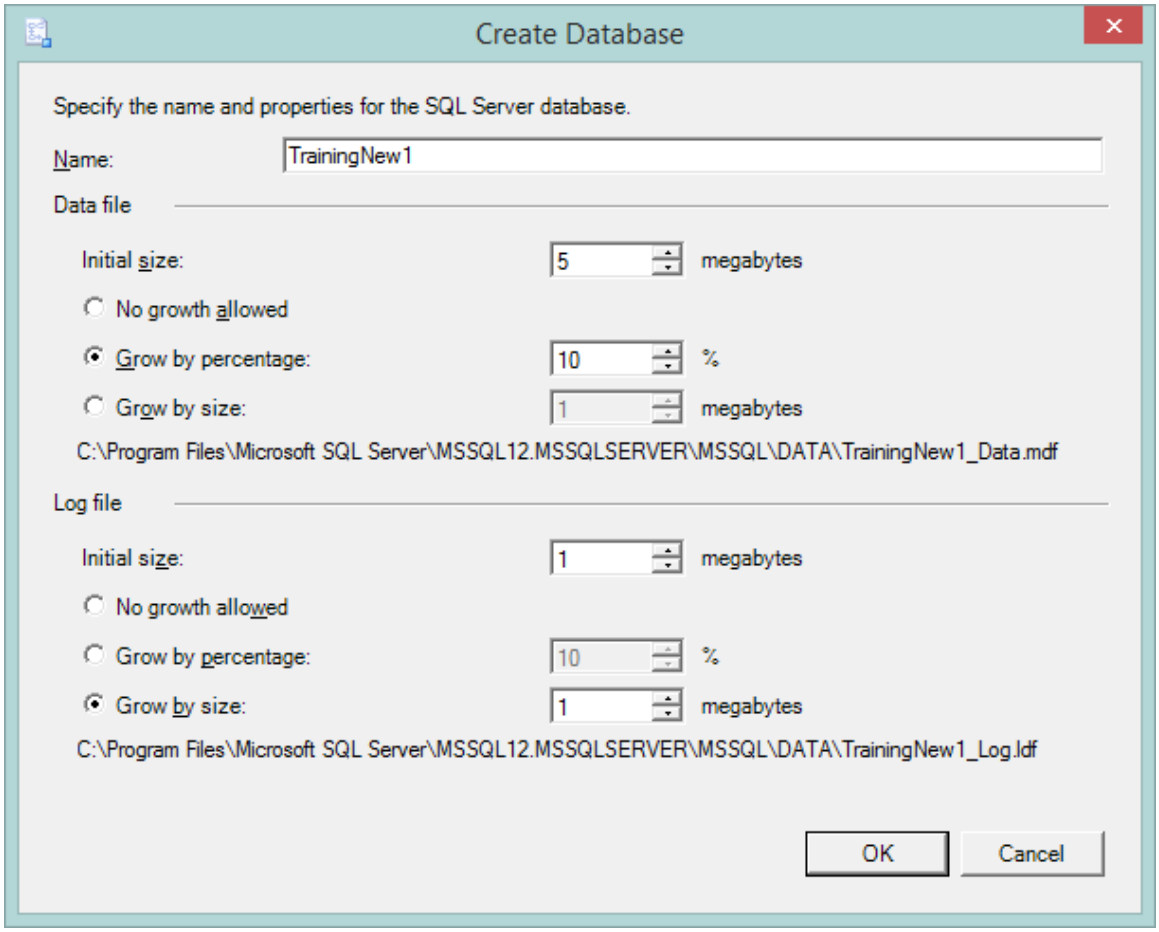
Now write the path and file name where we want to save the exported database and select the Excel Version to export into.

Now click Next >. If proper Excel provider is installed; it will export the data into MS Excel format.

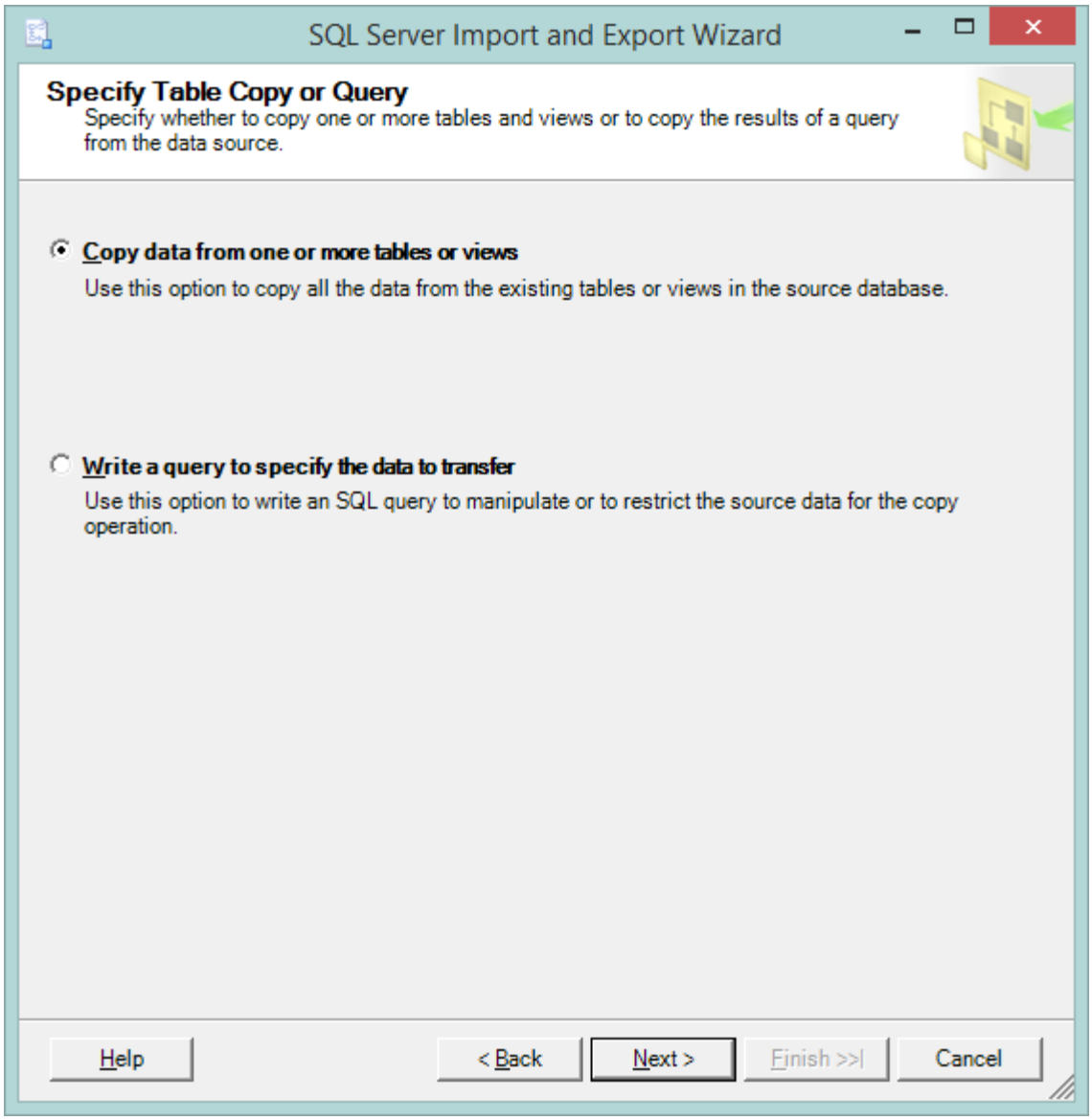


In case, we want to export the database on to another SQL Server, select SQL Server Native Client xx.xx from the Destination dropdown. Select the Server name and Authentication type. Now, select the Database on which we want to override or click New … button and write the completely new database name into the Name box and click OK.

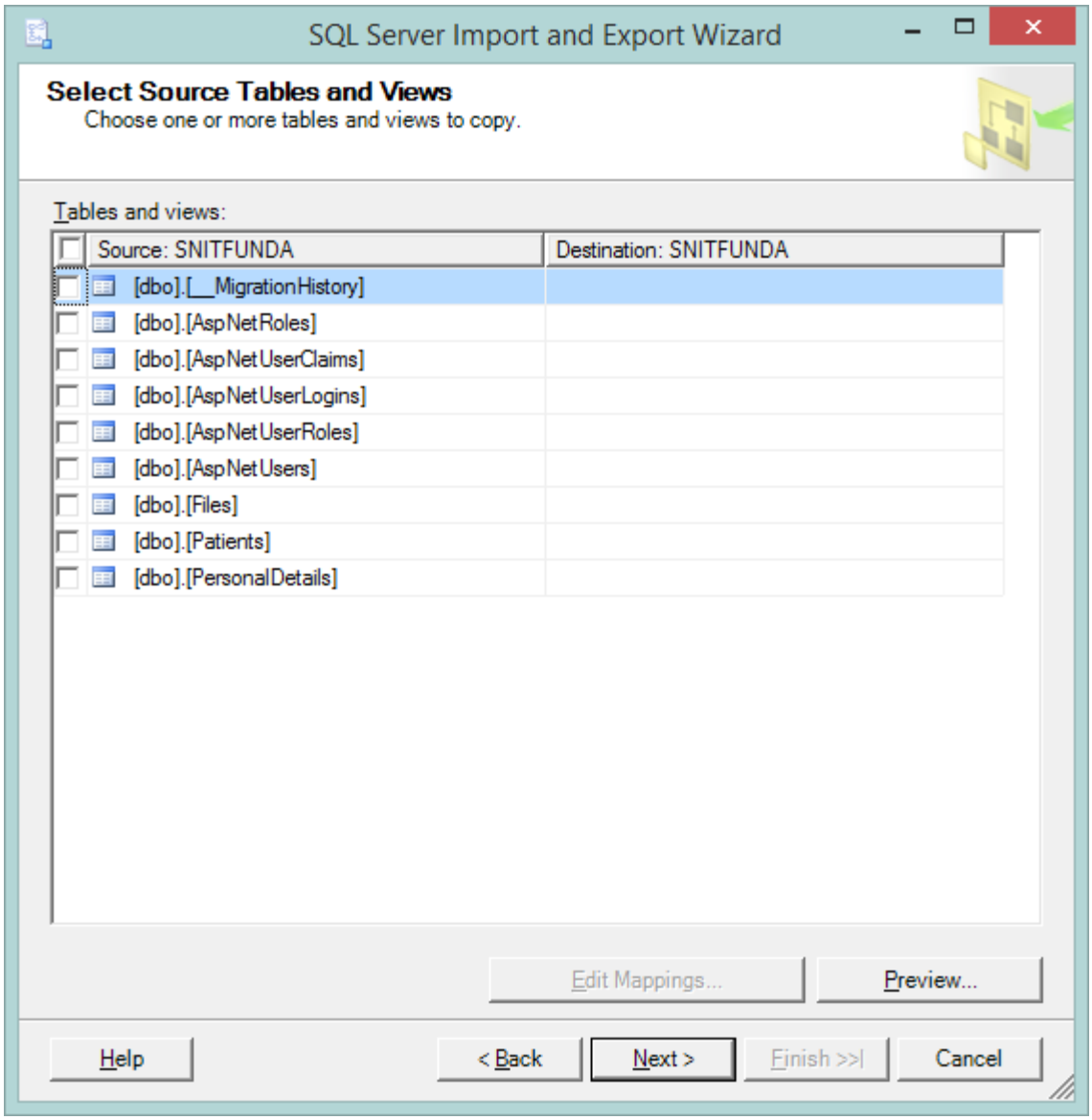

SQL Server Import and Export Wizard
Choose a Destination
Specify where to copy data to.
Destination:
SQL Server Native Client 11.0
Server name:
SNITFUNDA
Authentication
Use Windows Authentication
Use SQL Server Authentication
User name:
Password:
Database:
TrainingDatbase1
Refresh
New...
Help
< Back
Next >
Finish >>|
Cancel



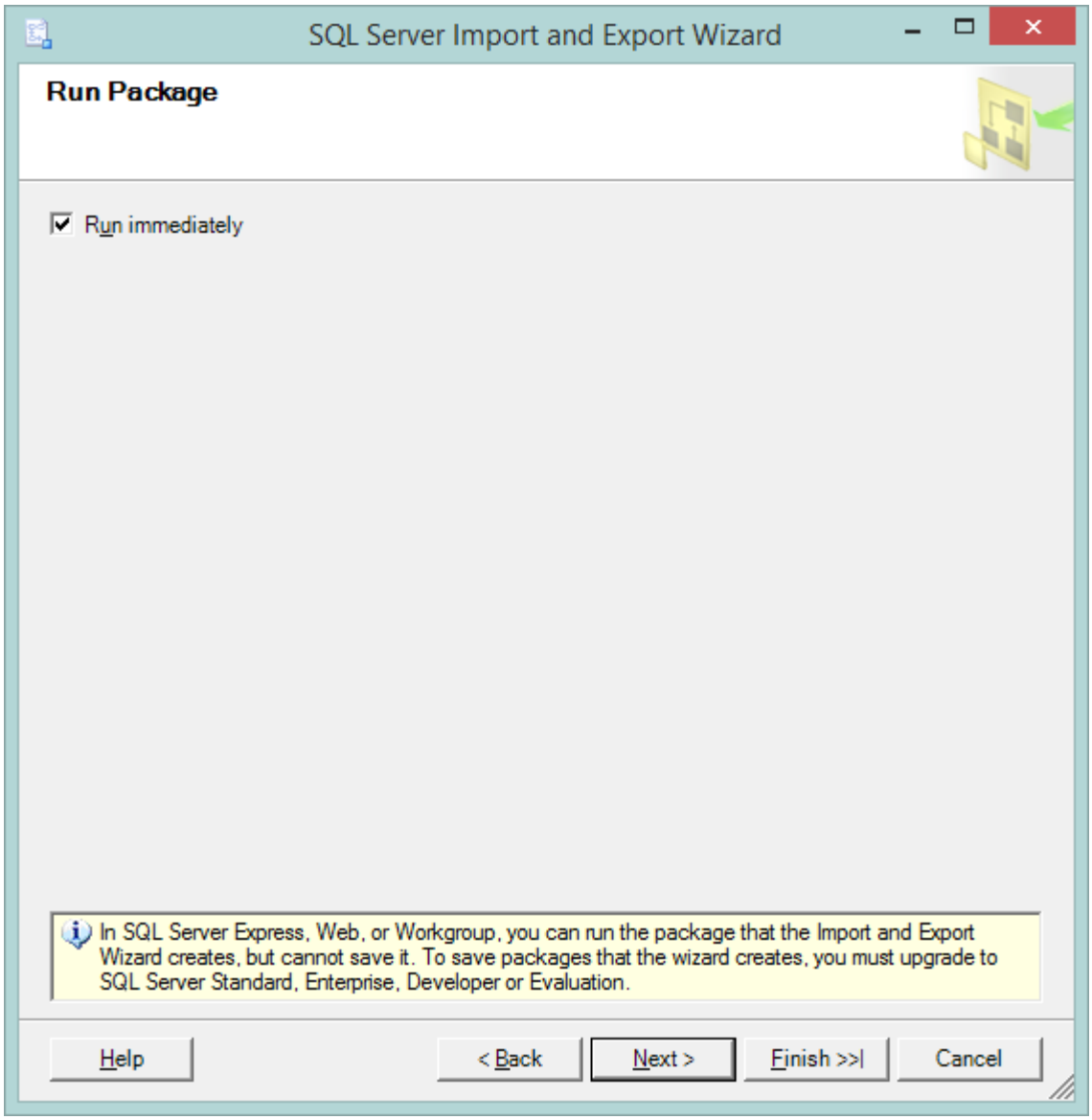
Now, click Next > button on the Choose Destination dialog box. Now in the following dialog box, select the first Radio button as shown below that will copy all tables and views to the new database.



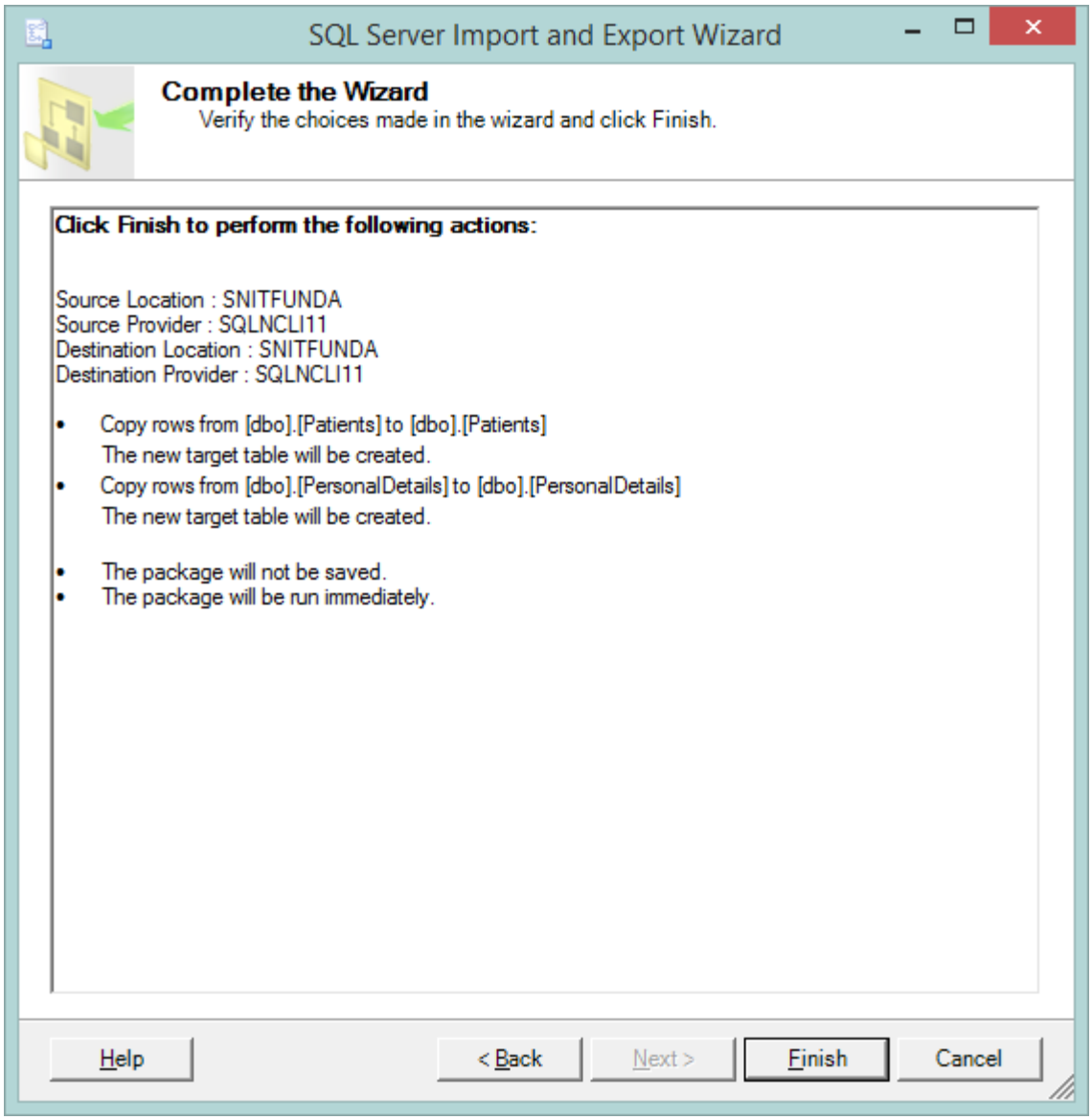
Click on Next >



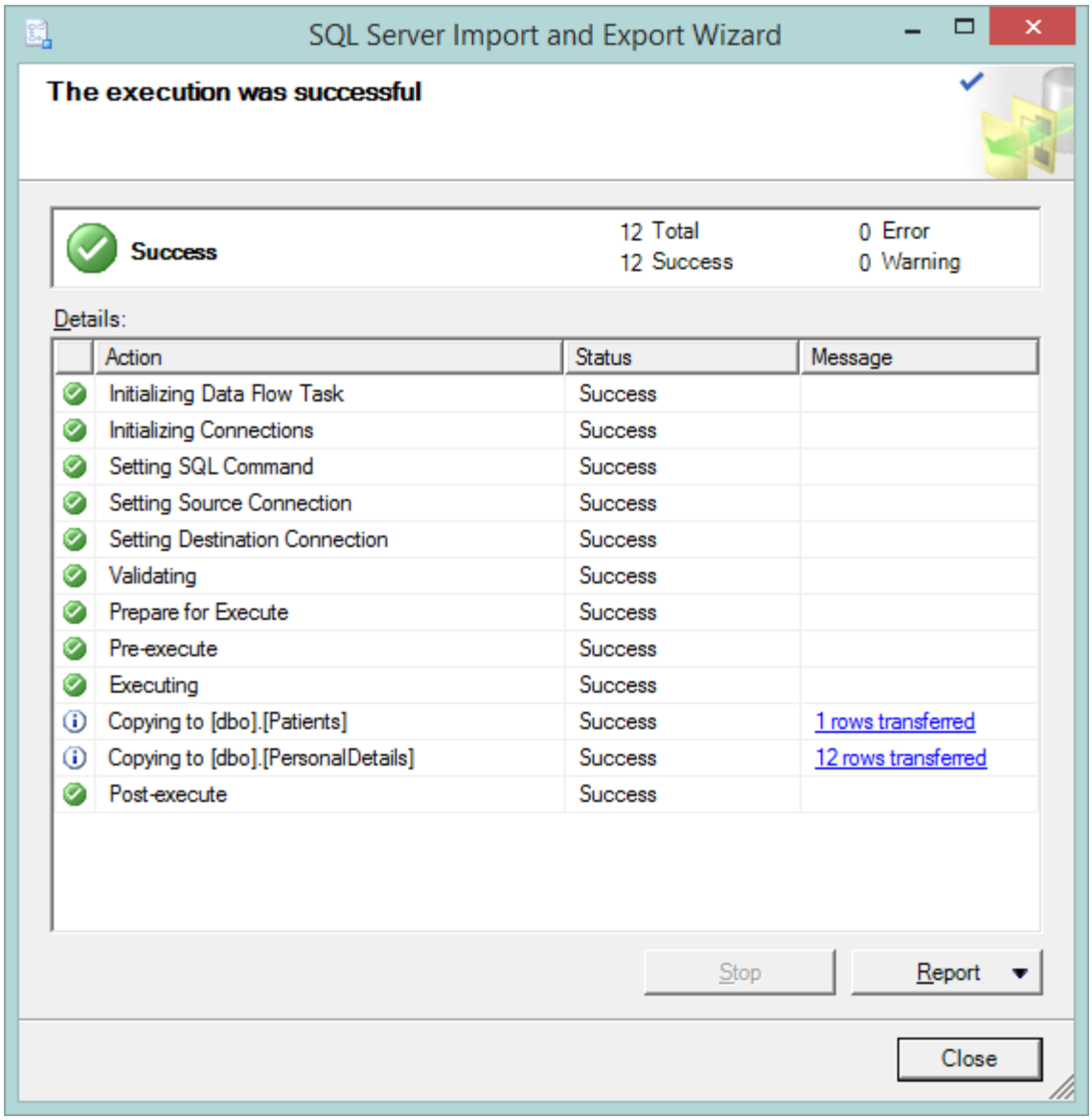
Select the database tables from the list of tables from the source database. Click Next >



Now, click Next > and then Finish.



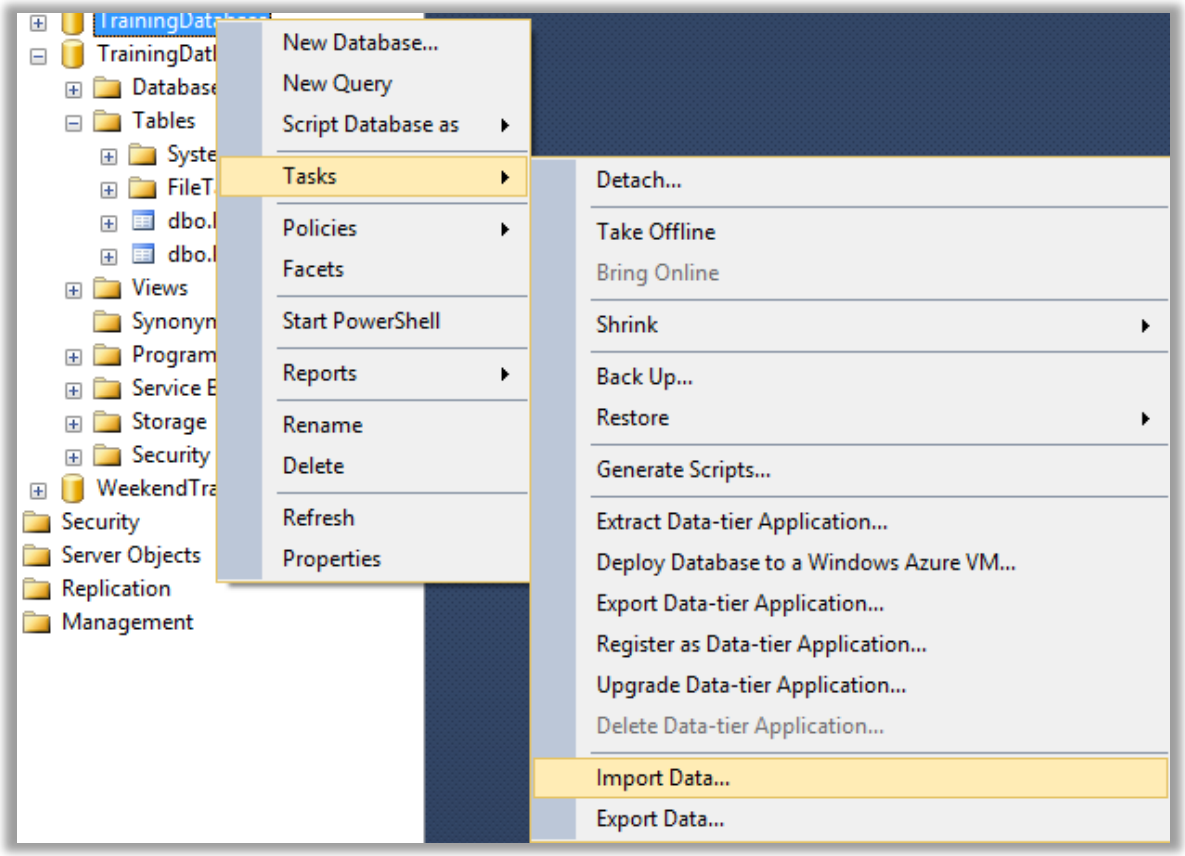
Clicking Finish, gives us the result like below that shows the success or failure status of each command executed to export the databse.

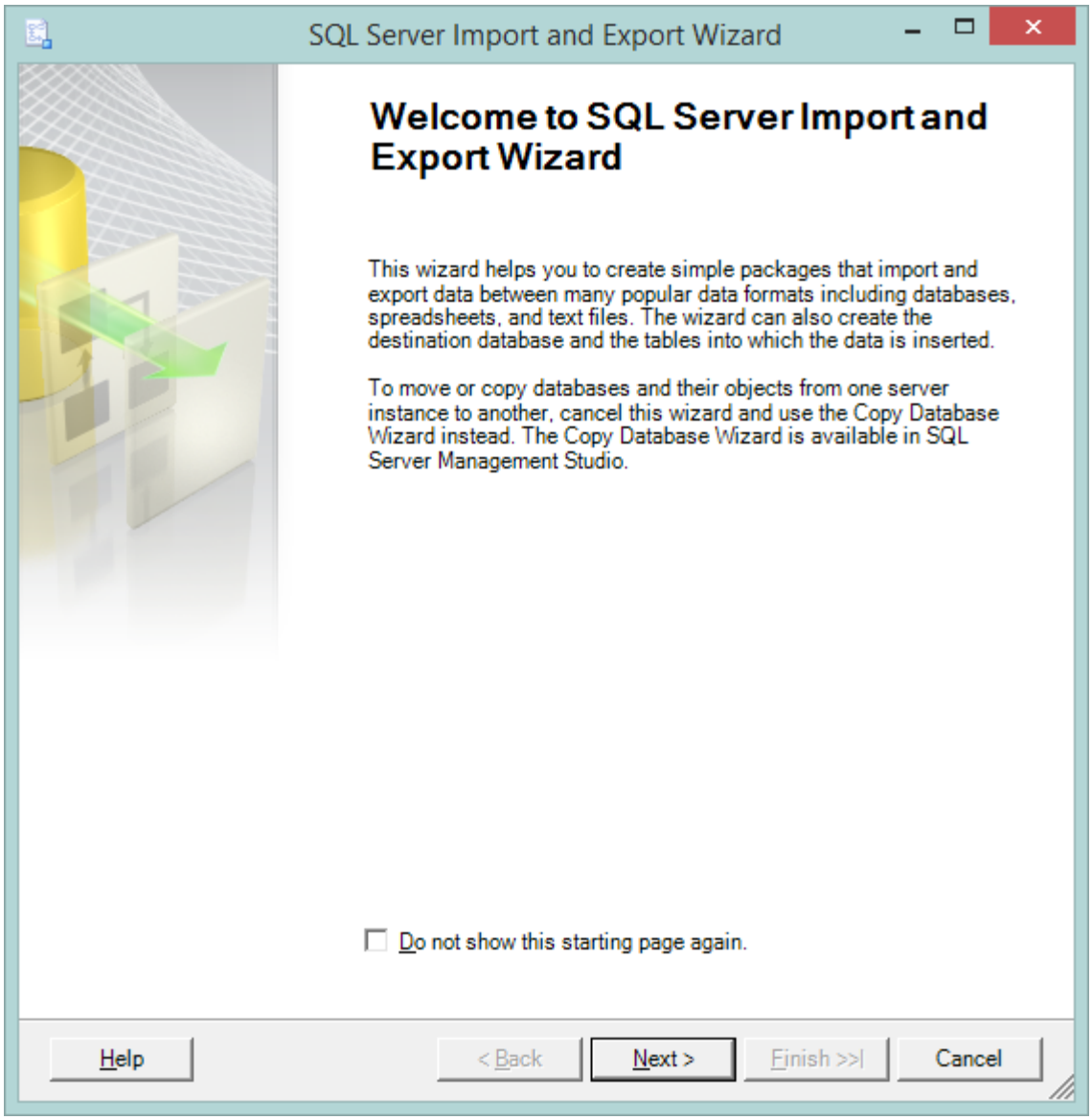


Note that exporting database will not export the Stored Procedure or functions from the source database, to create a exact copy of the source database, use Backup and Restore as explained in above points.

11. How to import the database into SQL Server?

Importing database from other sources or other database server to current database server is almost same as exporting the database. Just select Import Data … option by right clicking the Database and going to Tasks.

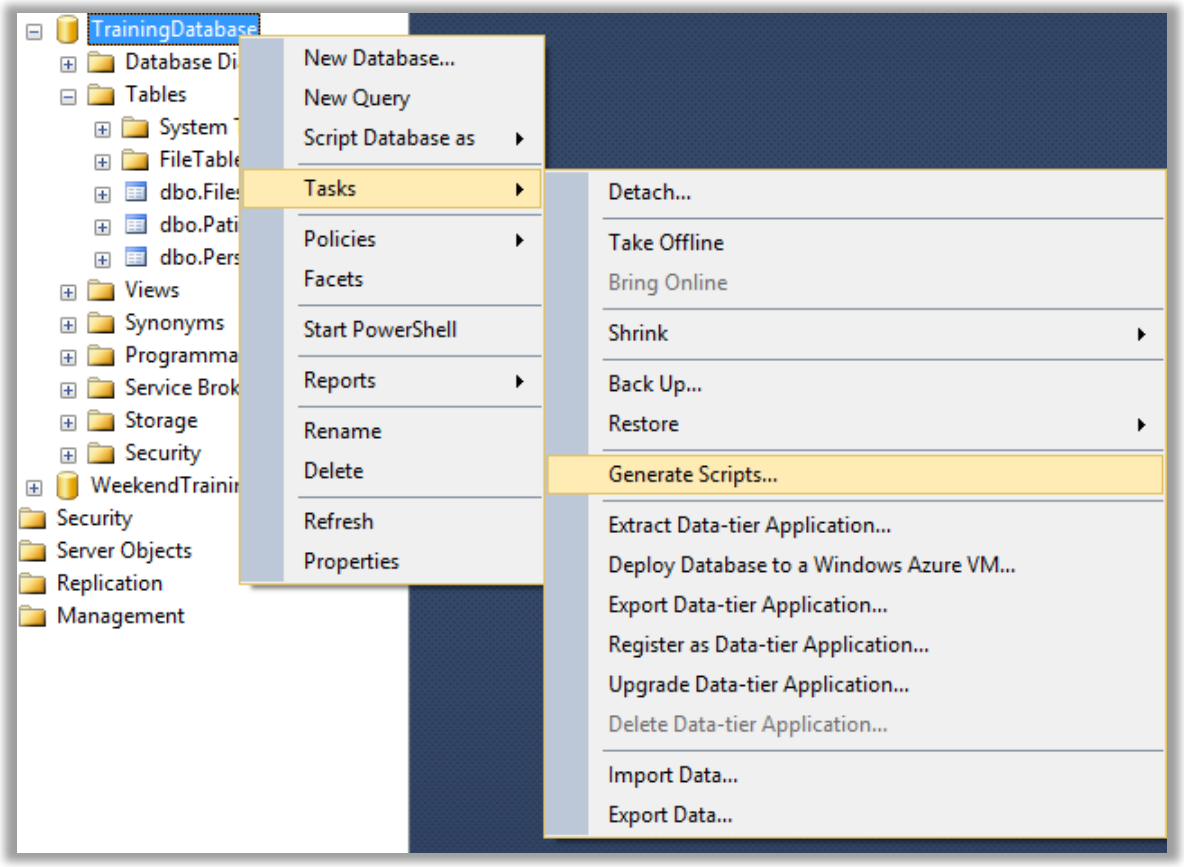

New Database...
New Query
Script Database as
Tasks
Policies
Facets
Start PowerShell
Reports
Rename
Delete
Refresh
Properties
Detach...
Take Offline
Bring Online
Shrink
Back Up...
Restore
Generate Scripts...
Extract Data-tier Application...
Deploy Database to a Windows Azure VM...
Export Data-tier Application...
Register as Data-tier Application...
Upgrade Data-tier Application...
Delete Data-tier Application...
Import Data...
Export Data...
Tables
Views
Storage
Security
Security
Server Objects
Replication
Management



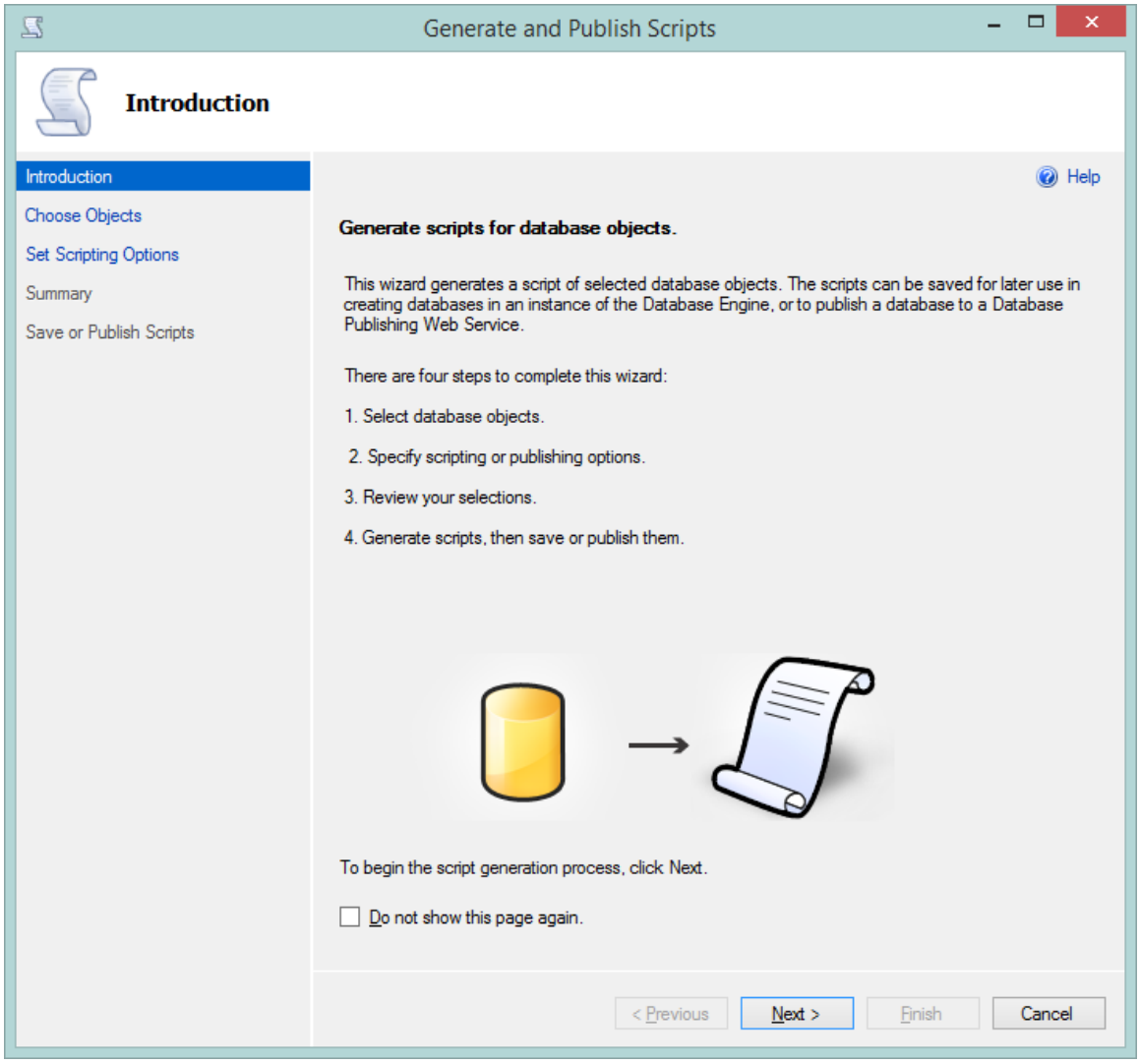
Click Next > and then follow the steps almost similar to the Export Data ….

12. How to generate SQL script of the whole database structure in SQL Server?

To generate script of the whole database structure so that the same database structure can be created on another server with no data, we can right click on Database and select Tasks > Generate Scripts…

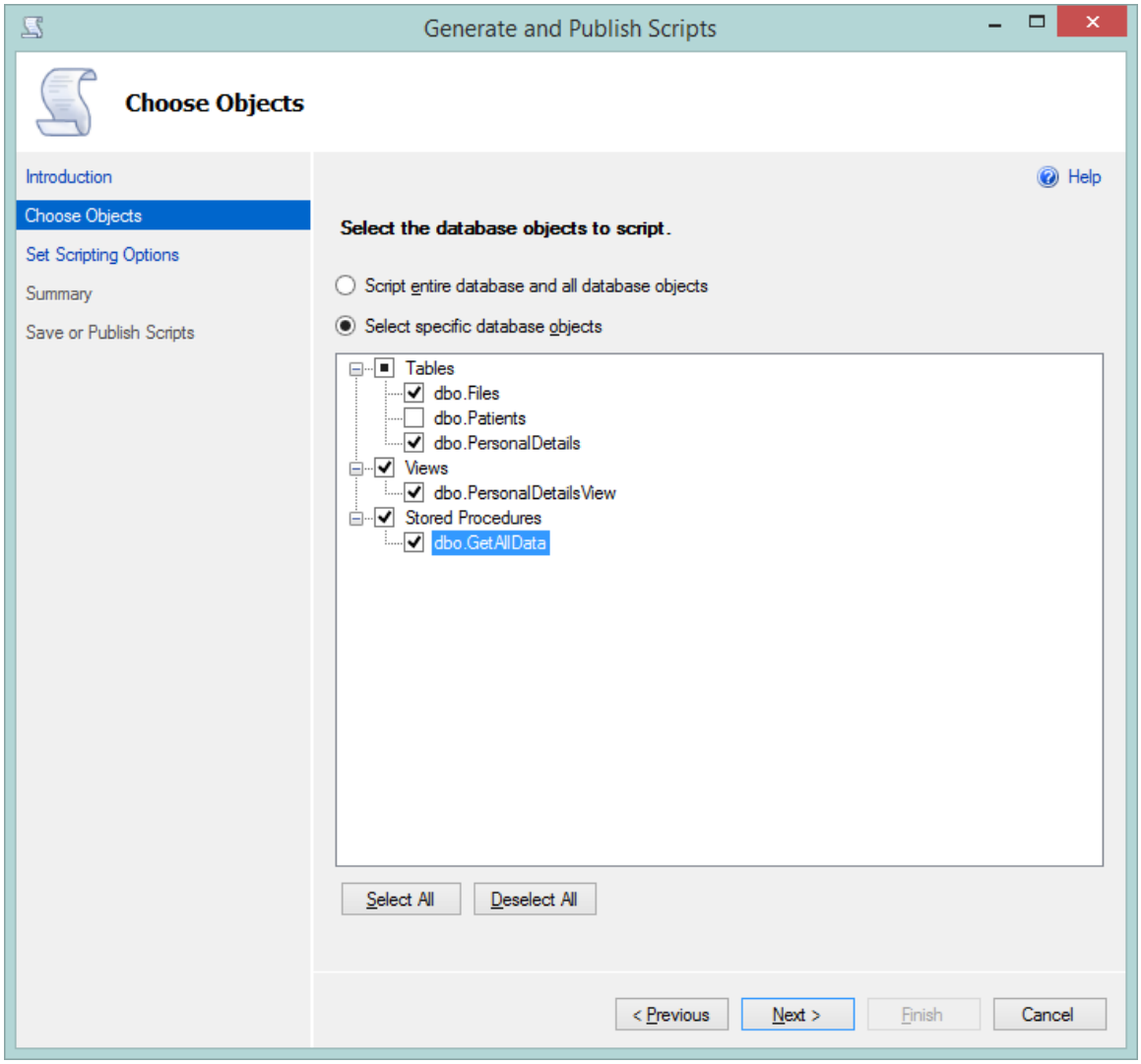


Select Next > on below dialog box.



In the following dialog box, select whether we want to script all database objects like Database tables, stored procedures, views or want to be very specific on which database object we want to generate script for.

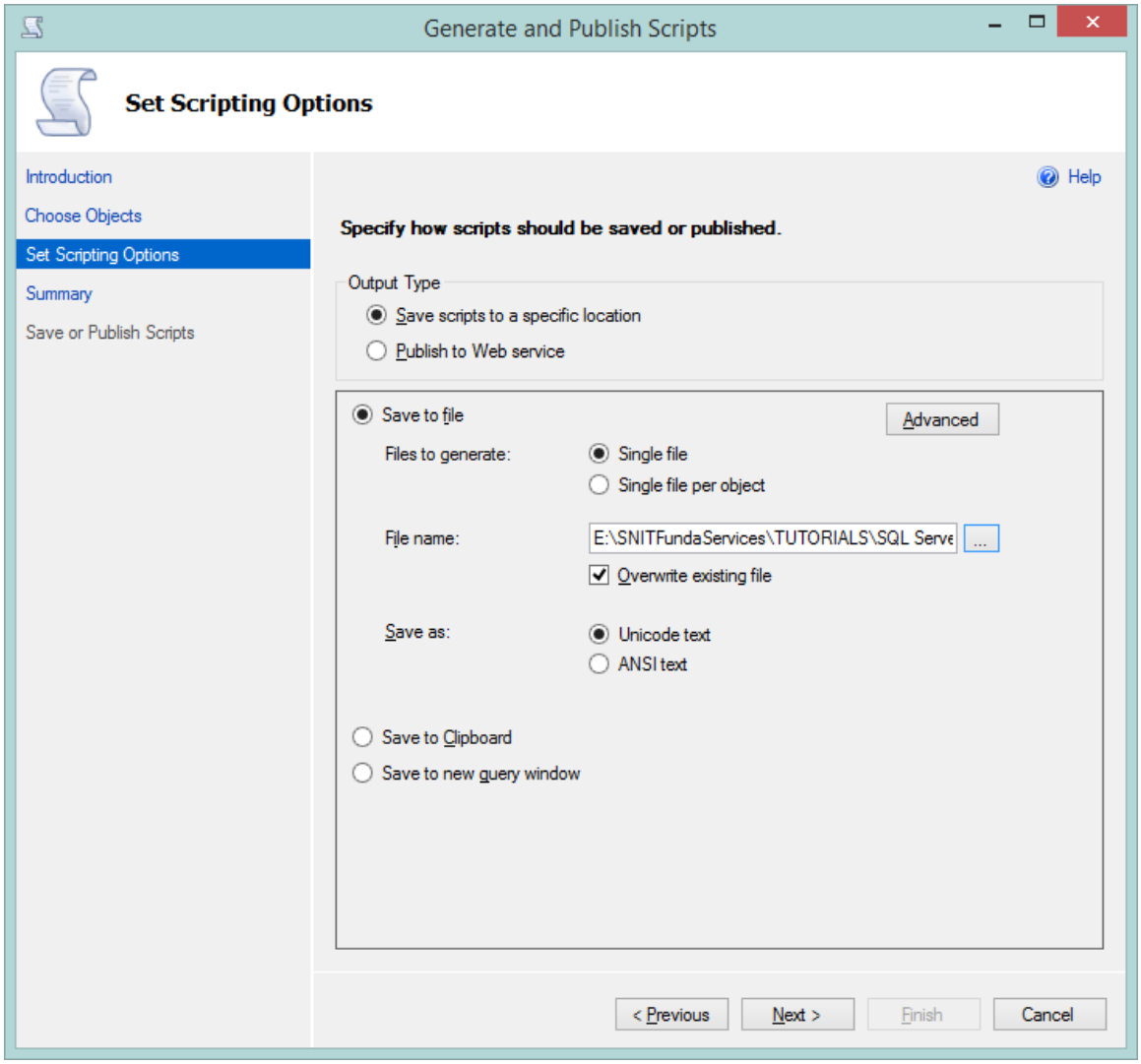
To be specific, select 2nd radio button and select appropriate Table, Views, Stored Procedures



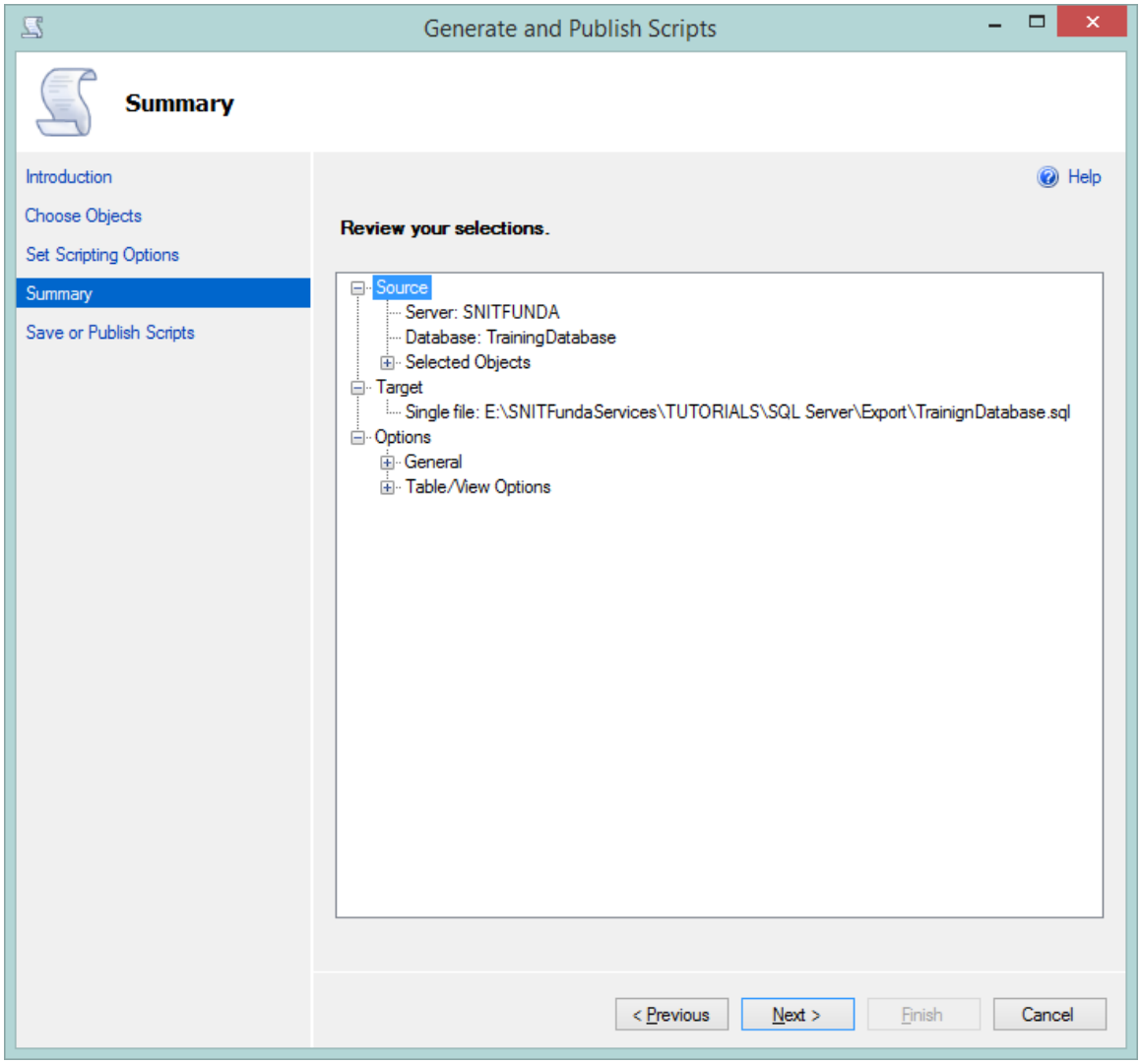
Click Next > and select proper Output type.

To save the file on the current system, select first radio button from the Output Type and then write the file name where we want to save the generated script file.

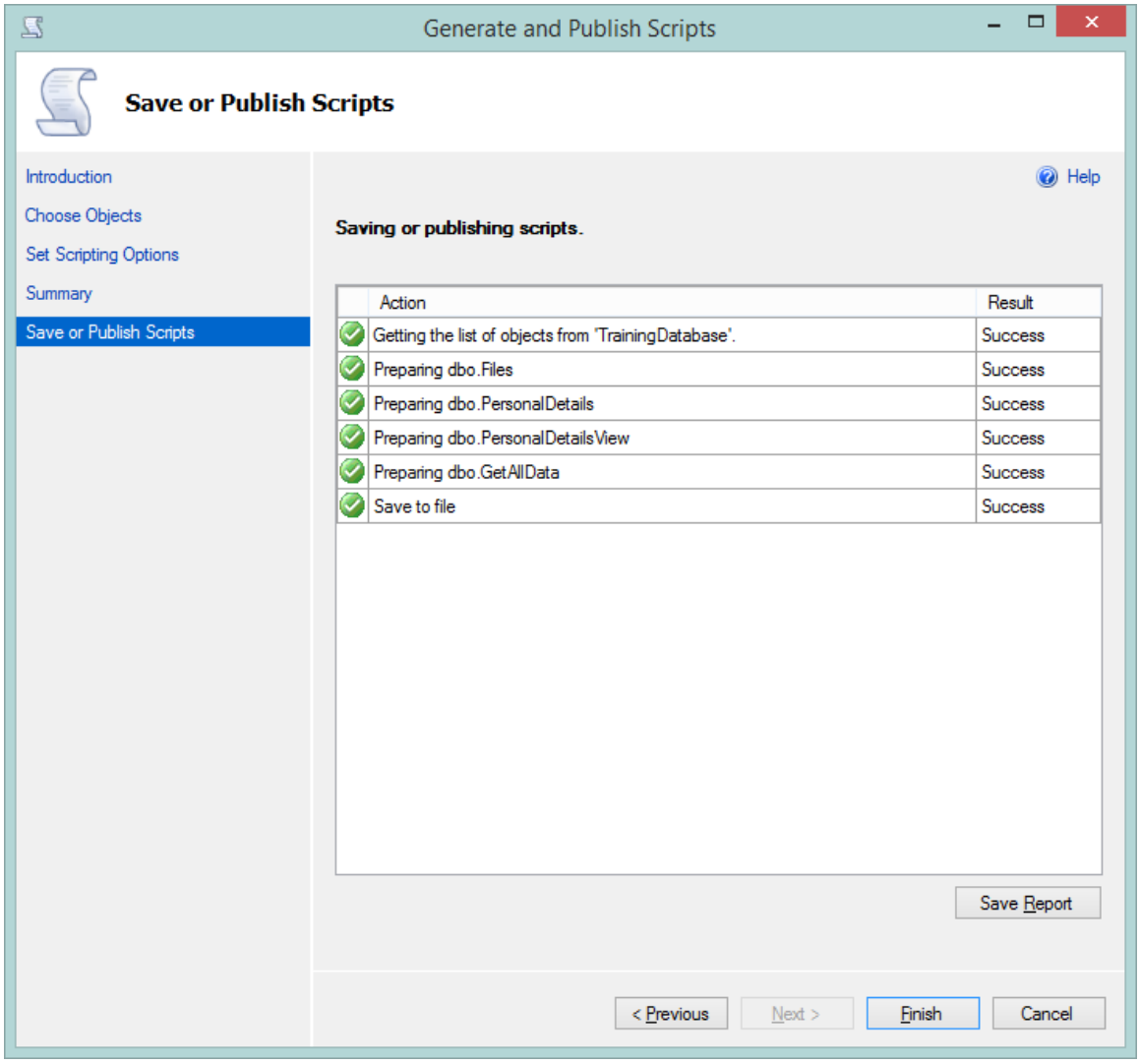
To save the generated in the Clipboard select “Save to Clipboard” radio button and to save the generated script in the new query window select “Save to new query window” radio button.


Generate and Publish Scripts
Set Scripting Options
Introduction
Choose Objects
Set Scripting Options
Summary
Save or Publish Scripts
Help
Specify how scripts should be saved or published.
Output Type
Save scripts to a specific location
Publish to Web service
Save to file
Advanced
Files to generate:
Single file
Single file per object
File name:
E:\SNITFundaServices\TUTORIALS\SQL Serve
Overwrite existing file
Save as:
Unicode text
ANSI text
Save to Clipboard
Save to new query window
< Previous
Next >
Finish
Cancel

Press Next >



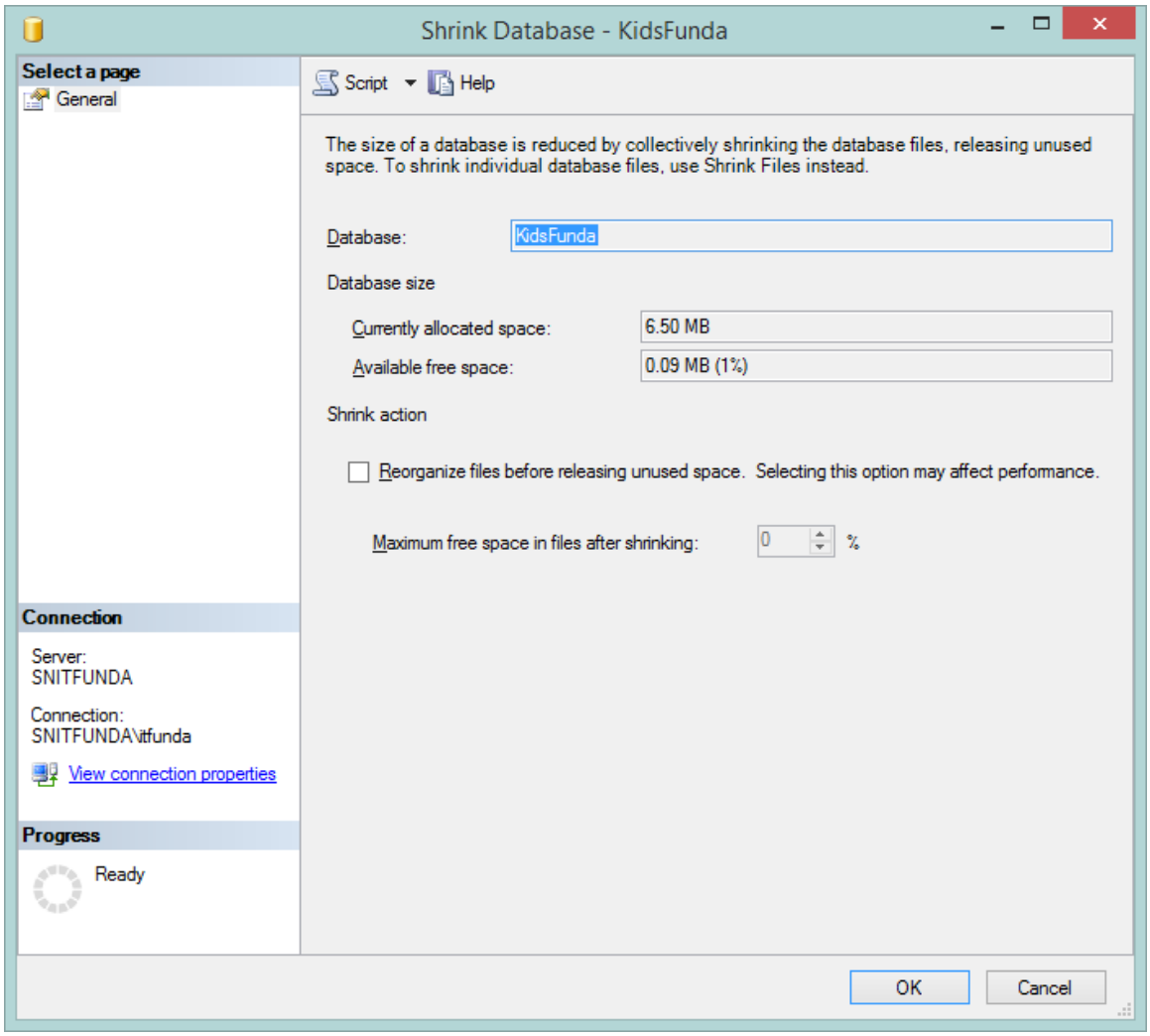
That will show the status of the script generated.



Note that the Generate Scripts…. do not generate the script to insert the existing data from the source database. It only creates script to create the selected objects such as Database tables, stored procedures etc.

13. How to shrink the database in SQL Server?

To shrink the database to release unused space by the database, we can right click the database to Shrink and go to Tasks > Shrink > Database. This opens the Shrink Database dialog box.



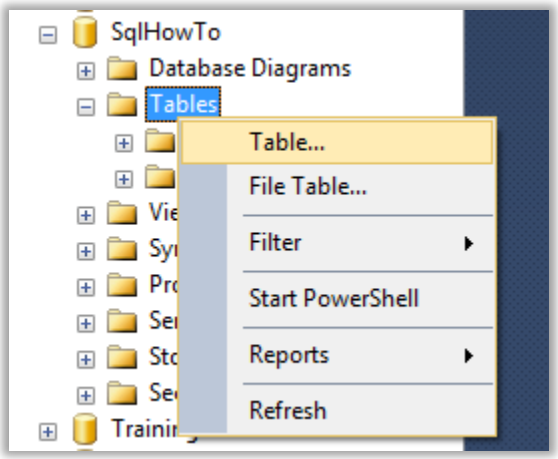
By default, it shows the currently allocated space and available free space that can be released. Now click on OK and that will release the available free space from the database file.

Table

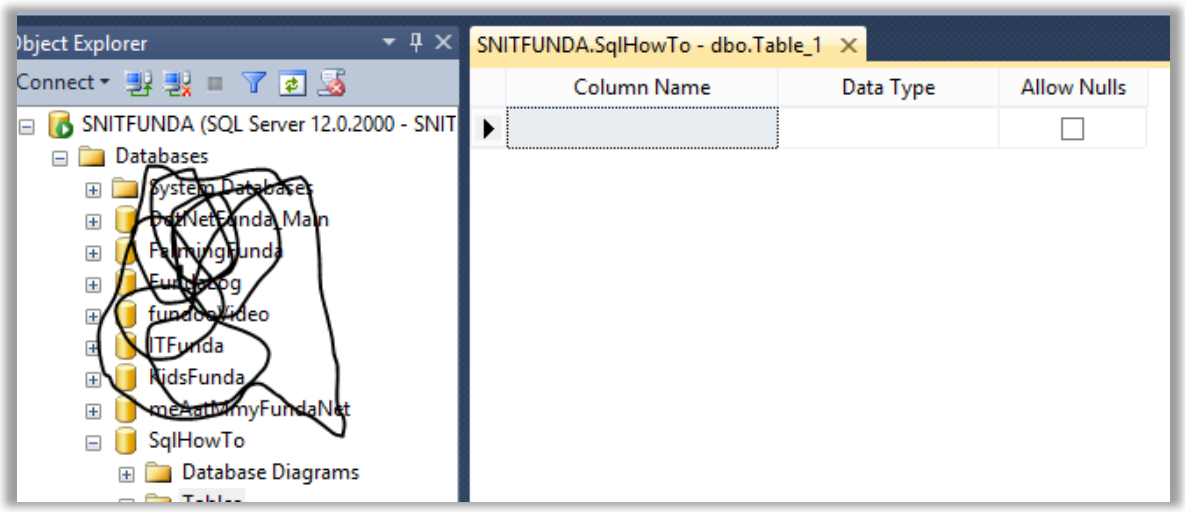
Table is an object in the database that actually holds the data into the database. Data is stored in the form of rows and columns into the database.

14. How to create a table in the database in SQL Server?

To create a table in the database, right click the Tables and select Table… option.

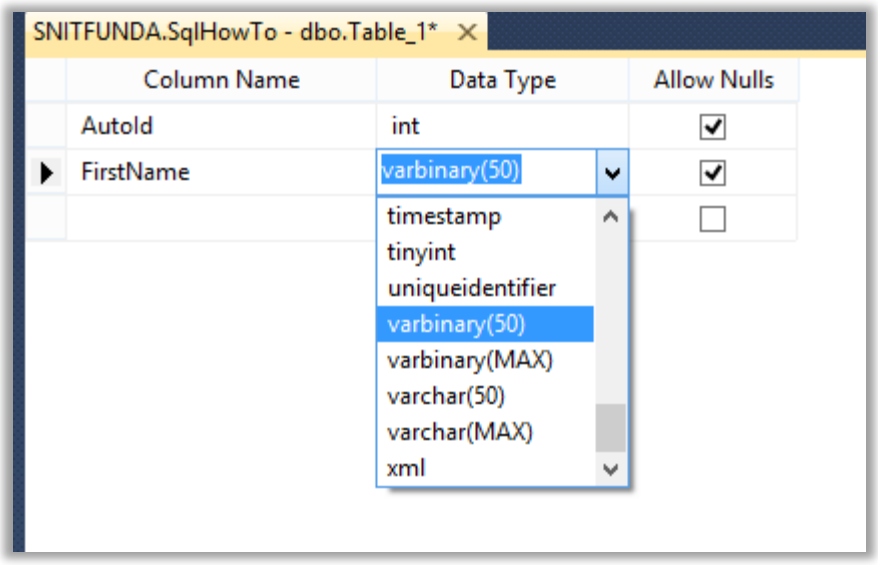


This opens up a New table panel in the right side window.

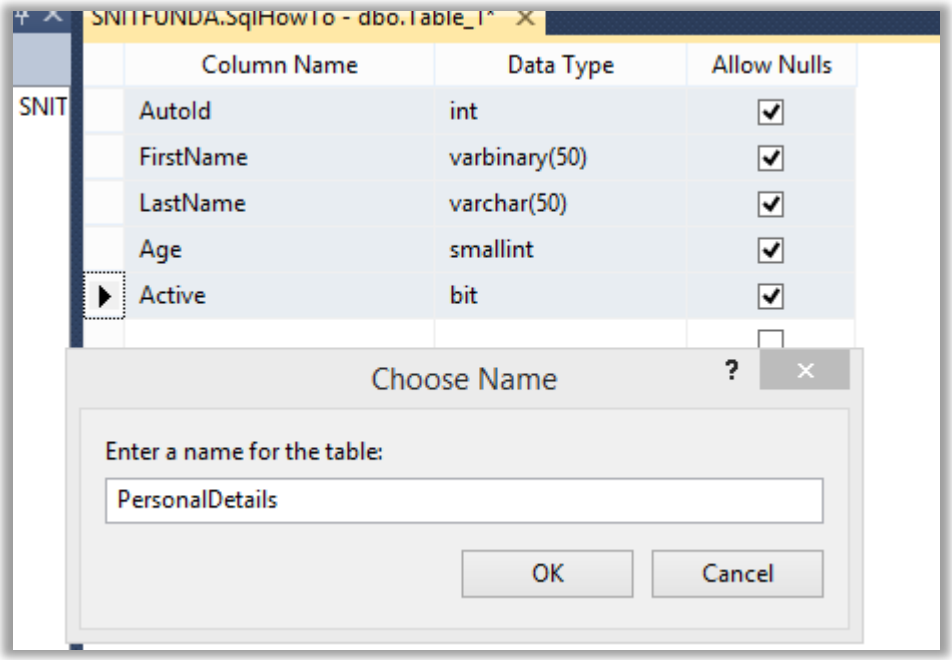


Here, we can mention the column name, data type for the column and whether null is allowed or not.

After writing the Column name, select the data type to store into the 2nd Column from the dropdown.

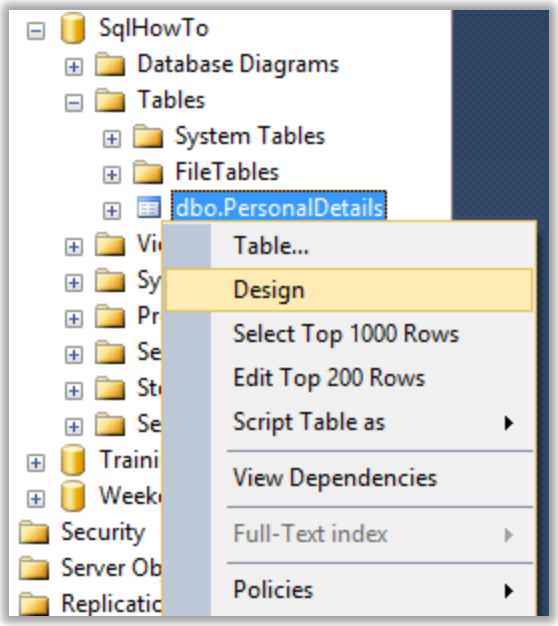


After writing the column name, click on the Save icon from top-left or File menu that gives a dialog box to write the name of this Table to save.

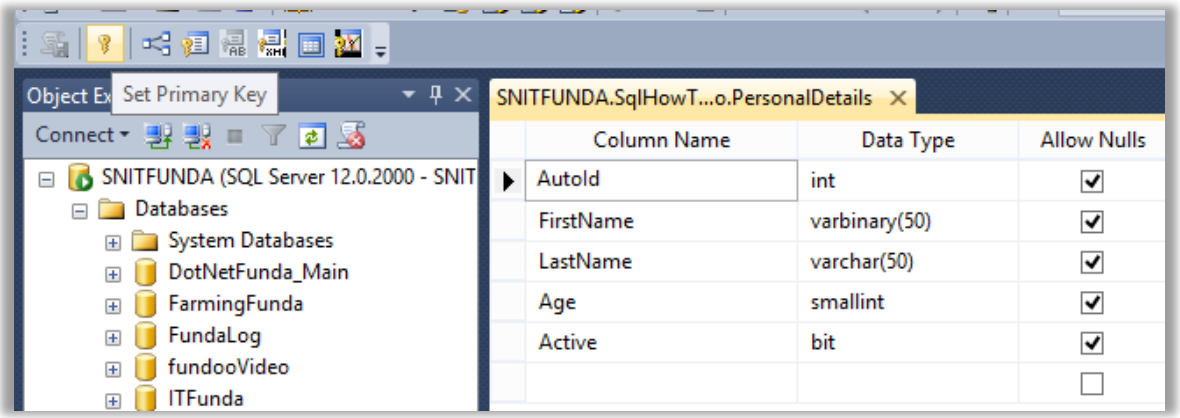


15. How to specify a primary key in the database table in SQL Server?

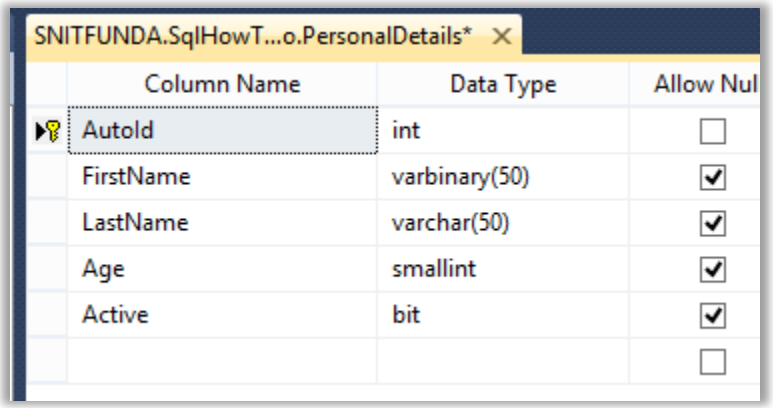
Open the database table in the Design view, select Design option by right clicking the table name from the database as shown below.



This opens up the table in design view. Now keep the mouse cursor on the column name which needs to be specified as primary key and click on Set Primary Key (key symbol) icon from the top-left as shown below.



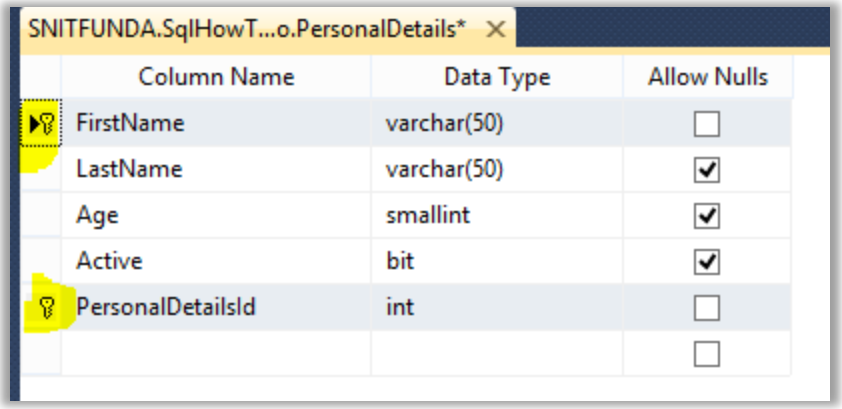
Clicking the icon marks the column as Primary key (key symbol beside it).



Do not forgot to click on the Save icon to save the table otherwise the changes will not affect in the database table.

16. How to set more than one columns of the table as Primary key in SQL Server database?

Yes, it is possible to create primary key on combination of columns on the table, to do that select columns to create primary key on by selecting the first column in the design mode by clicking on the left most box and the hold the ctrl key (on keyboard) and clicking on the 2nd column left most box.



This creates primary key on both columns and notifying it by keeping the key symbol for both columns.

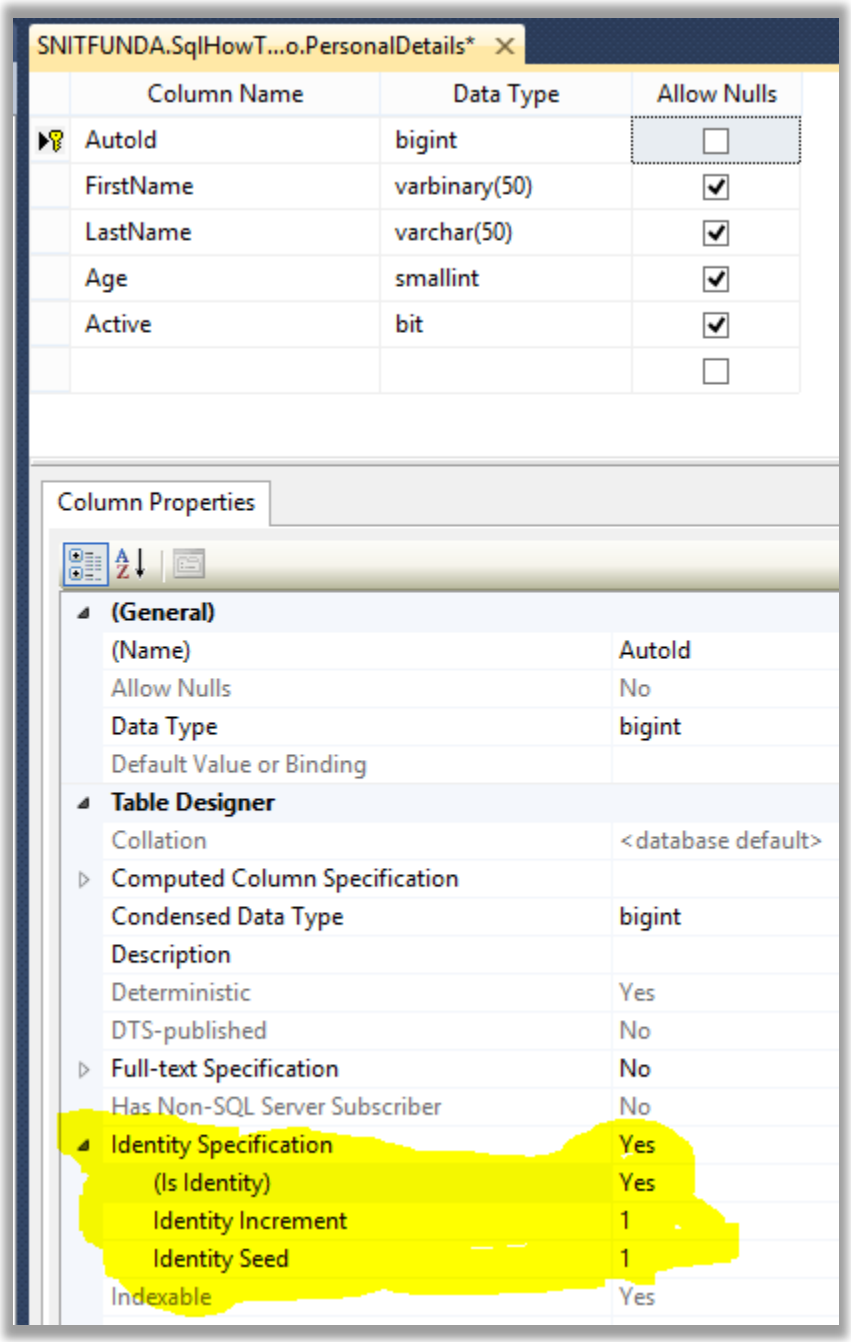
17. How to specify auto-increment column in the database while creating a new table in SQL Server?

Auto-increment column is very useful if we want a unique key for each record in the database table. This is generally treated as primary key and used to select, update, and delete records into the database table.

To specify an auto-increment column in the database table, the column name must be of Integer type (Int, BigInt etc.).

Select the column by clicking on the column and then see the Column Properties panel below it.

Go to Identity Specifications and explore it. Make (Is Identity) row as Yes and by default Identity Increment row and Identity Seed row become 1. In case we want to automatically increase the value of this column by 2 (like 1, 3, 5, 7 etc.) then change the value of Identity Seed to 2.



After setting this, save it by clicking the Save icon from top-left.

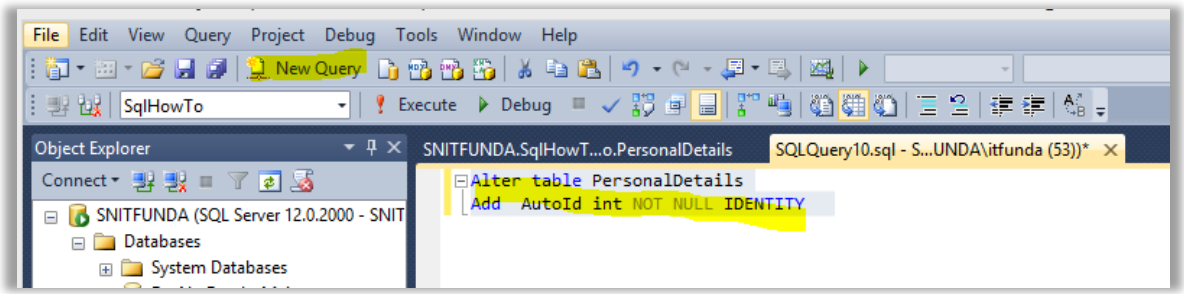
Important:

Please note that in SQL Server Management Studio Express, the auto-increment column can only be created at the time of creating a fresh table. Alter table, doesn't provide option to add a new column with auto-increment facility.

18. How to add a new column in the existing database table as Auto Increment column in the SQL Server?

Adding a new column as auto increment to the existing database table is not possible through the SQL Server Management studio (SSMS), so we do it by SQL Script. Open a new query window by clicking on the New Query icon from top left and run following command

```
Alter table PersonalDetails
Add  AutoId int NOT NULL IDENTITY
```



This adds AutoId column to the PersonalDetails table with auto – increment type so that entering new record into this table automatically increase the value of the AutoId column.

19. How to specify increment and initial value to the new auto increment column from the code in SQL Server?

To specify see value and increment value to the auto increment column in the existing database table, we pass parameter to the IDENTITY.

The 1st parameter is the initial value and the 2nd parameter is the step value.

```
Alter table PersonalDetails
Add  AutoId int NOT NULL IDENTITY (1, 5)
```

In this case, the 1st row AutoId value will be 1 and 2nd row AutoId value will be 6 and so on as the increment parameter is 5.

20. How to set a column as primary key to the existing database table in SQL Server?

To set existing column as primary key, run following sql script in the Query window.

```
Alter table PersonalDetails
ADD Primary Key (AutoId)
```

Here AutoId column will be set as primary key to the PersonalDetails table.

21. How to create a new column in the existing database table as Auto increment and primary key?

To create a new column in the existing database table as primary key and auto increment, execute following script in the query window against the database.

```
Alter table PersonalDetails
Add  AutoId int NOT NULL IDENTITY (1, 1)  Primary key
```

This adds a new column as AutoId with auto increment value and set this column as Primary key of the table.

22. How to and which data type should be used for what type of data?

Following are the data type and its range

To store any kind of integer data, we can use following data types. Notice the Range column that is the min and max limit of data we can store.

Data type	Range	Storage
bigint	-2^63 (-9,223,372,036,854,775,808) to 2^63-1 (9,223,372,036,854,775,807)	8 Bytes
int	-2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)	4 Bytes
smallint	-2^15 (-32,768) to 2^15-1 (32,767)	2 Bytes
tinyint	0 to 255	1 Byte

To store any decimal or numeric data, we can use following data types.

Dat type	Precision	Storage bytes
Decimal Numeric	**1 - 9**	**5**
	10-19	**9**
	20-28	**13**
	29-38	**17**

Numeric maps to the Decimal data type of the .NET and it is functionally equivalent to the Decimal data type.

To store small number with floating decimal point, float data type can be used.

Data type	Range	Storage
float	**- 1.79E+308 to -2.23E-308, 0 and 2.23E-308 to 1.79E+308**	**Depends on the value of n**
real	**- 3.40E + 38 to -1.18E - 38, 0 and 1.18E - 38 to 3.40E + 38**	**4 Bytes**

Float – maps to the Single data type of .NET

Real – maps to the Double data type of the .NET

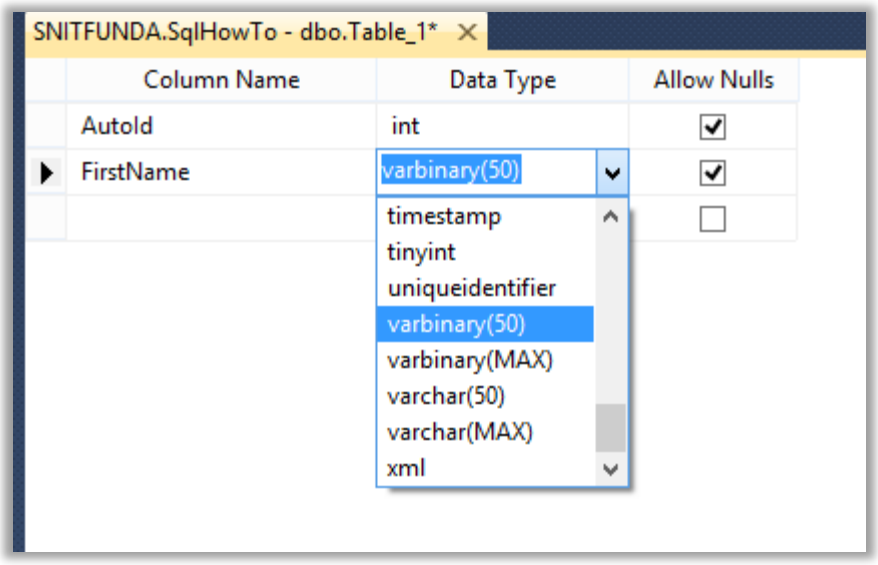
To store monetary data, money or smallmoney data type can be used.

Data type	Range	Storage
money	**-922,337,203,685,477.5808 to 922,337,203,685,477.5807**	**8 bytes**
smallmoney	**- 214,748.3648 to 214,748.3647**	**4 bytes**

Data type	
Char(n)	**Used to store fixed length characters**

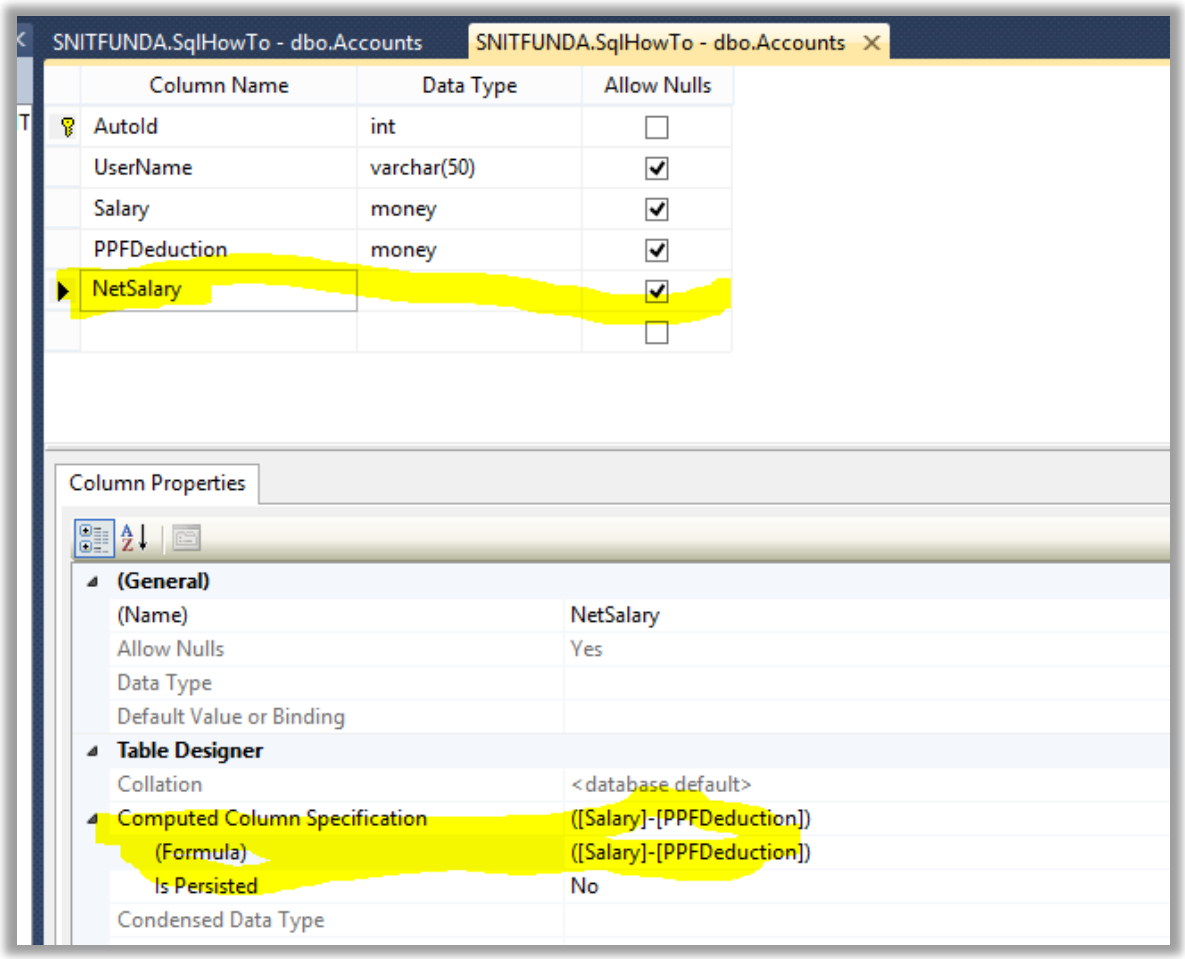
Varchar(n)	**Used to store variable length characters**
Nchar(n)	**Used to store fixed length Unicode UCS-2 characters set**
Nvarchar(n)	**Used to store variable length Unicode UCS-2 characters set**
Time	**Used to store time (hh:mm:ss]**
Date	**Used to store date (YYYY-MM-DD)**
Datetime	**Used to store date along with time with fractional seconds**
Smalldatetime	**Used to store date along with time**
Datetime2	**Stores the date along with time with larger fractional precious seconds**
Bit	**To store true (1) or false (0)**
Binary	**To store binary data of fixed length**
Varbinary(n)	**To store binary data of variable length**
Uniqueidentifier	**A 36 character length unique GUID data can be stored**
Text	**To store variable length large data**
Ntext	**To store variable length large Unicode data**
Image	**To store variable length binary data (to store image or file into database table)**

To select a data type for a column in the database table, click on the Data type dropdown and select it.



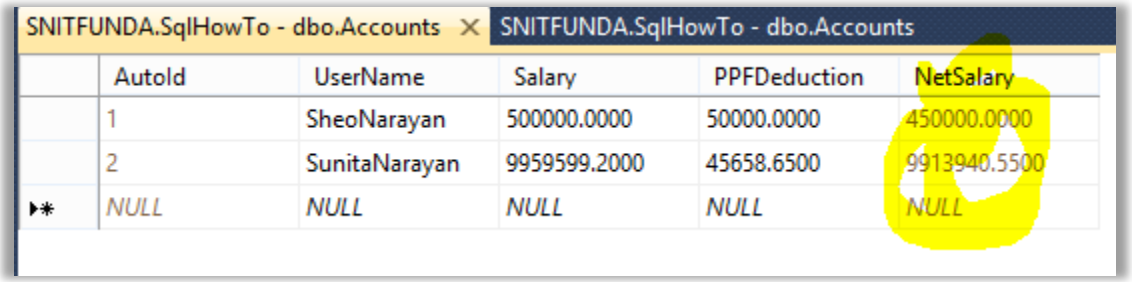
23. How to create computed column (eg. Salary column = Ctc column – Pf column = Net column) in SQL Server?

To create a computed column in in SQL Server, open the table in design view and add a column and go to Column Properties. Explore Computer Column Specification and write the Formula.



In this case, we have written computer column as NetSalary whose value is Salary column value – PPFDeduction column value.

While inserting record for this database table, we no need to write data for NetSalary, writing value in Salary and PPFDeduction automatically calculates the NetSalary value.



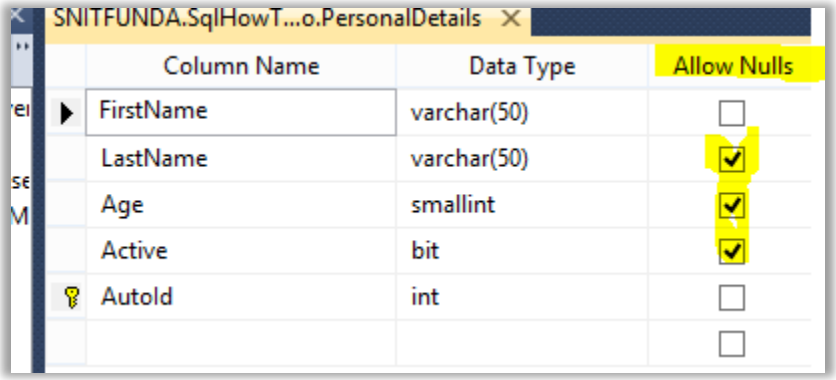
24. How & why to specify a database table column as nullable in SQL Server?

Let's first understand the difference between NULL and Empty

NULL – is the absence of value that is completely blank

Empty – is a string that is empty

To make a column nallable (that can hold null value), we to check "Allow Nulls" checkbox at the time of creating/designing the database table.

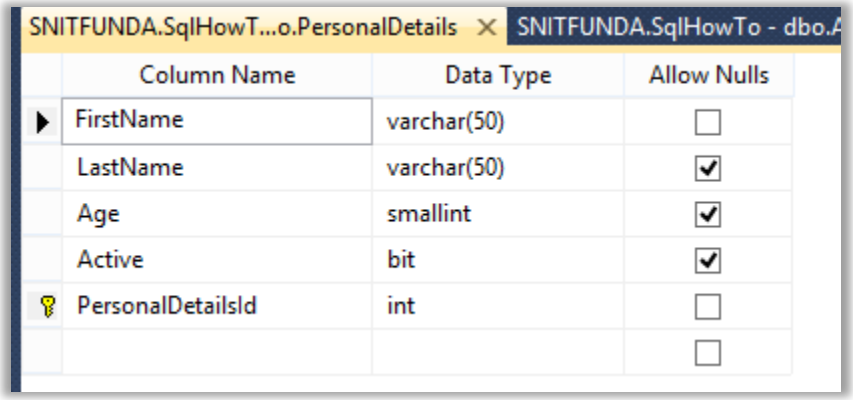


This can be set to any type of data type Int, varchar, datetime etc.

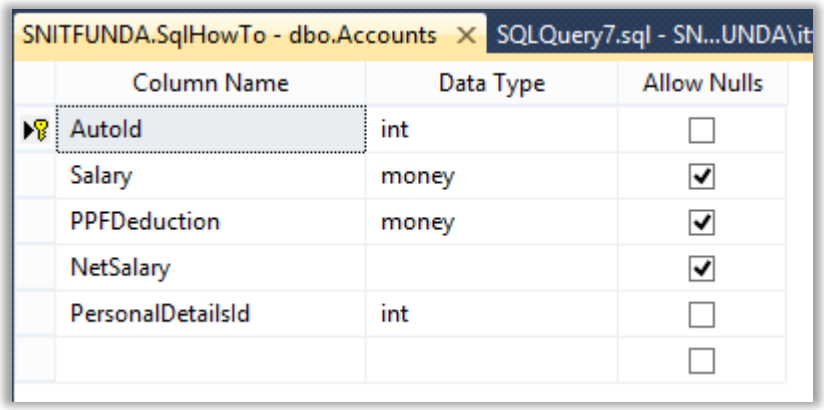
25. How to add primary key and foreign key relationship in a Sql Server database table?

In order to demonstrate this, we have created two tables as shown below

PersonalDetails table



Accounts table



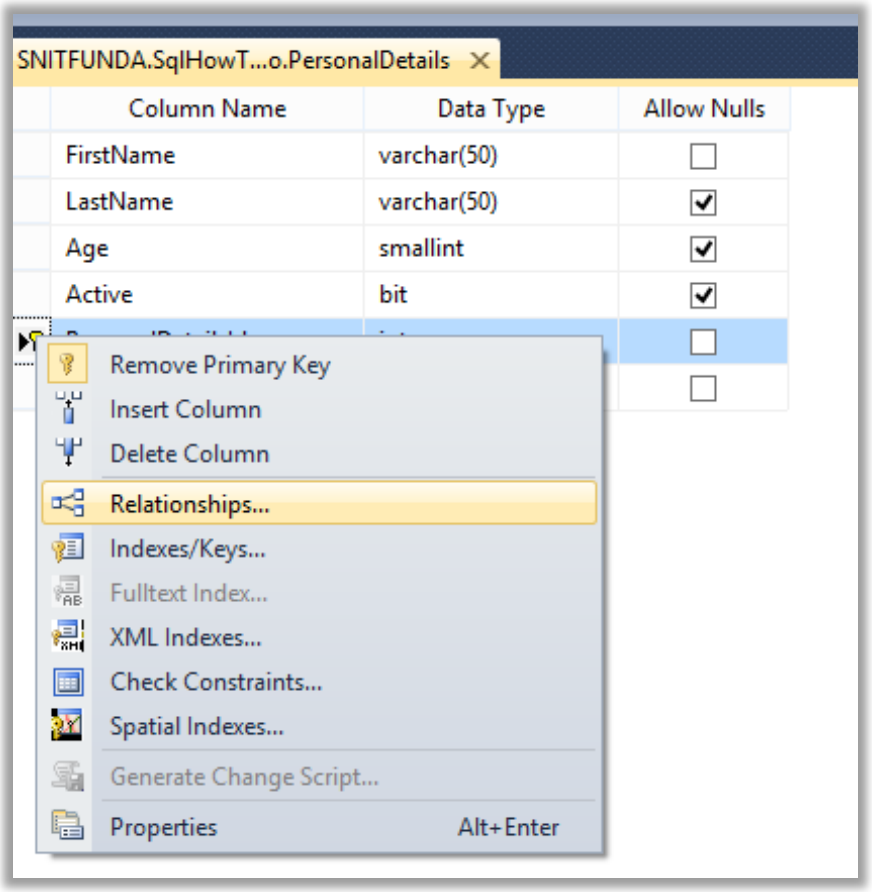
Here the primary personal data is being stored into PersonalDetails table and finance related details of the person is being stored into Accounts table.

Accounts table is linked with the PersonalDetails table with PersonalDetailsId column in the Accounts table. That means the Accounts.PersonalDetailsId column will hold the value of PersonalDetails.PersonalDetailsId for each record in the PersonalDetails table.

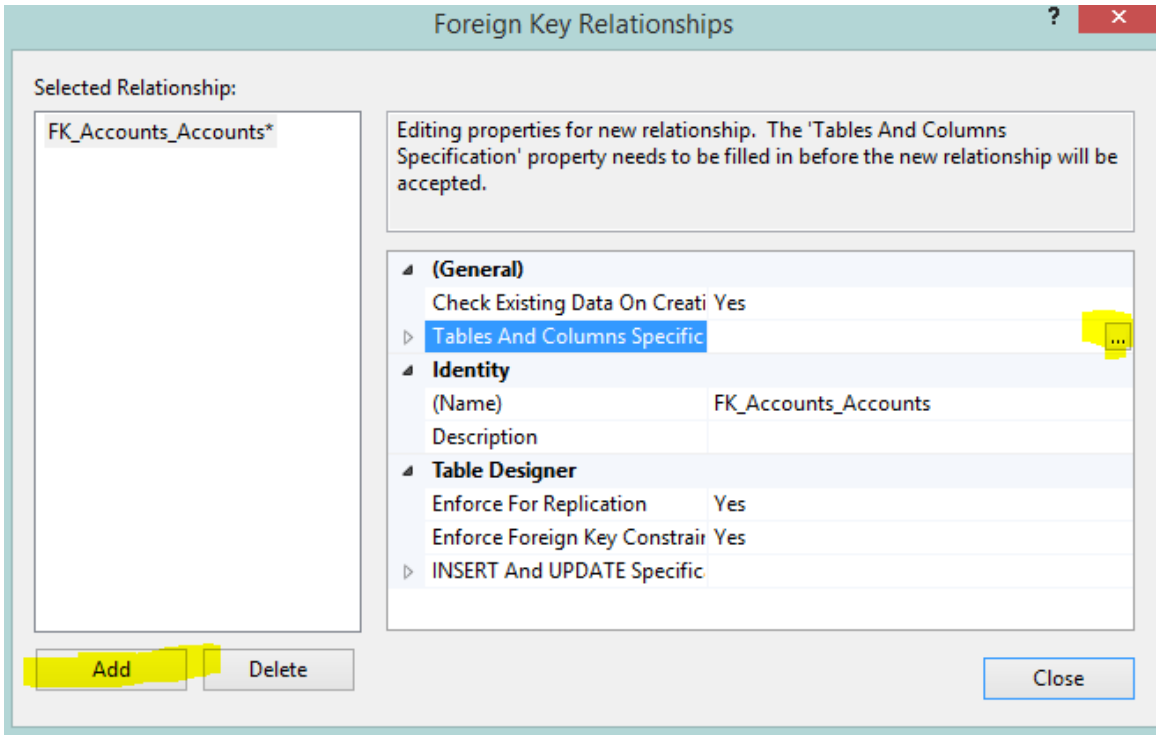
Here Primary key table is PersonalDetails and Foreign key table is Accounts as it is holding the primary key value into its PersonalDetailsId column.

To make the relationship, the PersonalDetailsId column must be primary key (learn how to set the column as primary key in the above How to points).

To create the primary key and foreign key relationship, right click the foreign key table columns (Accounts table) and select Relationships…

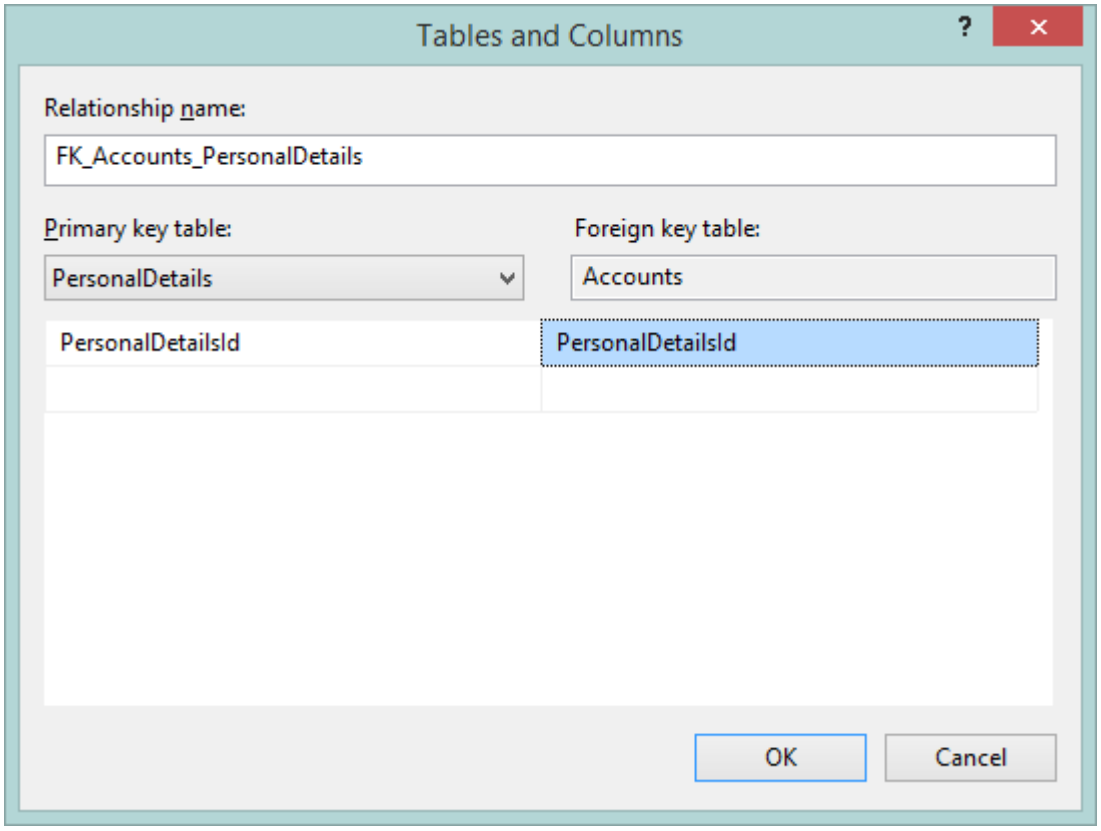


In the Foreign Key Relationships dialog box, click Add button.

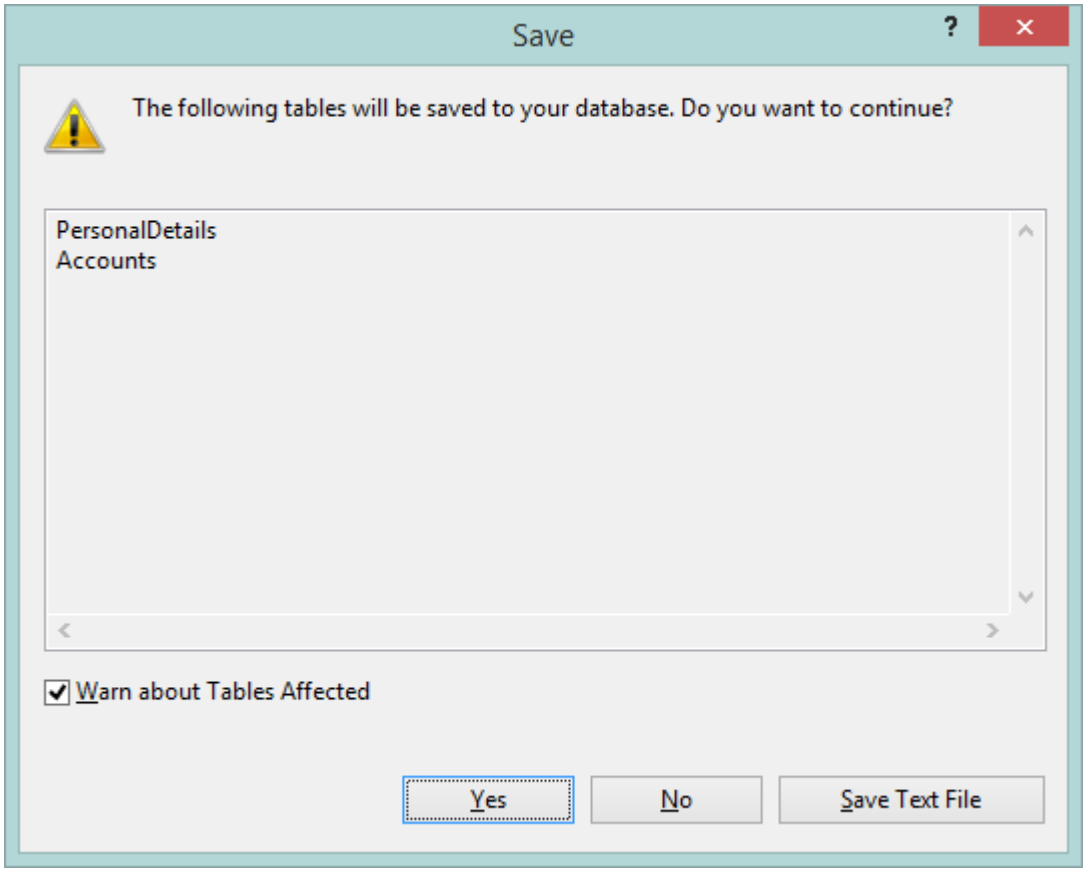


That will by default add a relationship in the left panel. Now click on the … box highlighted above.

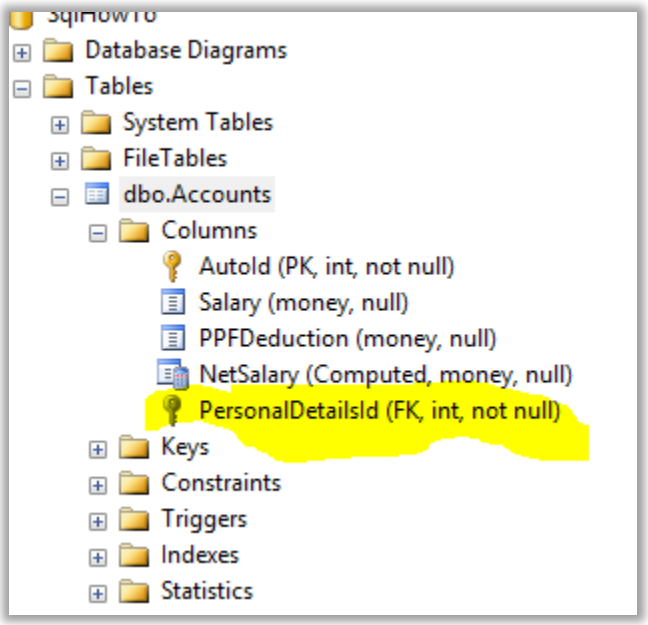
Now click on the Primary key table dropdown and select PersonaDetails and then click on the rows below to select PersonalDetailsId from the columns link and then in the Foreign key table below the table name and click the rows and select PersonalDetialsId, now click OK as shown below.



Then again click on Close button of the Foreign Key Relationships dialog box. At last click on the Save icon that will show below dialog box on which click yes to save the database table relationship just now created.



To confirm whether the foreign key has been created, explore the columns of the Accounts table in the Object Explorer and see the gray key icon against PersonalDetialsId column.



26. How to create indexes in the Sql Server database table?

Creating Indexes in the database table columns helps us to search the data on that particular easily and quickly. The overall performance of the searching becomes much better.

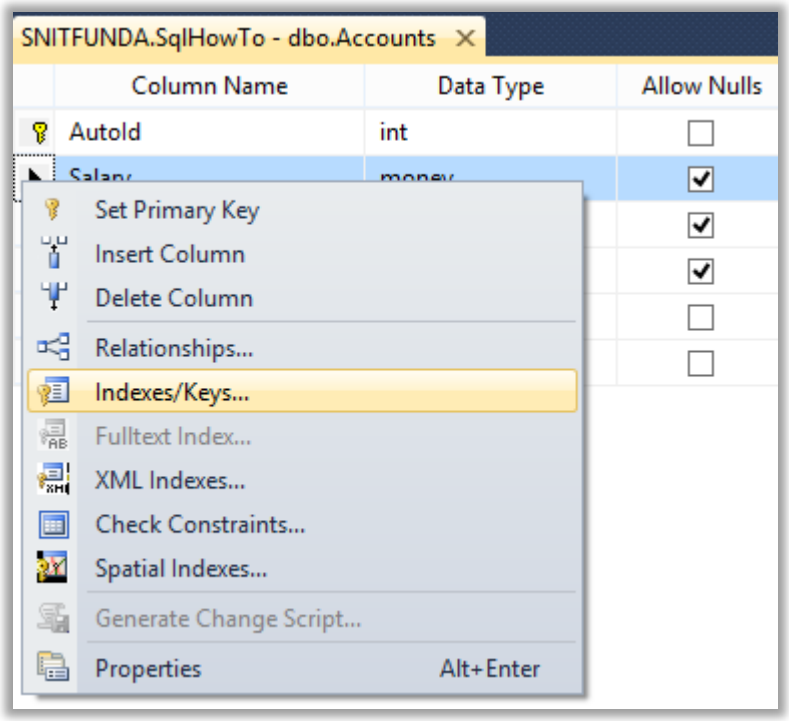
When we create a primary key in the table, by default it creates clustered index on the primary key column.

- Clustered Index: Clustered index sorts and stores the data rows in the table based on their key values when we insert the record. In general, there is only one clustered index in a single table.
- Nonclustered Index: A nonclustered index contains index key values and each key value entry has a pointer to the data row that contains the key value in the database table.

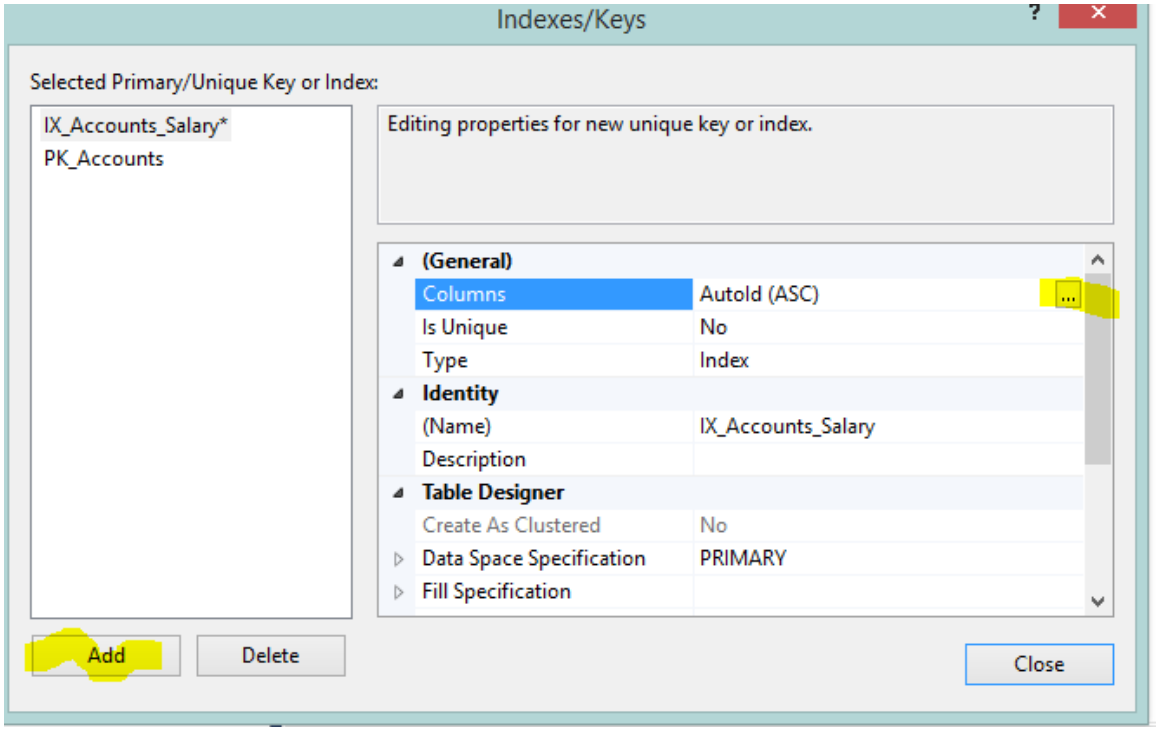
  It is generally created on columns other than primary key on which search needs to be done to improve the search performance.

So to create a clustered index, we do not need to do anything. Just creating a primary key in the table does the job.

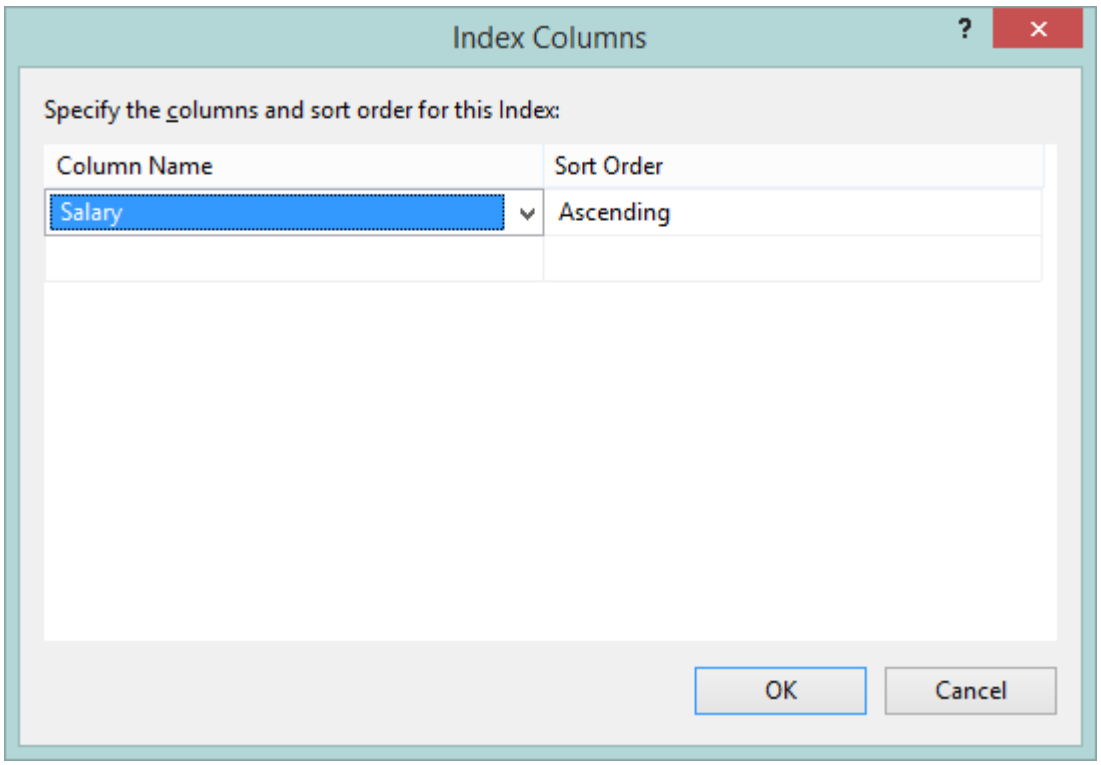
To create nonclustered Index, right click the column in the design view and select Indexes/Keys...



This shows Indexes/Keys dialog box as displayed below



Now click on Add button and change the (Name) value to some meaningful name to make this name unique. Now click on … button that shows Index Columns dialog box.



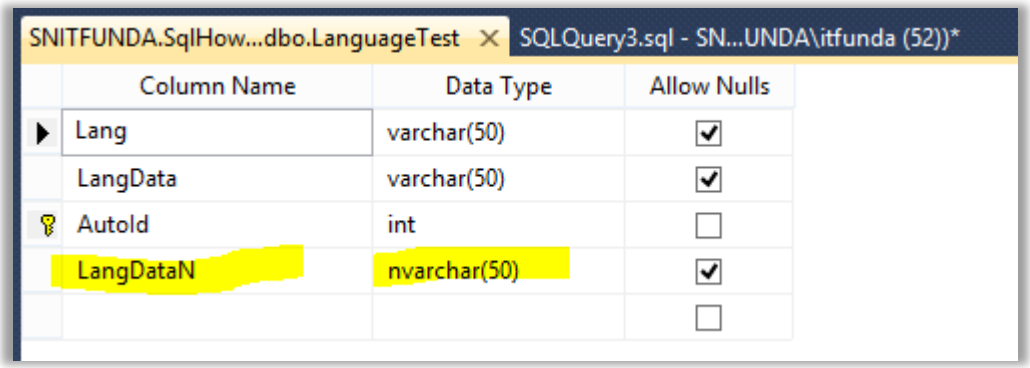
Select Column name on which we need to create nonclustered index and sort order. Click on OK and then Close on the Indexes/Keys dialog box. Now click on Save icon to save the index created.

There can be more than one nonclustered index in a single table. However it is suggested not to create unnecessary indexes as these indexes also has an overhead and instead of increasing performance it may decrease the search performance.

27. How to store data of other language in the Sql Server database table?

To store data of other language than English in SQL Server, we use Unicode compatible data types in the column such as nvarchar, nchar, ntext otherwise proper data is not stored in the database. Also while inserting data into those columns we prefix the data with "N" character.

In order to demonstrate this, we have created a sample demo database table named LanguageTest



Where the LangData and LangDataN columns will be used to store data.

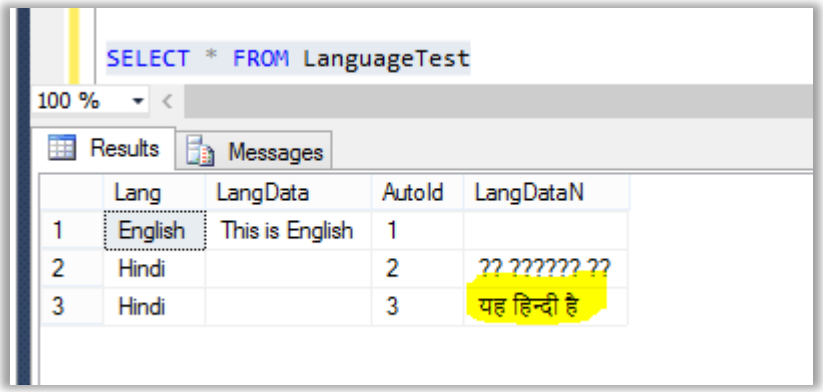
LangData is of varchar type and LangDataN is of nvarchar type (Unicode compatible)

We have entered three records into this table with the help of below INSERT statements. Write below query in the Query window and click on "! Execute" icon at the top-left.

```
INSERT INTO LanguageTest (Lang, LangData, LangDataN) VALUES ('English', 'This is English', '')
INSERT INTO LanguageTest (Lang, LangData, LangDataN)  VALUES ('Hindi', '', 'यह हिन्दी है')
INSERT INTO LanguageTest (Lang, LangData, LangDataN)  VALUES ('Hindi', '', N'यह हिन्दी है')
```

Note that in the above insert statements, the second statement is inserting data into LangDataN column that is of nvarchar type and third statement is also inserting data into LangDataN column but by prefixing "N".

Now, the result looks like below.



When we inserted data into LangDataN column without prefixing "N", the data appears as"????....." but when we insert data by prefixing "N" in the LangDataN, it insert correct data in correct format.

Remember that if we are inserting data into LangDataN column using .NET program, we do not need to prefix the data with "N". Like in below example.

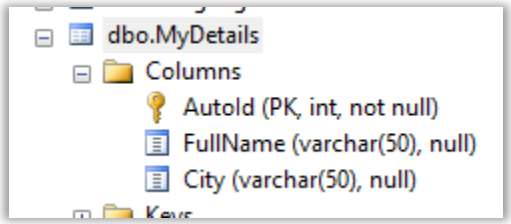
```
string connStr = "Data source=SNITFUNDA;Initial
Catalog=SqlHowTo;Integrated Security=true;";
        using (SqlConnection conn = new SqlConnection(connStr))
        {
            string sql = "INSERT INTO LanguageTest (Lang, LangData,
LangDataN)  VALUES (@lang, @langData, @langDataN)";

            using (SqlCommand cmd = new SqlCommand(sql, conn))
            {
                cmd.Parameters.AddWithValue("@lang", "Hindi");
                cmd.Parameters.AddWithValue("@langData", "");
                // string hindiData = "Nयह हिन्दी है"; - NO prefexing is
required
                string hindiData = "यह हिन्दी है";
                cmd.Parameters.AddWithValue("@langDataN", hindiData);
                conn.Open();
                cmd.ExecuteNonQuery();
            }
        }
```

Table – Create, Read, Update, Delete (CRUD)

28. How to insert a new record into SQL Server database?

To create a record into SQL Server database table, we use insert statement.

Assuming that we have MyDetails table whose structure is below.



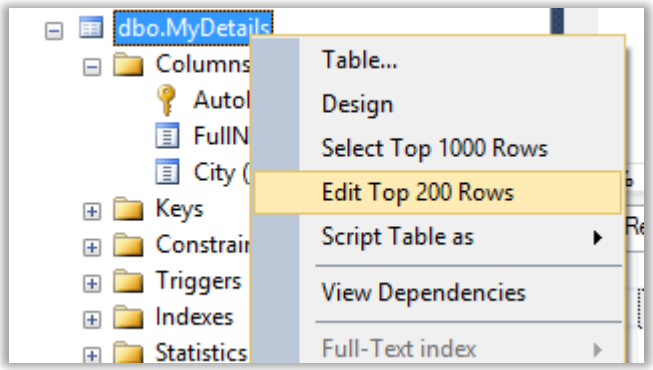
Using SQL Script

```
INSERT INTO MyDetails
(FullName, City)
VALUES
('John Singh', 'New Delhi')
```

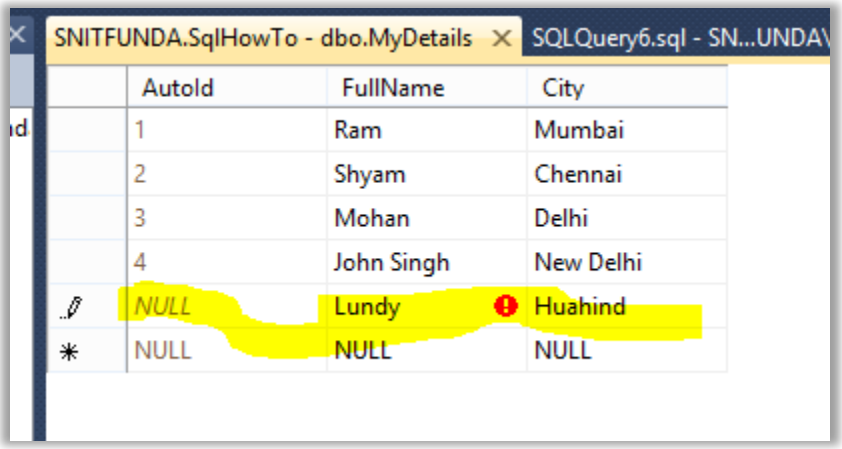
Execute the above query in the Query window by writing it into the Query Window and clicking on "! Execute" icon at the top-left. This will get one record inserted into the database. Here VALUES is the reserved keyword that is preceded by the column names to insert and appended with the value to insert into the respective columns.

Using SSMS (Sql Server Management Studio) menus

Right click the database table and select "Edit Top xxx Rows"



This brings the table in Edit mode, now write data for FullName and City column, do not try to write data into AutoId column as that is auto increment and is automatically filled.



29. How to read data from SQL Server database table?

To read data from SQL Server database, we use SELECT statement.

Using SQL Script

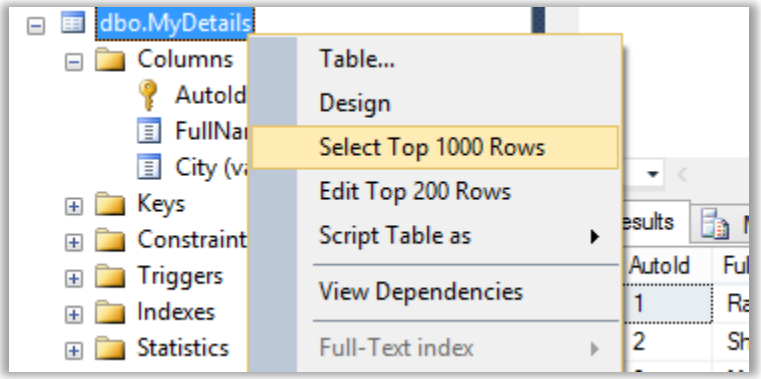
```
SELECT * FROM MyDetails
```

Using SQL Script – Column wise

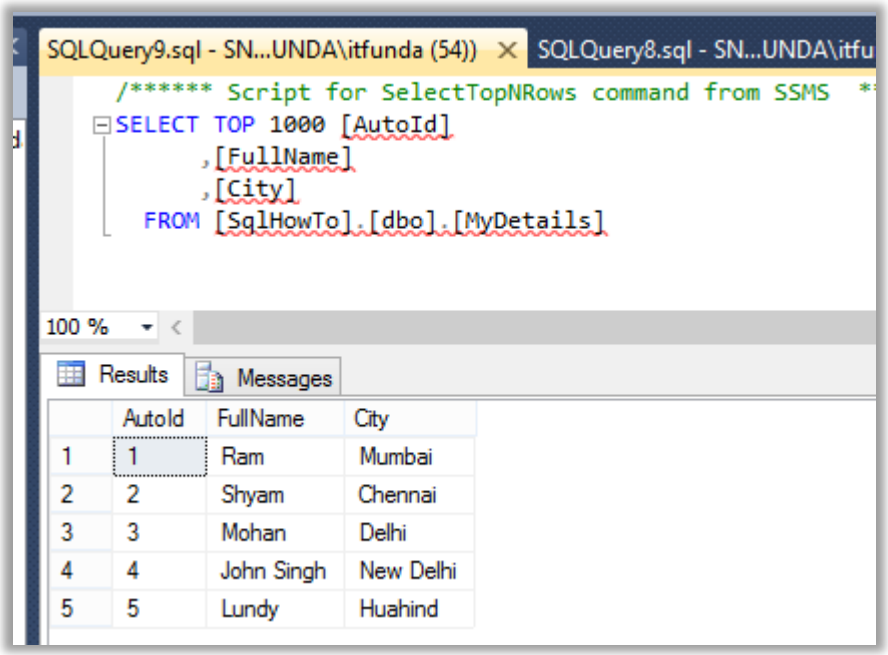
```
SELECT AutoId, FullName, City FROM MyDetails
```

In above scripts, MyDetails is the name of the database table, AutoId, FullName and City is the name of the columns into this database table.

Through SSMS menus



Right click the database table and choose "Select Top xxxx Rows that brings a query window and result in the right side panel of the SSMS.



It basically, writes the SQL statement and executes it that shows top 1000 records (max) from the database table.

30. How to insert multiple records into the Sql Server database table at once?

To store multiple records into SQL Server database table, separate batch of columns values by comma (,)

First approach

```
INSERT INTO MyDetails (FullName, City)
VALUES
('Ram', 'Mumbai') ,
('Shyam', 'Chennai') ,
('Mohan', 'Delhi')
```

In the above insert statement, we have separated the columns blocks values by",". Here are 3 set of columns values for MyDetails database table.

i. `('Ram', 'Mumbai')`
ii. `('Shyam', 'Chennai')`
iii. `('Mohan', 'Delhi')`

Second approach

Another approach of inserting multiple records into the database table, we use UNION ALL separated by SELECT and the column values.

```
INSERT INTO MyDetails (FullName, City)
SELECT 'Gita', 'Gita nagar'
UNION ALL
SELECT 'Roselin', 'Roselin First'
UNION ALL
SELECT 'Olga', 'Olga Mati'
```

In this case, three records will get inserted into MyDetails database table.

31. How to update records into the Sql Server database table?

To update records into SQL Server database, we can use Update sql statement as shown below.

First approach – using SQL Script

```
Update MyDetails SET
FullName = 'Ram Modified',
City = 'City Modified'
WHERE AutoId = 1
```

Here, we are modifying FullName and City column of the MyDetails database table record whose AutoId is 1.

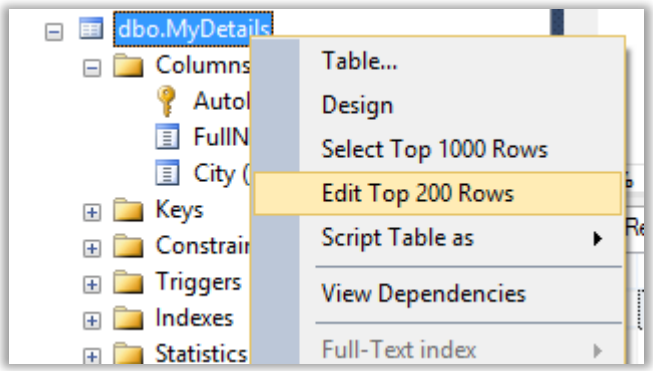
Important:

Notice the Where clause, this where clause is very important. In case we forgot to mentioned the Where clause in the Update statement, all records from MyDetails table will get updated.

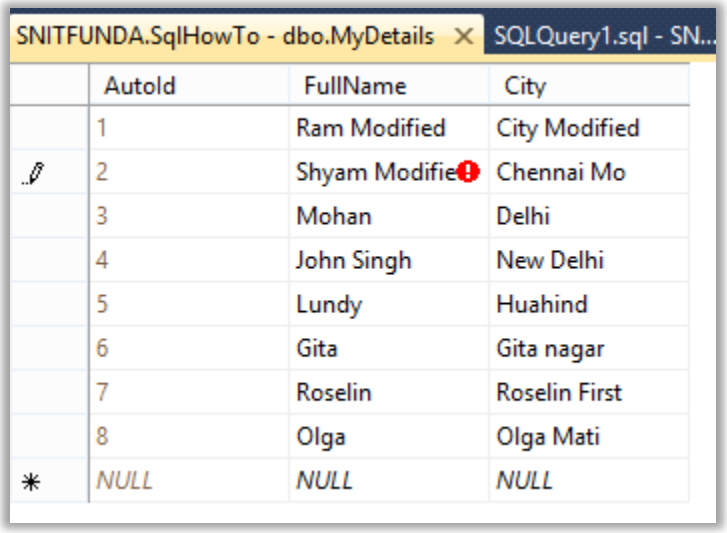
Always check for Where clause in the Update statement to avoid updating all records.

Second approach – using SSMS command

Choose "Edit Top xxx Rows" by right clicking the database table that opens up the table in update mode



Now modify the data by clicking on the editable cell and press Enter key to save the data.



	AutoId	FullName	City
	1	Ram Modified	City Modified
	2	Shyam Modifie	Chennai Mo
	3	Mohan	Delhi
	4	John Singh	New Delhi
	5	Lundy	Huahind
	6	Gita	Gita nagar
	7	Roselin	Roselin First
	8	Olga	Olga Mati
*	NULL	NULL	NULL

32. How to delete a record from the Sql Server database?

To delete a record from database table, we use follow below apporach

First approach – Sql script

```
DELETE MyDetails
  WHERE AutoId = 2
```

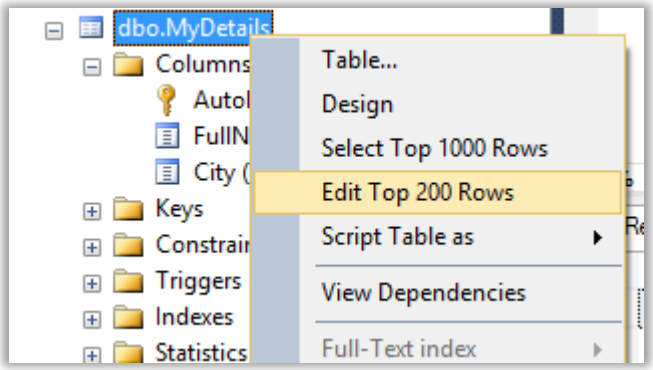
Execute the above Delete statement in Query window and it will delete the record from MyDetails table whose AutoId value is 2.

Important

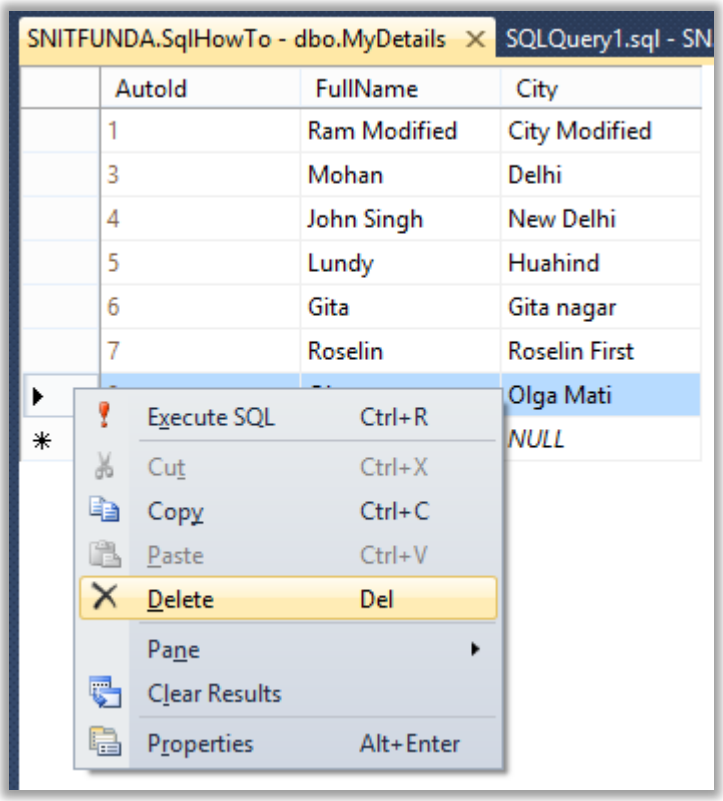
Notice the Where clause to filter the record to delete. If we forgot to mention the Where clause, it will delete all records from the database table.

Second approach

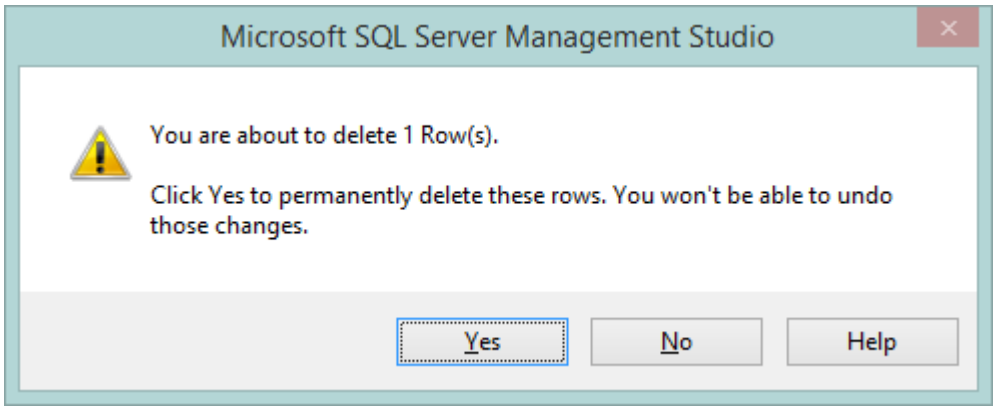
Deletion can also be performed by going to the Edit mode of the database table by right clicking table name and selecting “Edit Top xxx Rows”



Now, select the row to delete by clicking on the left first blank cell and then right click the row and choose Delete or press Del button from the keyboard.



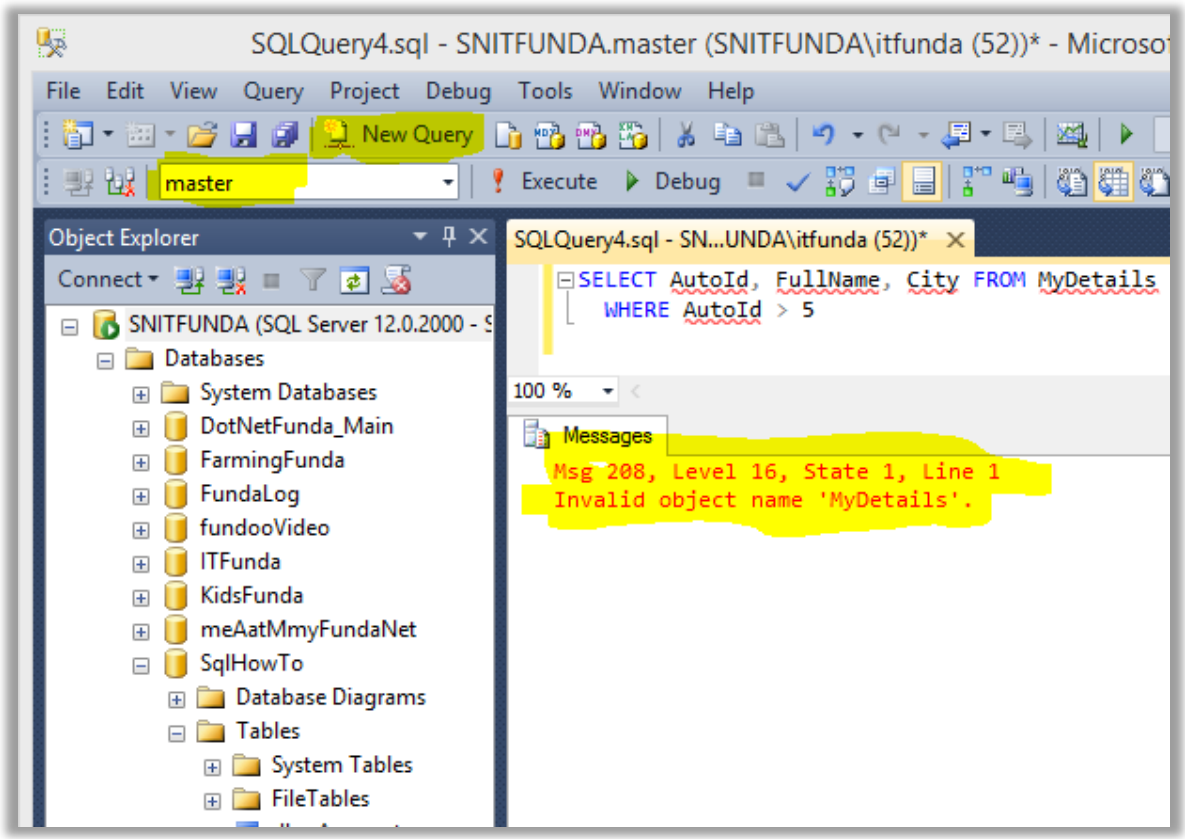
Selecting Delete opens up a confirmation dialog box



We need to click "Yes" in order to delete the record otherwise we can click on "No" to ignore it.

33. How to filter records of the database table in SQL Server?

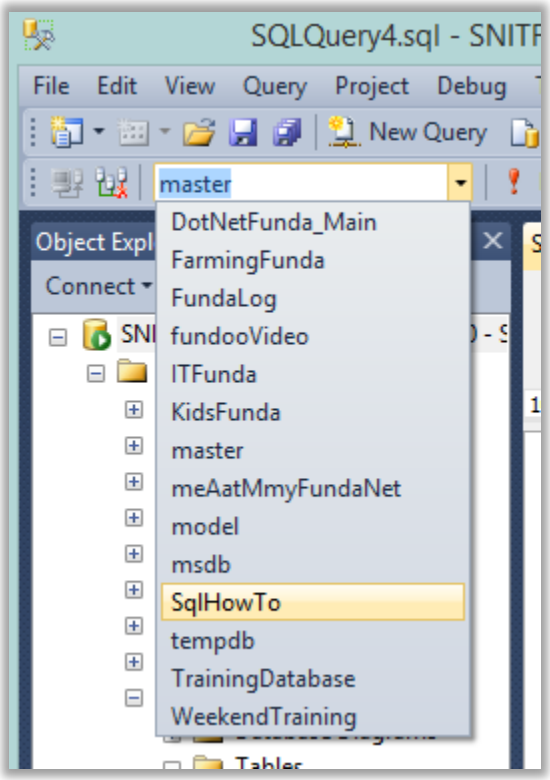
Open up a New Query window by clicking on the "New Query" icon from the top-left.



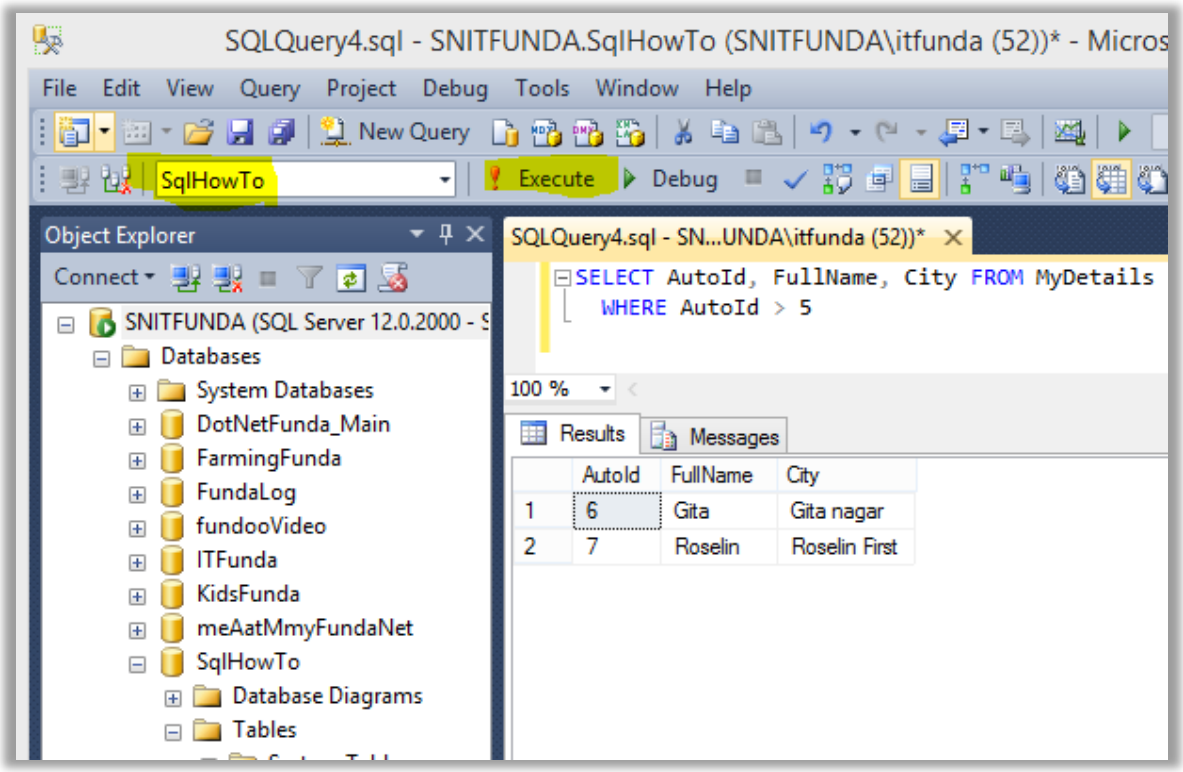
This opens up a new query window in the right panel. Writing Select statement as it is shown in above picture and clicking on "! Execute" icon throws error as shown above because the "MyDetails"

table doesn't exists into the database selected. Note that currently the database selected for this query window is "master".

To select our database, we need to click on the database dropdown as shown below and click on our database name.



Now click on "! Execute" and it will give desired result.



Filtering records based on AutoId

Notice the Sql statement in this case

```
SELECT AutoId, FullName, City FROM MyDetails
  WHERE AutoId > 5
```

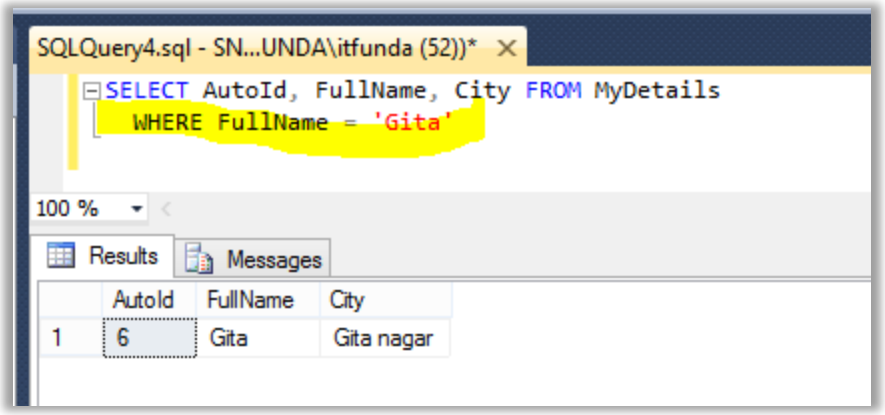
Here, we are instructing SQL Server that give us only those records from the MyDetails table whose AutoId value is more than 5.

Filtering records based on FullName

Similarly, we can also have the Where clause checking for the FullName column

```
SELECT AutoId, FullName, City FROM MyDetails
  WHERE FullName = 'Gita'
```

Here we are filtering records where FullName value is 'Gita'

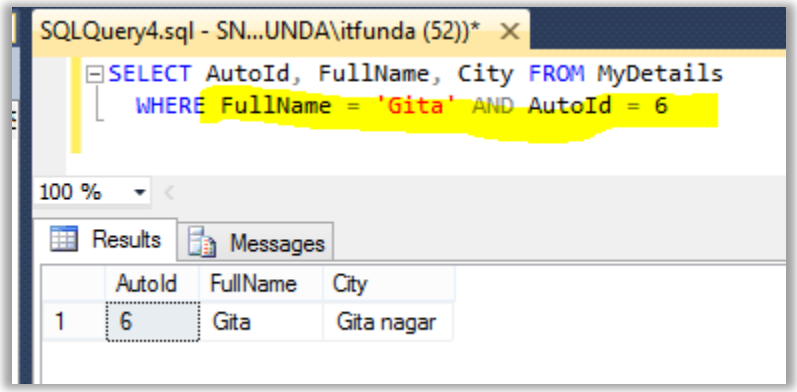


Filtering records based on more than one columns - AND

We can also filter records based on multiple columns with AND condition

```
SELECT AutoId, FullName, City FROM MyDetails
  WHERE FullName = 'Gita' AND AutoId = 6
```

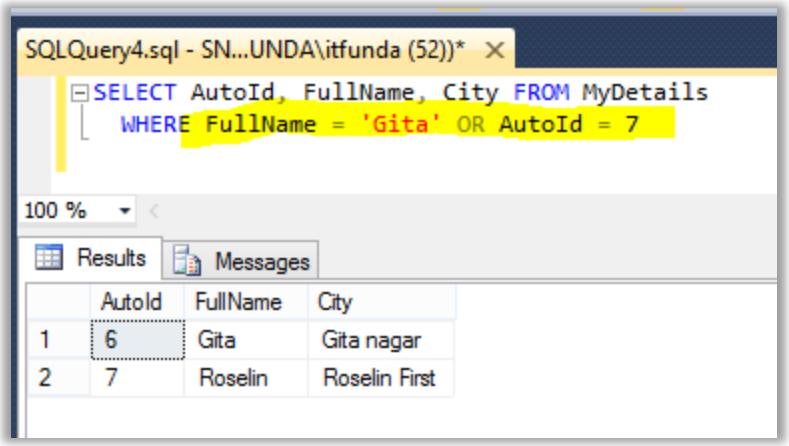
In this case, we will get only those records from MyDetails table where FullName is 'Gita' AND AutoId is '6'.



Filtering records based on more than one columns - OR

We can also filter records based on multiple columns with OR condition

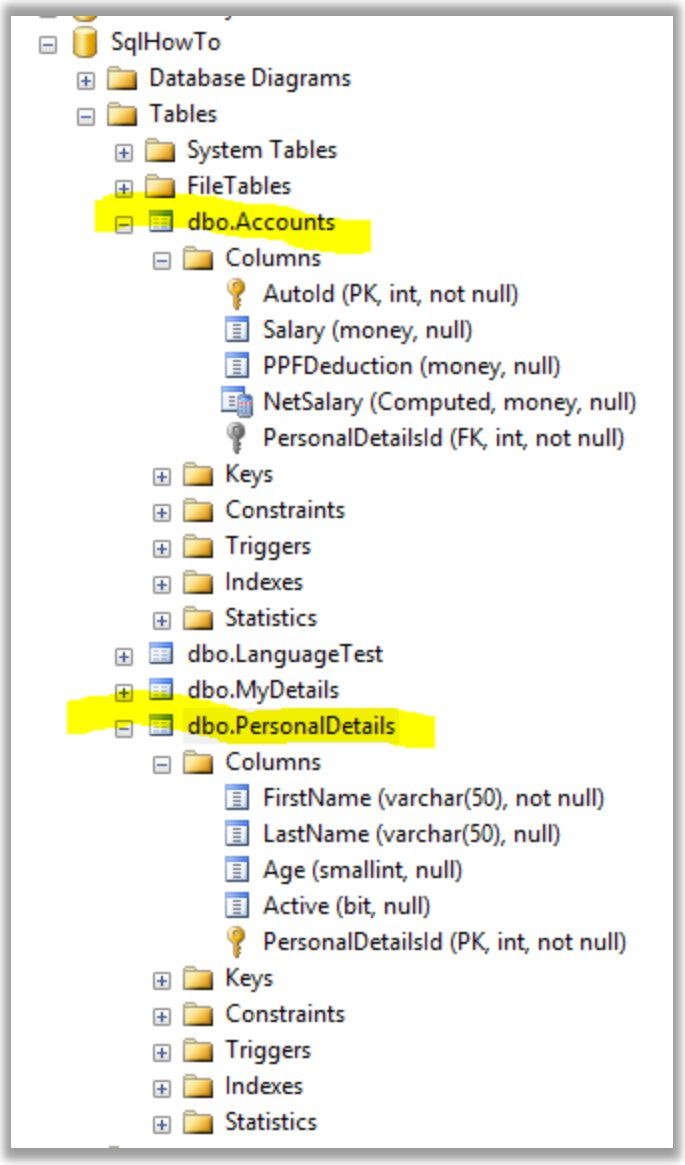
```
SELECT AutoId, FullName, City FROM MyDetails
  WHERE FullName = 'Gita' OR AutoId = 7
```



In this case, we will get all records from MyDetails table whose FullName is 'Gita' OR AutoId is '7', so it is showing both records as both conditions are match in our database table..

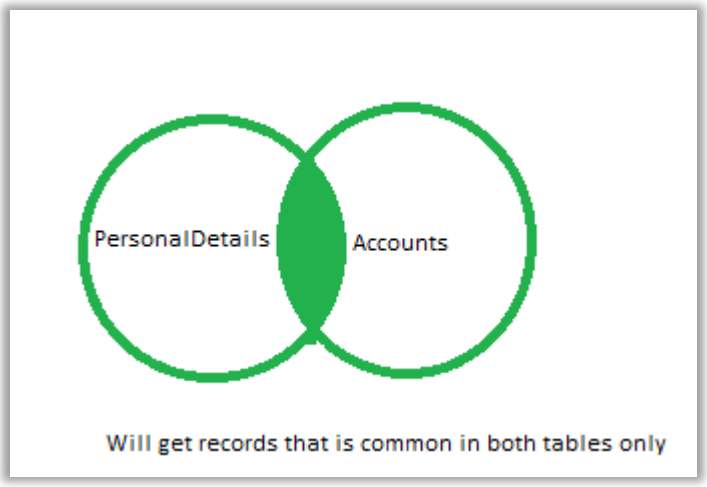
Joins

To demonstrate the joins, we will be working on two tables PersonalDetails and Accounts. Where the Primary key table is PersonalDetails with PersonalDetailsId is the primary key. The Foreign key table is Accounts with PersonalDetailsId.



34. How to join more than one table and get the results in SQL server?

Inner join or Join is used to join more than one tables and get only those records whose linked columns are present in both tables.



More than one table can be joined and get the results in following ways

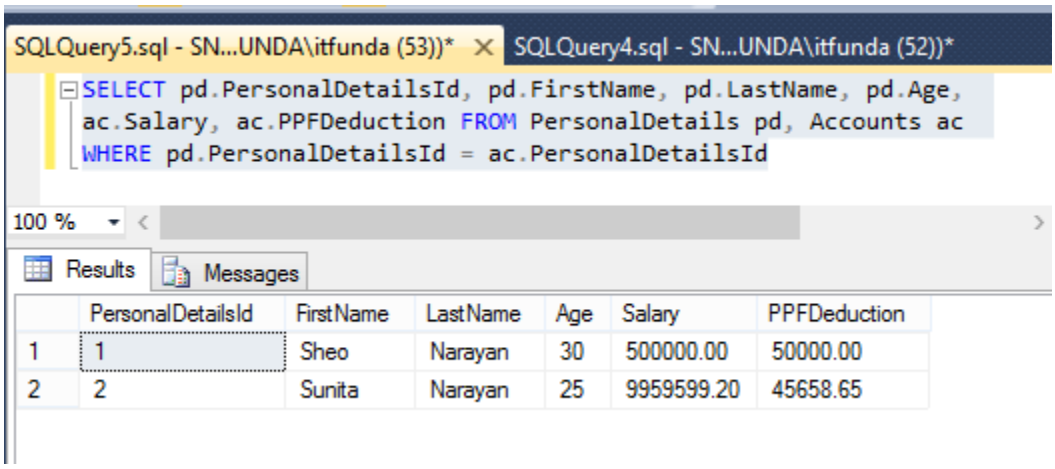
First Approach

Run following SQL statement in Query window by selecting our database.

```
SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
ac.Salary, ac.PPFDeduction FROM PersonalDetails pd, Accounts ac
WHERE pd.PersonalDetailsId = ac.PersonalDetailsId
```

Here, we are joining PersonalDetails and Accounts table. “pd” is the alias of “PersonalDetails” table and “ac” is the alias of “Accounts” table.

In the above Sql statement, we are instructing the database to select PersonalDetailsId, FirstName, LastName, Age from PersonalDetials table and Salary, PPFDeduction from Accounts table by where the value of PersonalDetails.PersonalDetialsId is equal to Accounts.PersonalDetailsId.



	PersonalDetailsId	FirstName	LastName	Age	Salary	PPFDeduction
1	1	Sheo	Narayan	30	500000.00	50000.00
2	2	Sunita	Narayan	25	9959599.20	45658.65

Similarly, more than two tables can also be joined provided at least one column is common in any of two tables.

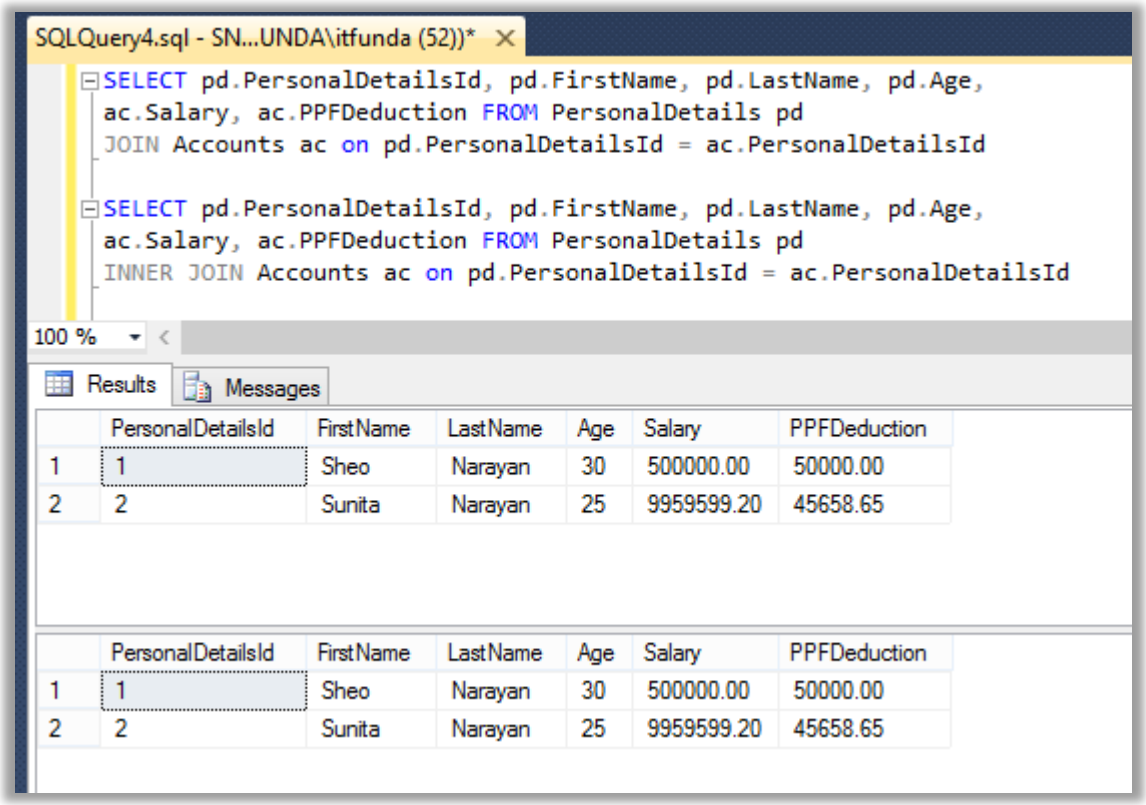
Second Approach

The same can be achieved using JOIN or INNER JOIN keyword also.

```
SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
ac.Salary, ac.PPFDeduction FROM PersonalDetails pd
JOIN Accounts ac on pd.PersonalDetailsId = ac.PersonalDetailsId

SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
ac.Salary, ac.PPFDeduction FROM PersonalDetails pd
INNER JOIN Accounts ac on pd.PersonalDetailsId = ac.PersonalDetailsId
```

Both of above statements would result the same output as we are using INNER JOIN instead of JOIN.

Here, we are getting the same output columns from both tables however the tables are being joined using JOIN or INNER JOIN keyword. Because of JOIN or INNER JOIN keywords, the approach of specifying there equality of common field "PersonalDetailsId" is different. Here we do not specify the common table in WHERE clause but using "on" keyword.



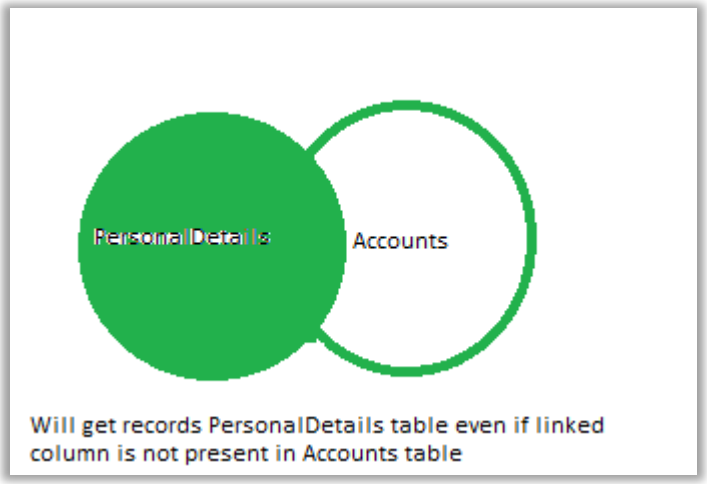
	PersonalDetailsId	FirstName	LastName	Age	Salary	PPFDeduction
1	1	Sheo	Narayan	30	500000.00	50000.00
2	2	Sunita	Narayan	25	9959599.20	45658.65

	PersonalDetailsId	FirstName	LastName	Age	Salary	PPFDeduction
1	1	Sheo	Narayan	30	500000.00	50000.00
2	2	Sunita	Narayan	25	9959599.20	45658.65

Both of above approaches would filter the same records from the database, so functionally all of three SQL Select statements are same.

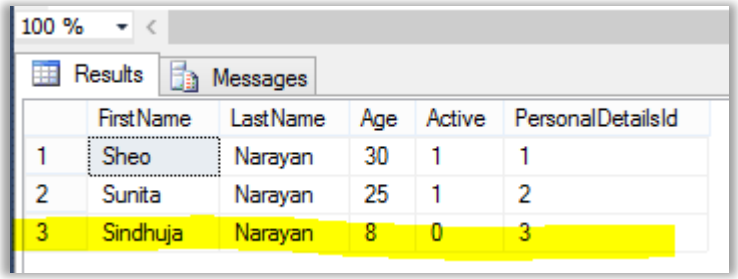
35. How to perform left join (or left outer join) in SQL Server?

Left outer join (or left join – both are same) is used to join more than one tables where we want all records from the left table (the table name that appears first in the query) even if their matching records are not present in right table (table name that appears next). If match exists in the right table, those records are presented I the result otherwise NULL values are displayed against record of left table.



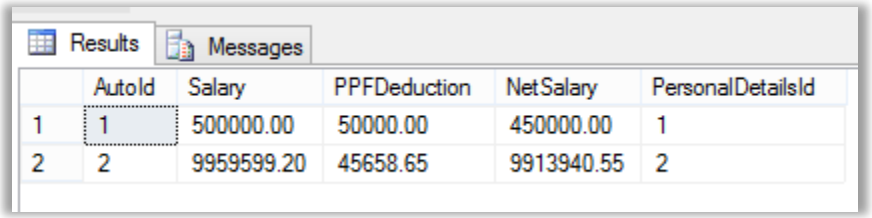
To demonstrate this example, we have two tables as shown below

PersonalDetails



	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1
2	Sunita	Narayan	25	1	2
3	Sindhuja	Narayan	8	0	3

Accounts



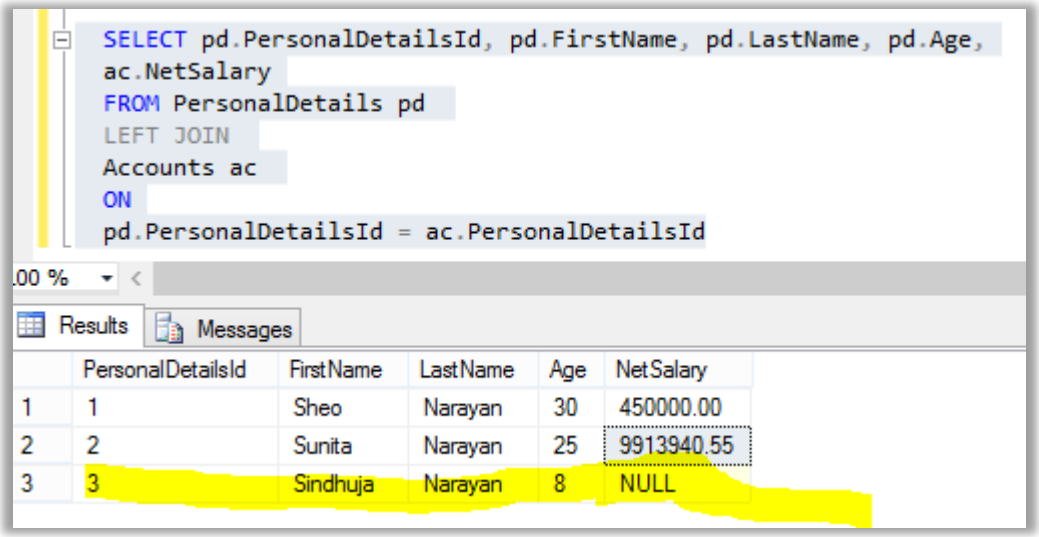
	AutoId	Salary	PPFDeduction	NetSalary	PersonalDetailsId
1	1	500000.00	50000.00	450000.00	1
2	2	9959599.20	45658.65	9913940.55	2

Now run following SQL Statement in the query window

```
SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
ac.NetSalary
FROM PersonalDetails pd
LEFT JOIN
Accounts ac
```

```
ON
pd.PersonalDetailsId = ac.PersonalDetailsId
```

Here, instead of JOIN, we have used LEFT JOIN keyword and the results clearly shows the difference. PersonalDetailsId value that doesn't exists in the Accounts table, shows NULL value for the NetSalary but for other records, it shows the value from the Accounts table as their PersonalDetailsId exists in the PersonalDetials table.



```
SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
ac.NetSalary
FROM PersonalDetails pd
LEFT JOIN
Accounts ac
ON
pd.PersonalDetailsId = ac.PersonalDetailsId
```

00 %

Results | Messages

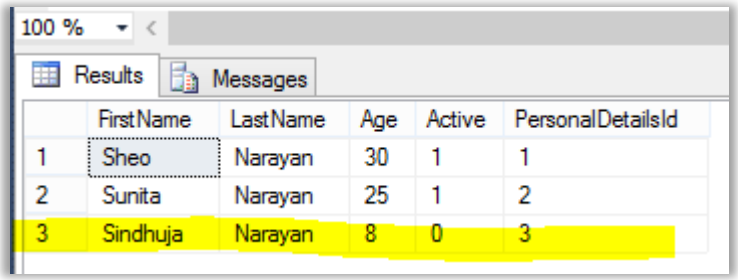
	PersonalDetailsId	FirstName	LastName	Age	NetSalary
1	1	Sheo	Narayan	30	450000.00
2	2	Sunita	Narayan	25	9913940.55
3	3	Sindhuja	Narayan	8	NULL

36. How to perform right join (or right outer join) in SQL Server?

Right outer join (or right join) is used to join more than one table where we want all records from the right table (the table that comes last in the query) and only matching records from left table (table name that comes first in the query).

To demonstrate this example, we have two tables as shown below

PersonalDetails

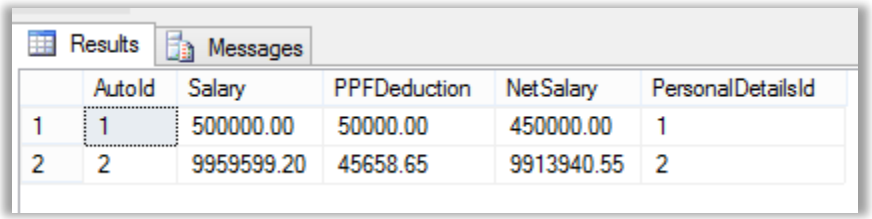


100 %

Results | Messages

	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1
2	Sunita	Narayan	25	1	2
3	Sindhuja	Narayan	8	0	3

Accounts



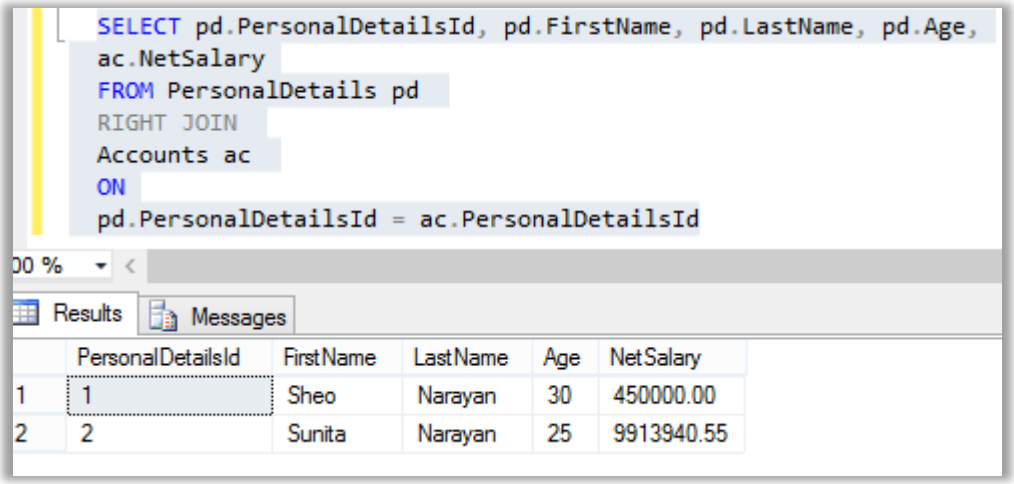
Results | Messages

	AutoId	Salary	PPFDeduction	NetSalary	PersonalDetailsId
1	1	500000.00	50000.00	450000.00	1
2	2	9959599.20	45658.65	9913940.55	2

Now run the following SQL statement in the query window.

```
SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
  ac.NetSalary
  FROM PersonalDetails pd
  RIGHT JOIN
  Accounts ac
  ON
  pd.PersonalDetailsId = ac.PersonalDetailsId
```

Here, instead of LEFT JOIN, we have used RIGHT JOIN that changes the result based on right side table data. So it will give all records from Accounts table and only matching records from the PersonalDetails table.



```
SELECT pd.PersonalDetailsId, pd.FirstName, pd.LastName, pd.Age,
ac.NetSalary
FROM PersonalDetails pd
RIGHT JOIN
Accounts ac
ON
pd.PersonalDetailsId = ac.PersonalDetailsId
```

00 %

Results | Messages

	PersonalDetailsId	FirstName	LastName	Age	NetSalary
1	1	Sheo	Narayan	30	450000.00
2	2	Sunita	Narayan	25	9913940.55

View

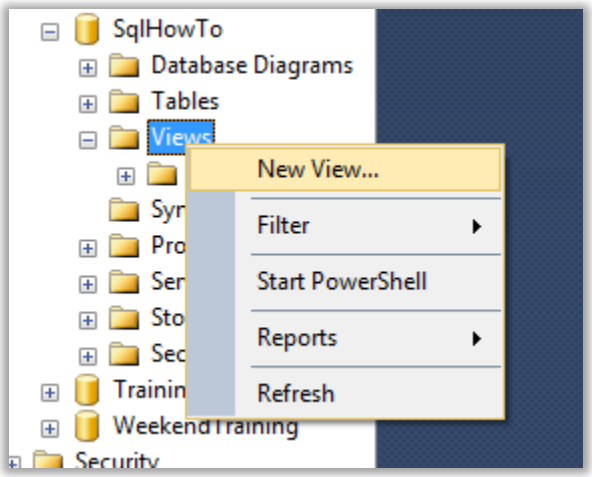
As the name suggest, a view is a virtual object whose result is derived from SQL query. View doesn't contains any physical data, but all data comes from the database table. View is only stored physically in the disk when it is indexed.

Views are used to

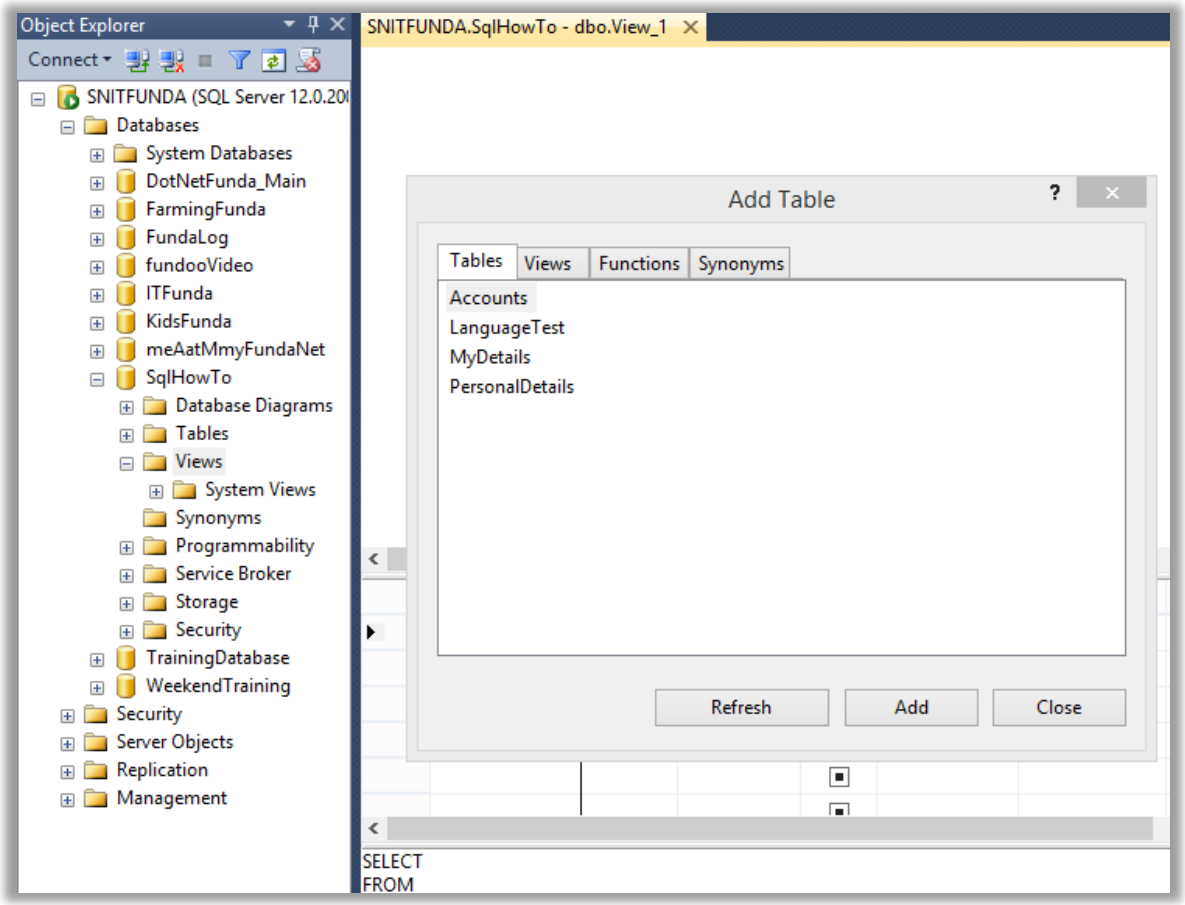
i. Filter the table data
ii. Hide some columns data from a database table and provide only limited set of columns
iii. Create a reusable set of data

37. How to create a View in SQL Server?

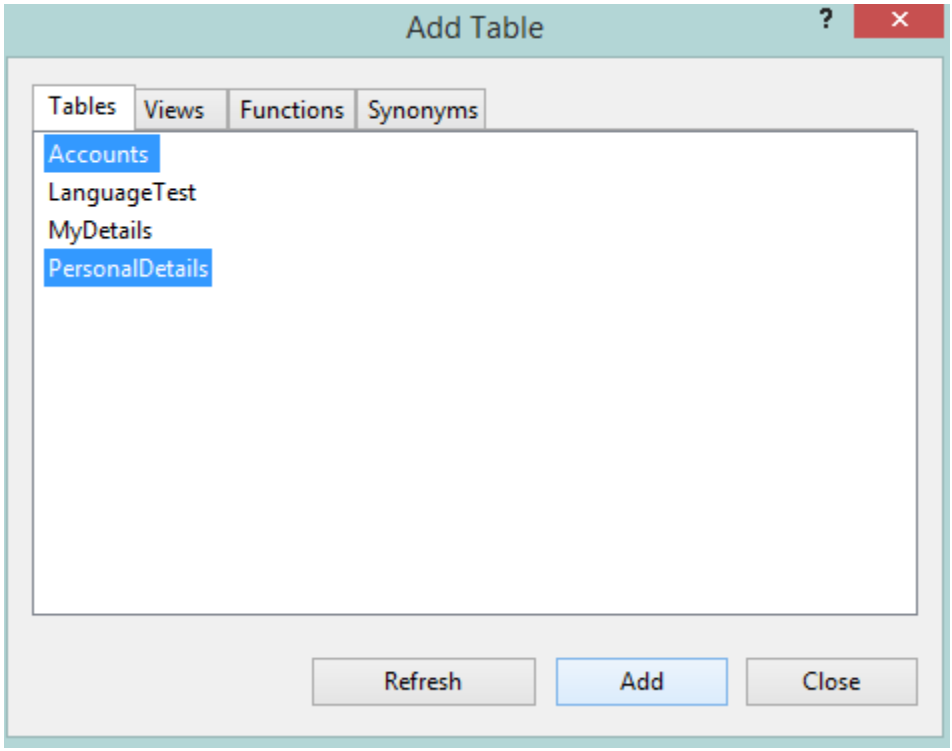
To create a View in SQL Server, explore the Database and right click the Views folder and click New View…



That shows an Add Table dialog box like below



Now click on the table we want to create View for, in this example we are going to select Accounts and PersonalDetails table by holding the ctrl (from keyboard) key and clicking on the table name. Now click on Add button.

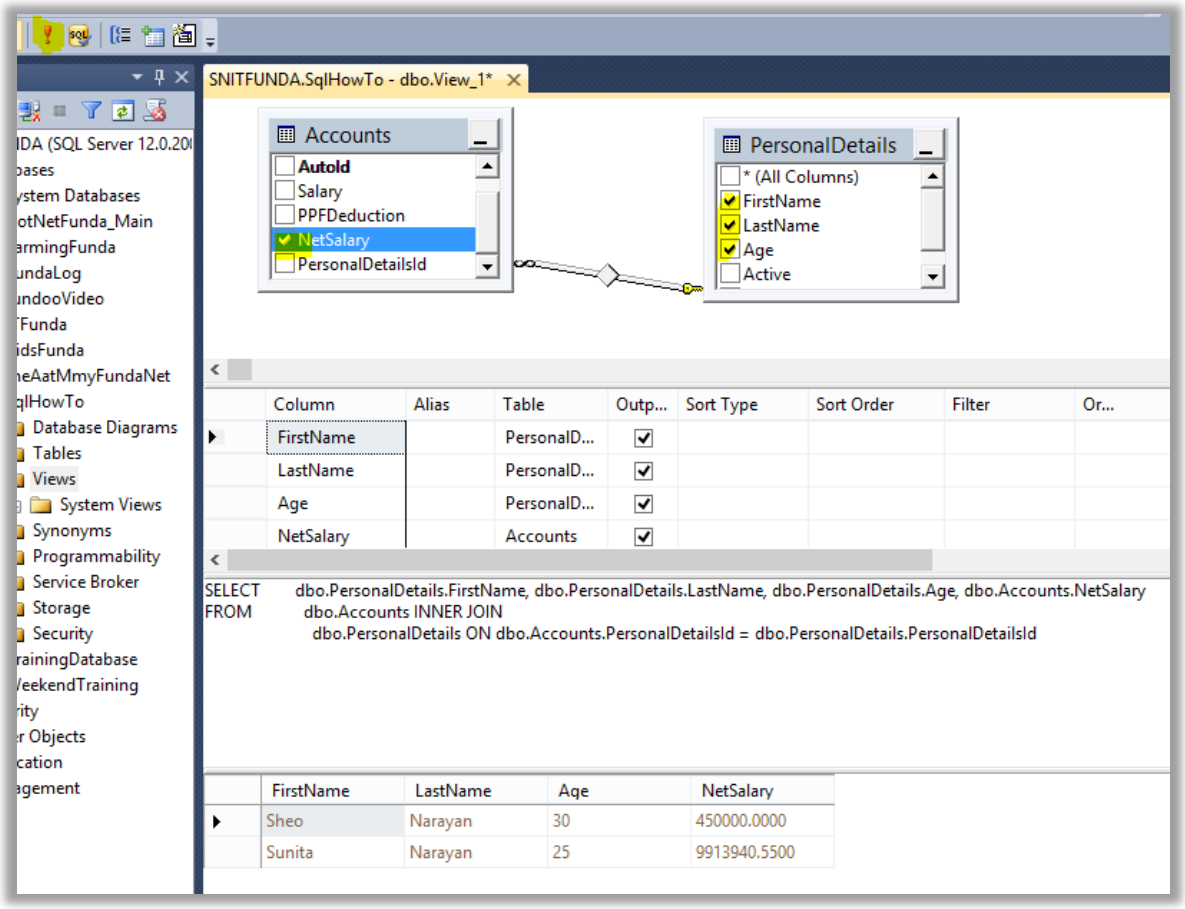


Now, click Close button. As the PersonalDetailsId column of PersonalDetails and Accounts table are same (and relationship is created) so we will see a relationship between these two columns.

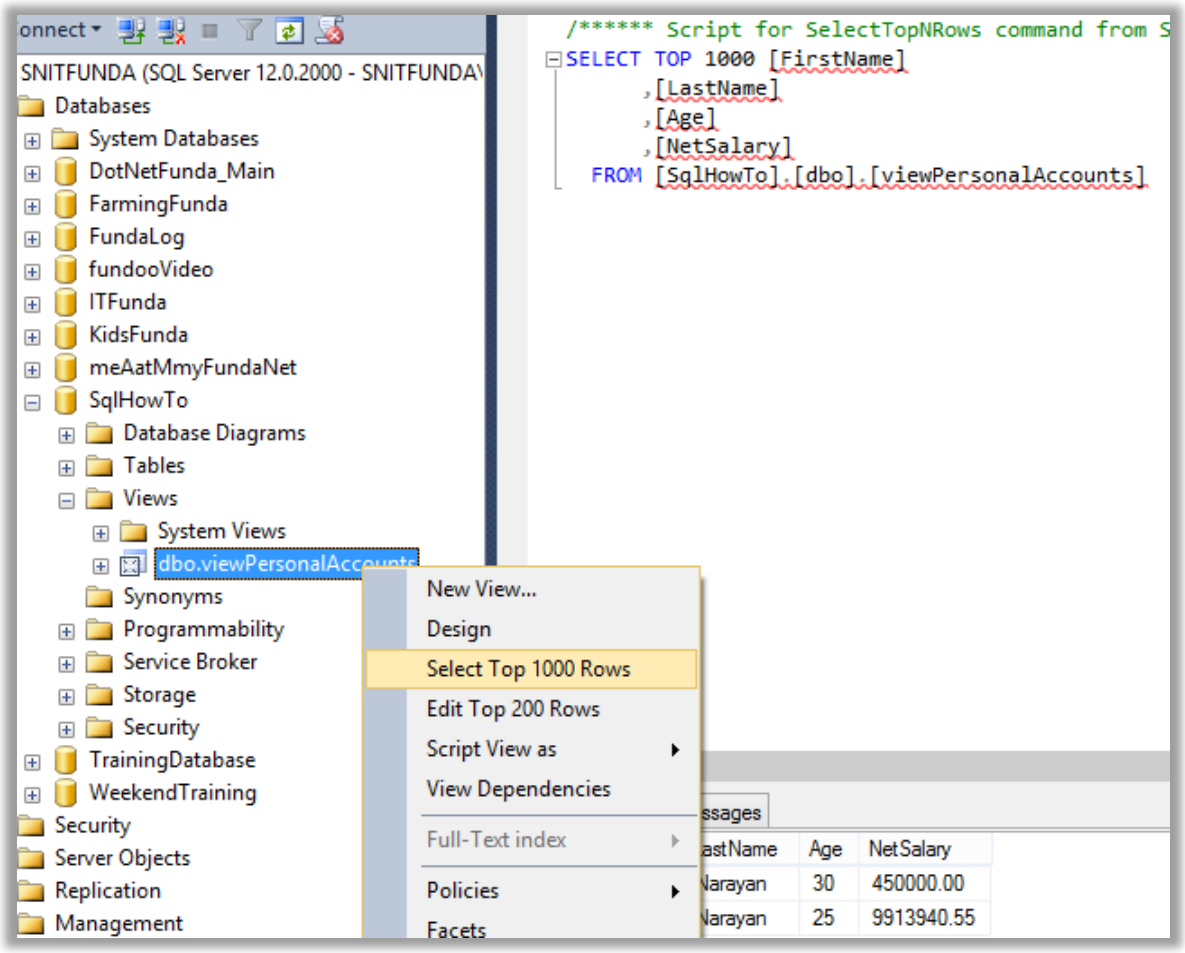
Now, check the columns checkbox from the list of tables selected from the 1st panel that will add those columns into the 2nd panel and corresponding SQL statements are written into the 3rd panel. When we want to see the result of the query automatically built after selecting the relationship and column names, we can click on "!" icon from the toolbar at the top-left and see the result in the 4th panel.

If we want to create another relationship between the column names of selected tables, we can click on the column of the table and hold and drag to another table column and leave it.

Once everything is done and we are happy with the result shown in the 4th panel, save the view by clicking on the Save icon from the toolbar in the top-left or from File menu (write the View name to save).



Now, refresh the Views folder again and we will see the View just created. To see the result of this Views, right click the View name and choose “Select Top xxx Rows” that will write the SQL select element for this View as if it is a database table and show the data in the below panel.



In this view result, notice that without writing the Joins statement between PersonalDetails and Accounts table, we are able to get columns of PersonaDetails and Accounts table because these join statements are already written when we had created the View.

In the same way, we can create a new view for very complex SQL Select statement and refer that view instead of writing that complex SQL statement every time.

Important

It is not mandatory to prefix the view name with “view” or “vw” (as shown above – viewPersonalAccounts) however it is recommended so that by reading name, we can understand that this is a View or a physical table. The way to work with physical database tables and views in the SELECT queries are same so there must be some identifiable difference in the name; so that a developer can easily know whether the object he is working with is a database table or view.

38. How to alter a Views in SQL Server database?

To alter a View, right click the View and choose “Design”, this will bring the View in the design mode as we had seen while creating the Views. Change whatever we want to change in top 3 columns and save it.

Variable declaration and setting value

Variables are important part of any programming or scripting language. We can declare variable and set its value in SQL Server. This is particularly useful while creating Stored Procedures, Functions etc. We shall discuss about these later on.

39. How to declare and use a variable in SQL Server?

Variables can be declared in SQL Server using DECLARE keyword and the variable name in SQL Server is prefixed with “@” character.

A variable declaration starts with DECLARE keyword followed by @variableName and then the data type of the variable.

```
DECLARE @firstName varchar(50)
```

In above case, @firstName is the variable name that is of varchar type and its size is 50 (max characters can be stored in this variable is 50).

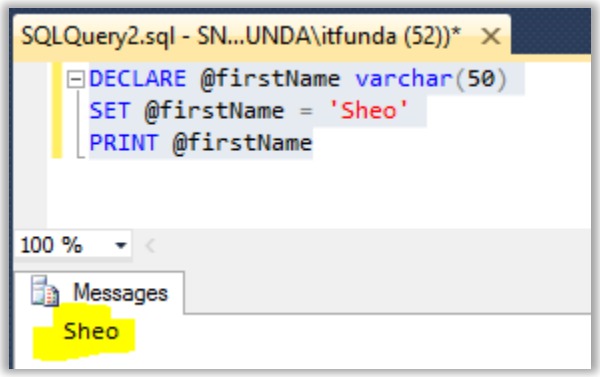
To set a value into the variable, we can use SET keyword followed by the variable name and then the value.

```
SET @firstName = 'Sheo'
```

To print the value stored into the variable, we can use PRINT keyword. The complete code snippet looks like below.

```
DECLARE @firstName varchar(50)
SET @firstName = 'Sheo'
PRINT @firstName
```

Running above set statements, ‘Sheo’ is written in the Message window.



Query

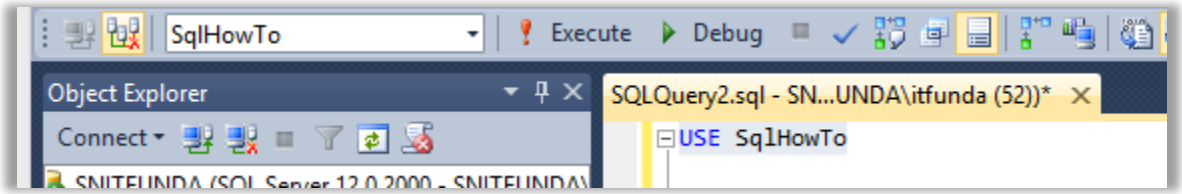
We use SELECT statement followed by database table field names to query records from the database.

40. How to select a database to work with in Query window in SQL Server?

We can use "use" keyword with the database name.

```
USE SqlHowTo
```

Above statement will change the database dropdown value to SqlHowTo like below

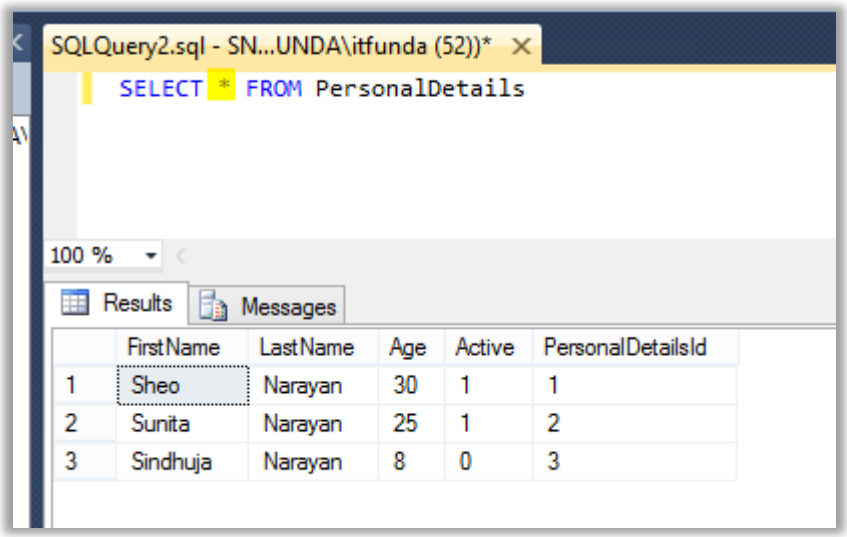


41. How to select all columns and records from the database table in SQL Server?

To get all columns and records from the database table, we can use "*" with the SELECT statement.

```
SELECT * FROM PersonalDetails
```

Execution of above statements in query window will show all records and columns data of the PersonalDetails table.



	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1
2	Sunita	Narayan	25	1	2
3	Sindhuja	Narayan	8	0	3

42. How to return only few columns data from a SQL Server database?

To return only few columns from a database table, we write column names followed by the SELECT statement. In general, columns are written separated by comma however if the column name contains blank space, we warp them with square bracket "[column name].

```
SELECT FirstName, LastName, Age FROM PersonalDetails
```

```
SELECT [FirstName], [LastName], [Age] FROM PersonalDetails
```

Above, both query would return the same result. In 1st case directly field names are written and in 2nd case field names are wrapped with "[ ]".

43. How to return top n records from the database in SQL Server?

To return top few records from the database, we use TOP keyword.

```
SELECT TOP(2) FirstName, LastName, Age FROM PersonalDetails
```

Above query will return Top 2 records (only FirstName, LastName, Age columns data) from PersonalDetails table.

```
SELECT TOP(2) * FROM PersonalDetails
```

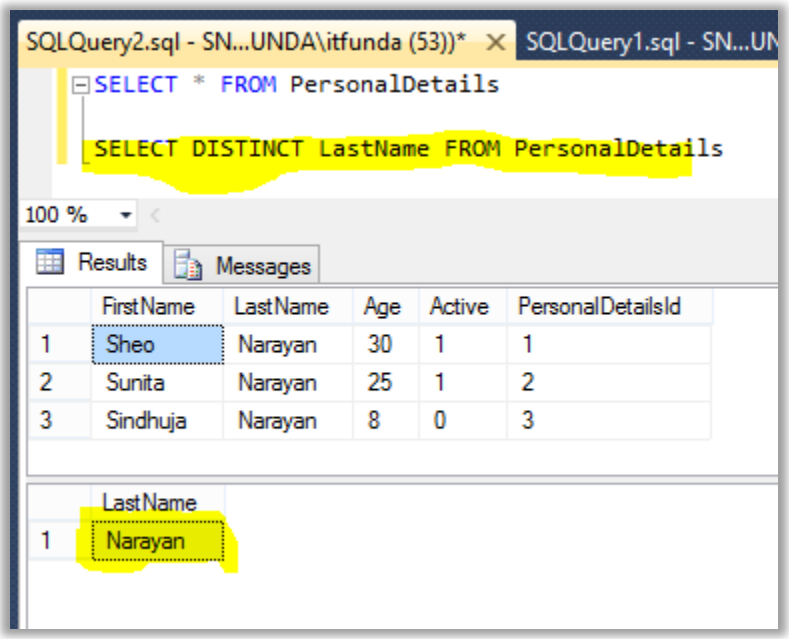
Above query would return Top 2 records (all columns) from PersonalDetails table.

44. How to retrieve distinct (unique) record from SQL Server database?

To return distinct (unique) value of a particular column from a database table, we use DISTINCT keyword.

```
SELECT DISTINCT LastName FROM PersonalDetails
```

Above query will only list unique LastName value from the PersonalDetails table.



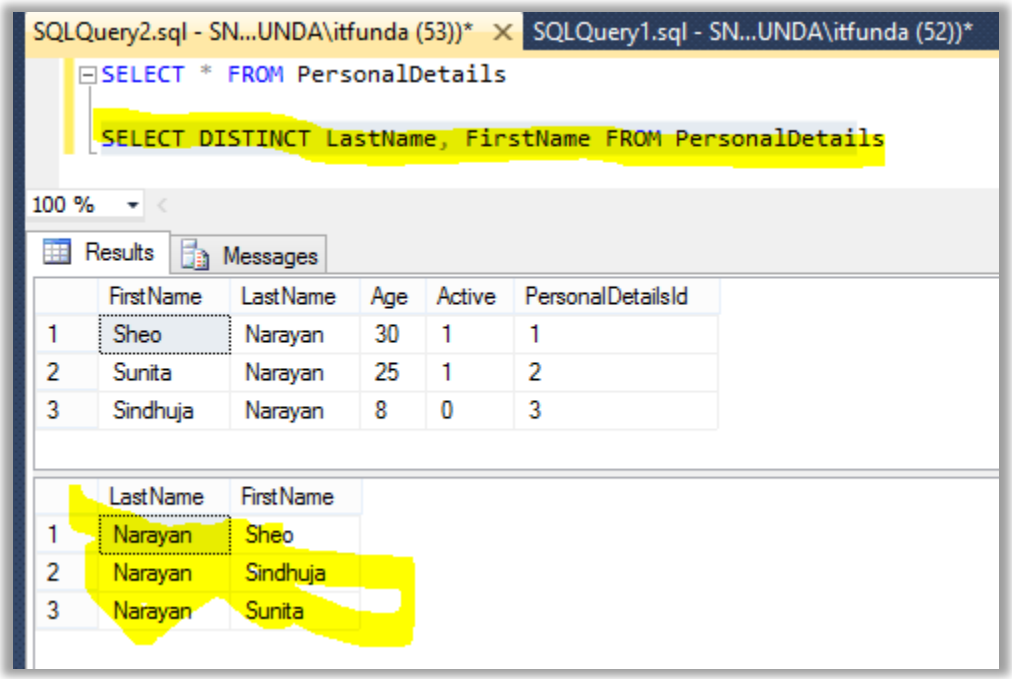
In above case, the all 3 records have LastName as “Narayan”, only one unique value so only “Narayan” is listed as the result of the 2nd query.

Important

If we keep more than only column with the DISTINCT keywords, uniqueness is determined with the combined value of all those columns.

```
SELECT DISTINCT LastName, FirstName FROM PersonalDetails
```

In above query, all three records will be given as result because the combined value of LastName and FirstNames are unique.



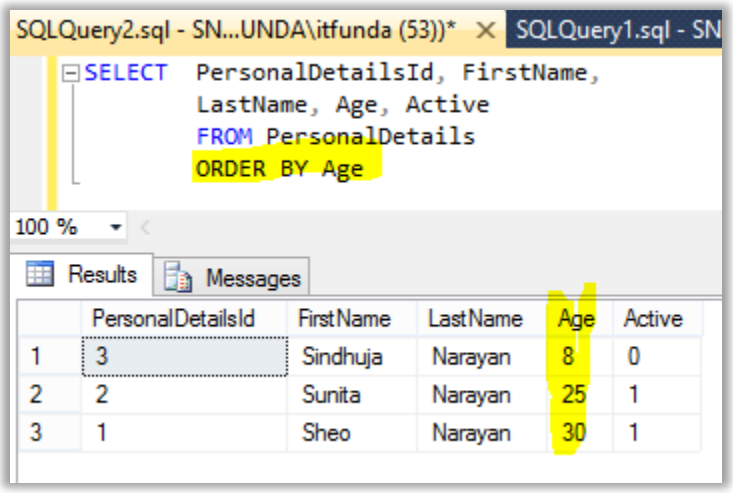
45. How to sorted records from the SQL Server database table?

To return sorted records from SQL Server database table, we use ORDER BY followed by column name to sort and ASC (for ascending order) and DESC (for descending order) keywords.

If ASC or DESC is not specicied, ASC is treated by default.

```
SELECT PersonalDetailsId, FirstName,
              LastName, Age, Active
              FROM PersonalDetails
              ORDER BY Age
```

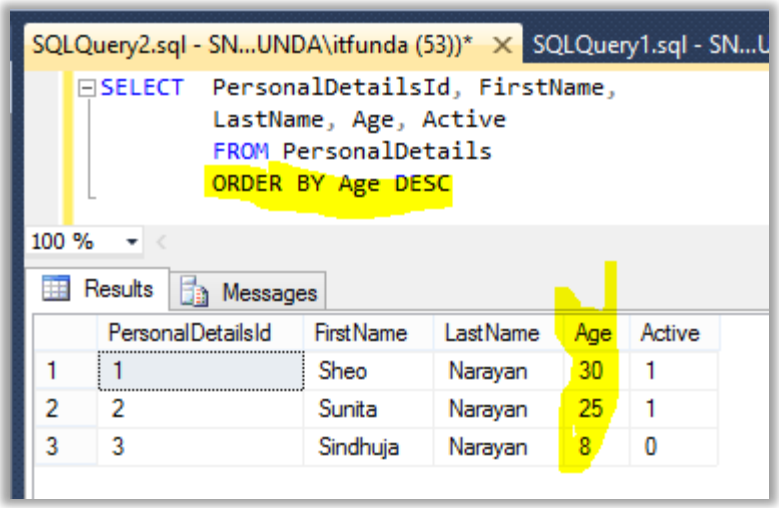
Above query would sort all records from the PersonalDetails table based on Age column value in ascending order.



See following query

```
SELECT PersonalDetailsId, FirstName,
              LastName, Age, Active
              FROM PersonalDetails
              ORDER BY Age DESC
```

Keeping the DESC keyword in the last, will sort the records based on Age in descending order.



46. How to sort records based on multiple columns in SQL Server database?

Records can be sorted based on multiple columns values too, to do that separate the column name by comma (“,”) followed by ASC or DESC keywords.

```
SELECT PersonalDetailsId, FirstName,
              LastName, Age, Active
```

```
FROM PersonalDetails
ORDER BY Age, Active DESC
```

Above query will give the sorted records based on Age (in ascending order as if not specified, ASC is treated by default) and then based on Active column values.

Important:

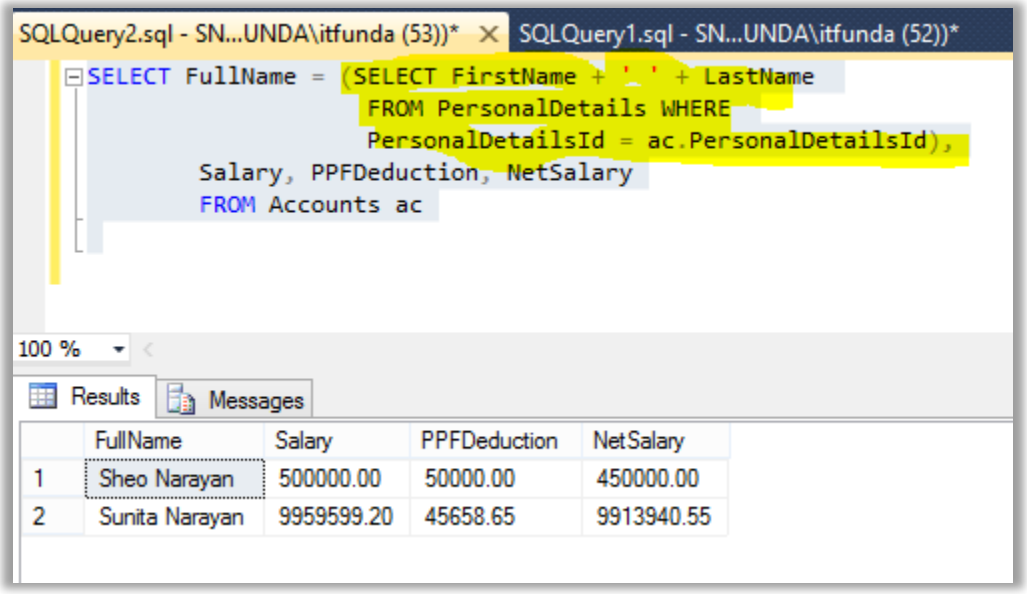
Sorting can be done on any type of data columns, its SQL Server responsibility to sort the record based on data type's values and give us the correct result.

47. How to use sub-query to return data from database?

Sub query can be used to a get data from another database table along with current database table used in the main query.

```
SELECT FullName = (SELECT FirstName + ' ' + LastName
                   FROM PersonalDetails WHERE
                   PersonalDetailsId = ac.PersonalDetailsId),
       Salary, PPFDeduction, NetSalary
       FROM Accounts ac
```

Above query returns FullName from PersonalDetails table (combination of FirstName and LastName) and Salary, PPFDeduction and NetSalary from Accounts table. Notice the alias "ac" used for Accounts tbale.



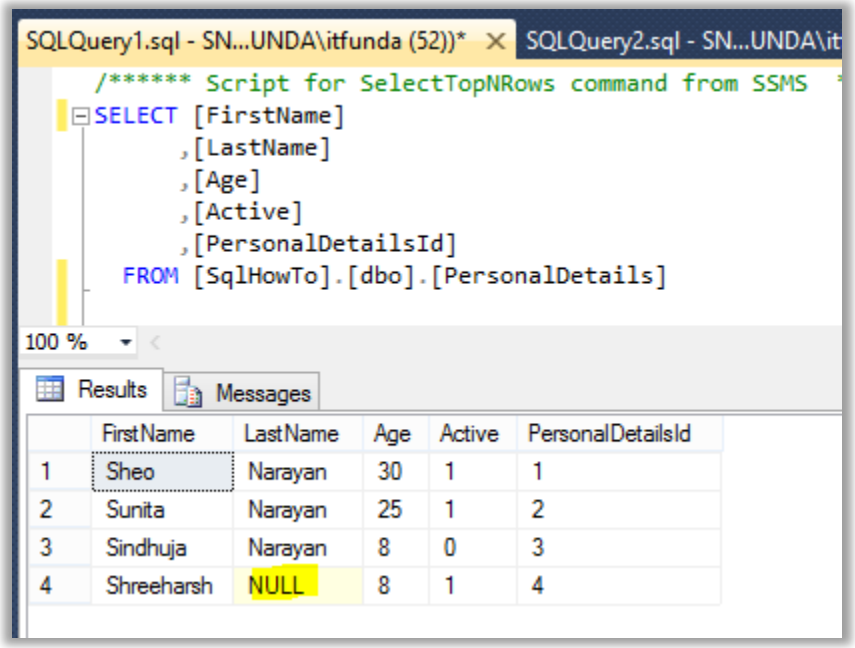
	FullName	Salary	PPFDeduction	NetSalary
1	Sheo Narayan	500000.00	50000.00	450000.00
2	Sunita Narayan	9959599.20	45658.65	9913940.55

In case, no records found in PersonalDetails table, null value would be returned.

48. How to get records having null value in a column of SQL Server database table?

To retrieve all records whose a particular column value is NULL, we can use “is” keyword followed by null.

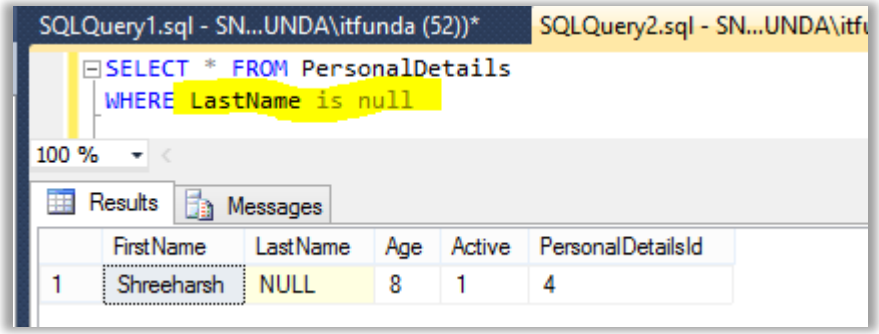
Assume that we have 4 records in the database table and we want to return only that record from the database whose LastName is null.



Execute following query in the query window.

```
SELECT * FROM PersonalDetails
WHERE LastName is null
```

This would return only one record from the PersonalDetails database table.



49. How to use not equal to condition in the query of SQL Server?

There are two variance of using not equal to. First is in case of checking for null value and other for any data type values. To check for null value, we use “not” keyword and for other datatype we use ‘<>’.

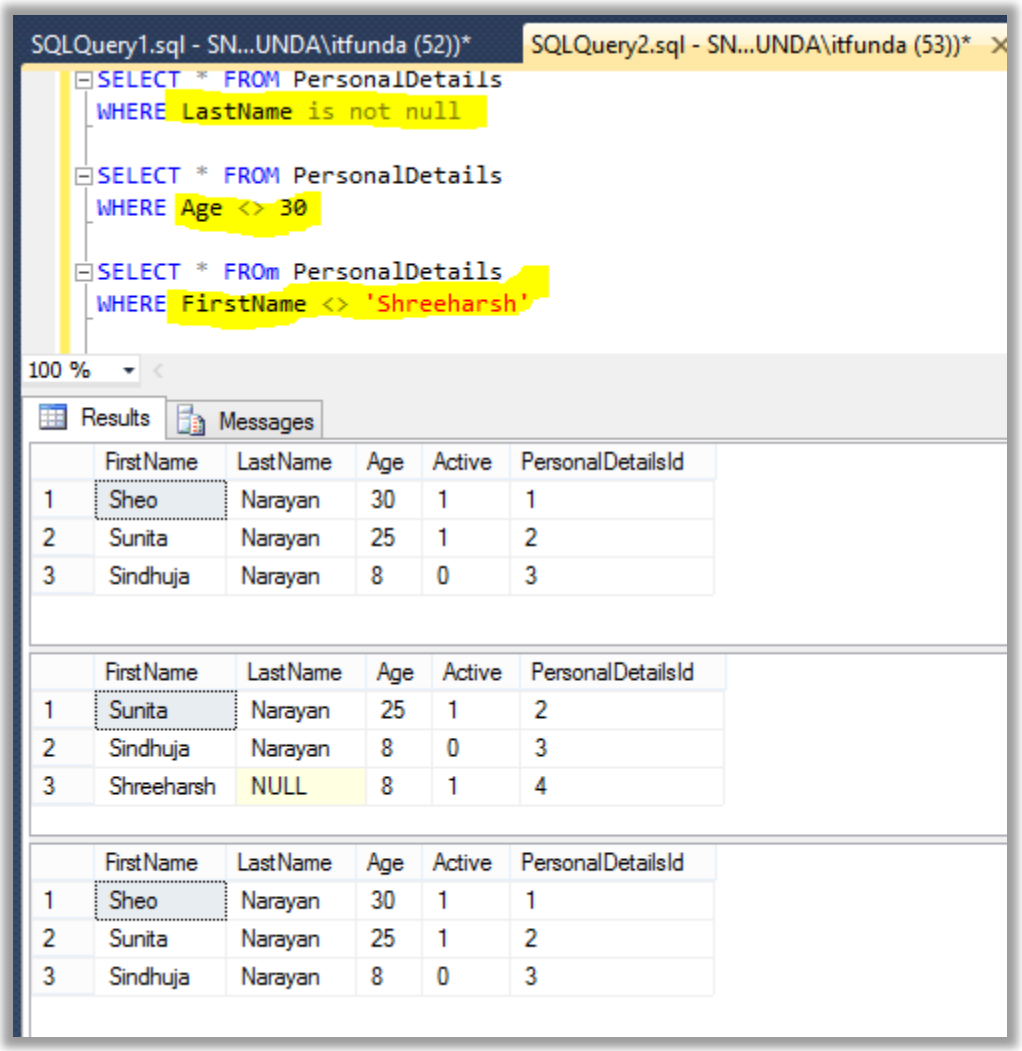
NOTE: ‘<>’ and ‘!=’ gives equal meaning in SQL Server and interchangeably can be used however ‘<>’ is recommended to use as it is of ANSI SQL standard.

```
SELECT * FROM PersonalDetails
WHERE LastName is not null

SELECT * FROM PersonalDetails
WHERE Age <> 30

SELECT * FROm PersonalDetails
WHERE FirstName <> 'Shreeharsh'
```

The 1st SELECT statement retrieves all records from the table whose LastName is not null.

The 2nd SELECT statement retrieves all records from the table whose Age is not 30.

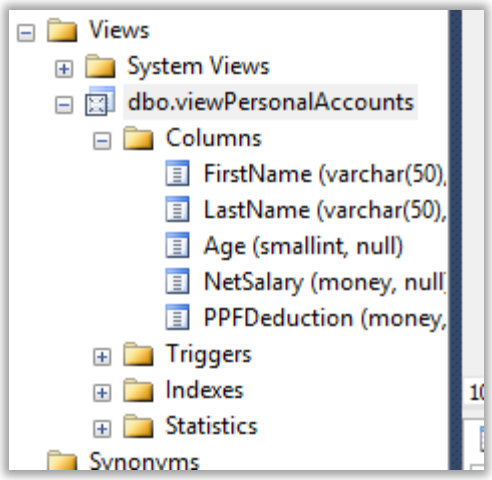
The 3rd SELECT statement retrieves all records from the table whose FirstName is not equal to ‘Shreeharsh’



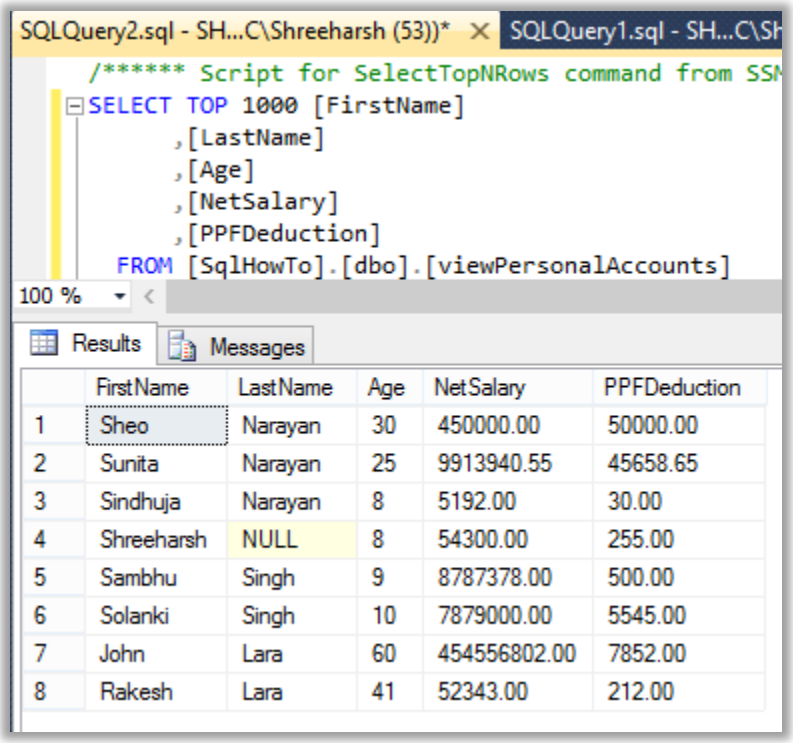
50. How to return records in Group by a column value in SQL Server?

To return records from the database based on group by a certain columns, we use Group by keyword. This is generally done by those columns whose values are repeatative in nature.

To demonstrate this, we have used a view that was created as part of View examples discussed above.



This view returns FirstName, LastName, Age from PersonalDetails table and NetSalary, PPFDeduction from Accounts table.

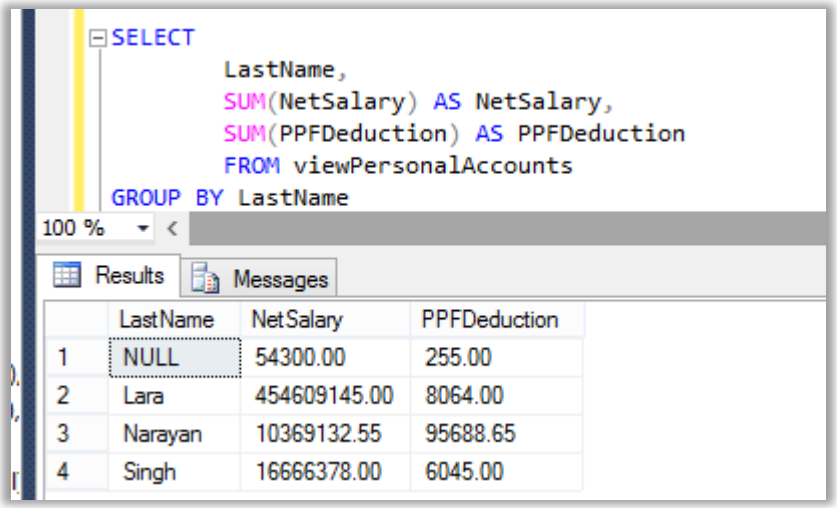


	FirstName	LastName	Age	NetSalary	PPFDeduction
1	Sheo	Narayan	30	450000.00	50000.00
2	Sunita	Narayan	25	9913940.55	45658.65
3	Sindhuja	Narayan	8	5192.00	30.00
4	Shreeharsh	NULL	8	54300.00	255.00
5	Sambhu	Singh	9	8787378.00	500.00
6	Solanki	Singh	10	7879000.00	5545.00
7	John	Lara	60	454556802.00	7852.00
8	Rakesh	Lara	41	52343.00	212.00

Now write following sql statement in the query window.

```
SELECT
            LastName,
            SUM(NetSalary) AS NetSalary,
            SUM(PPFDeduction) AS PPFDeduction
            FROM viewPersonalAccounts
GROUP BY LastName
```

This will sum NetSalary, PPFDeduction and give us the result grouped by LastName



	LastName	NetSalary	PPFDeduction
1	NULL	54300.00	255.00
2	Lara	454609145.00	8064.00
3	Narayan	10369132.55	95688.65
4	Singh	16666378.00	6045.00

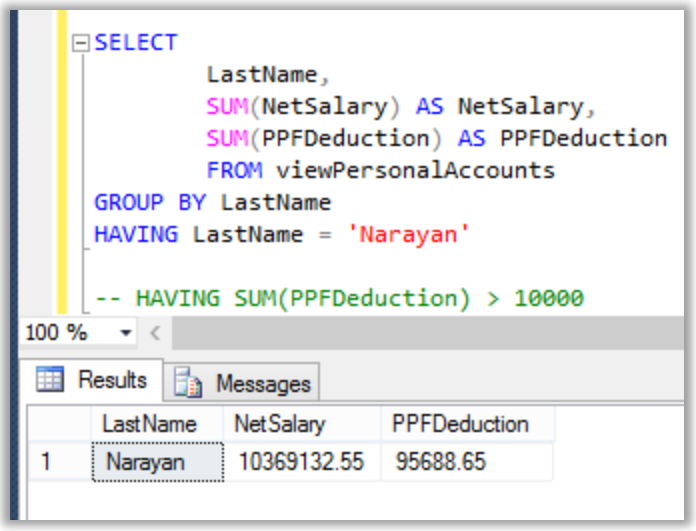
Note: It is important to have the columns that are being used as Group by in the SELECT statement.

51. How to filter result returned from Group by clause in SQL Server?

To filter the result returned from the GROUP BY clause, we use HAVING clause. Remember that we can't use WHERE clause to filter records with the GROUP BY clause.

```
SELECT
            LastName,
            SUM(NetSalary) AS NetSalary,
            SUM(PPFDeduction) AS PPFDeduction
            FROM viewPersonalAccounts
GROUP BY LastName
HAVING LastName = 'Narayan'
```

In the above statement, we have filtered the result of GROUP BY having LastName = 'Narayan'.



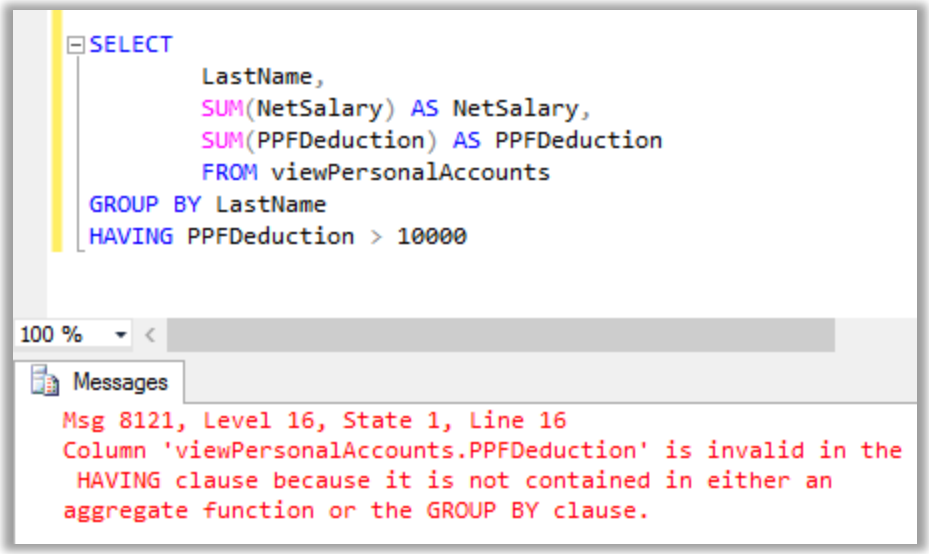
Similarly, we can also filter by the SUM of either NetSalary or PPFDeduction.

Important

Only those fields are allowed in the HAVING clause that is the part of SELECT statement. If SELECT statement has Aggregate function with column names, HAVING clause must have that Aggregate functions with the column name.

Like, below statement will not work

```
SELECT
            LastName,
            SUM(NetSalary) AS NetSalary,
            SUM(PPFDeduction) AS PPFDeduction
            FROM viewPersonalAccounts
GROUP BY LastName
HAVING PPFDeduction > 10000
```



As the SELECT statement as SUM aggregate functions with the PPFDeduction but HAVING is directly trying to compare the value of PPFDeduction column.

52. How to use wild card characters in SQL Server database query?

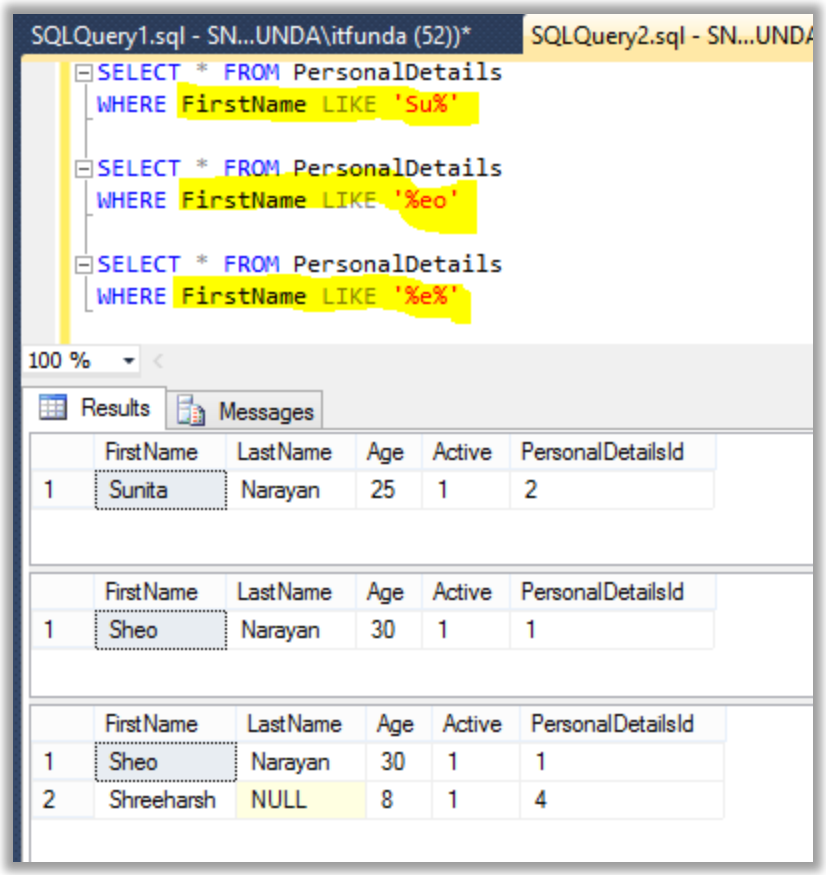
One use of wild characters we have already seen in retrieving all records from the database. (SELECT * FROM PersonalDetails). It can also be used to retrieve records that starts with or ends with or contains a particular characters.

```
SELECT * FROM PersonalDetails
WHERE FirstName LIKE 'Su%'

SELECT * FROM PersonalDetails
WHERE FirstName LIKE '%eo'

SELECT * FROM PersonalDetails
WHERE FirstName LIKE '%e%'
```

The 1st SELECT statement returns all records from the table whose FirstName starts wtih “Su” followed by any characters.

The 2nd SELECT statement returns all records from the table whose FirstName ends with “eo” preceded with any characters.

The 3rd SELECT statement returns all records from the table whose FirstName contains “e” anywhere in the FirstName column value.

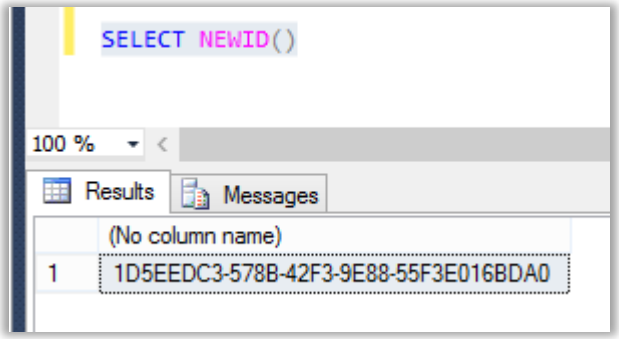


53. How to get a random unique value in SQL Server?

To get a random unique value in SQL Server, we use NEWID function.

```
SELECT NEWID()
```

Every execution of the above stamen gives a new unique value.

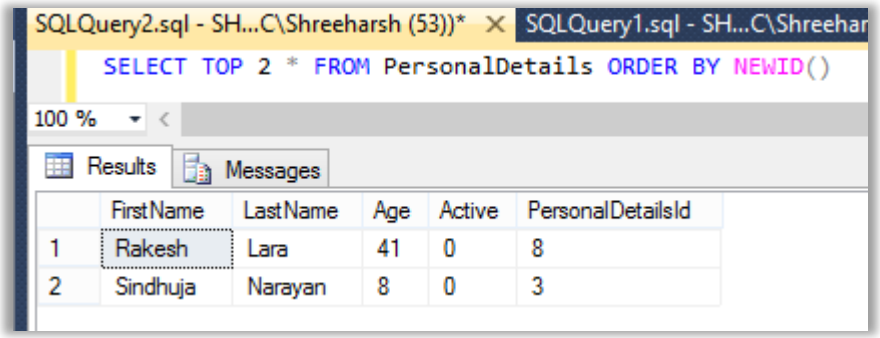


54. How to retrieve random records from the database table in SQL Server?

To retrieve random records from database table, we use NEWID inbuilt function.

Execute below statement in Query window and we will get random two records from the PersonalDetails table. If we need more or less, simply change the value of "2" in the query.

```
SELECT TOP 2 * FROM PersonalDetails ORDER BY NEWID()
```



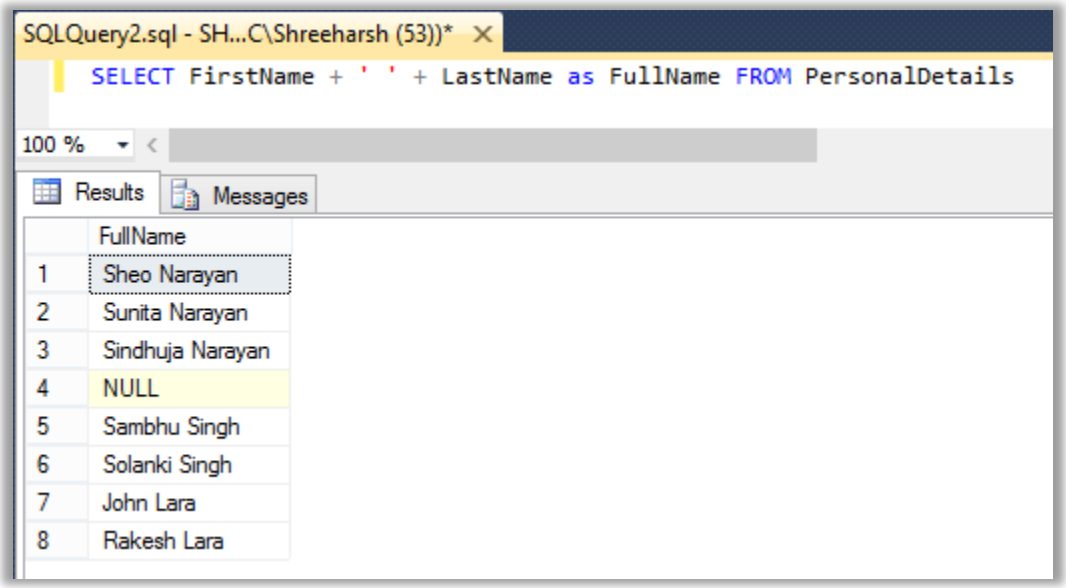
NEWID() function randomly creates a unique ID and sort the record based on that and list TOP 2 records from the database table.

55. How to merge two column value as one in SQL Server?

To merge two columns value as one, we can concatenate it as one and use alias for that value. This is the simplest way to do it.

```
SELECT FirstName + ' ' + LastName as FullName FROM PersonalDetails
```

Here the combination of FirstName and LastName is separated by a blank space and given as FullName.

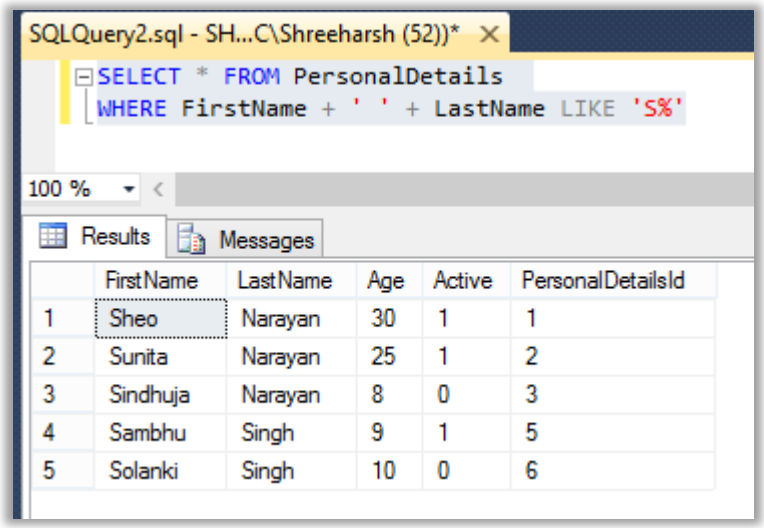


56. How to get records from the table that contains a specific word (in SQL Server)?

To get records from the table that contains a specific word, we can combine the columns in a single value and use wild card characters.

```
SELECT * FROM PersonalDetails
WHERE FirstName + ' ' + LastName LIKE 'S%'
```

Above script will list all records from the PersonalDetails table whose combination of FirstName and LastName starts with "S".

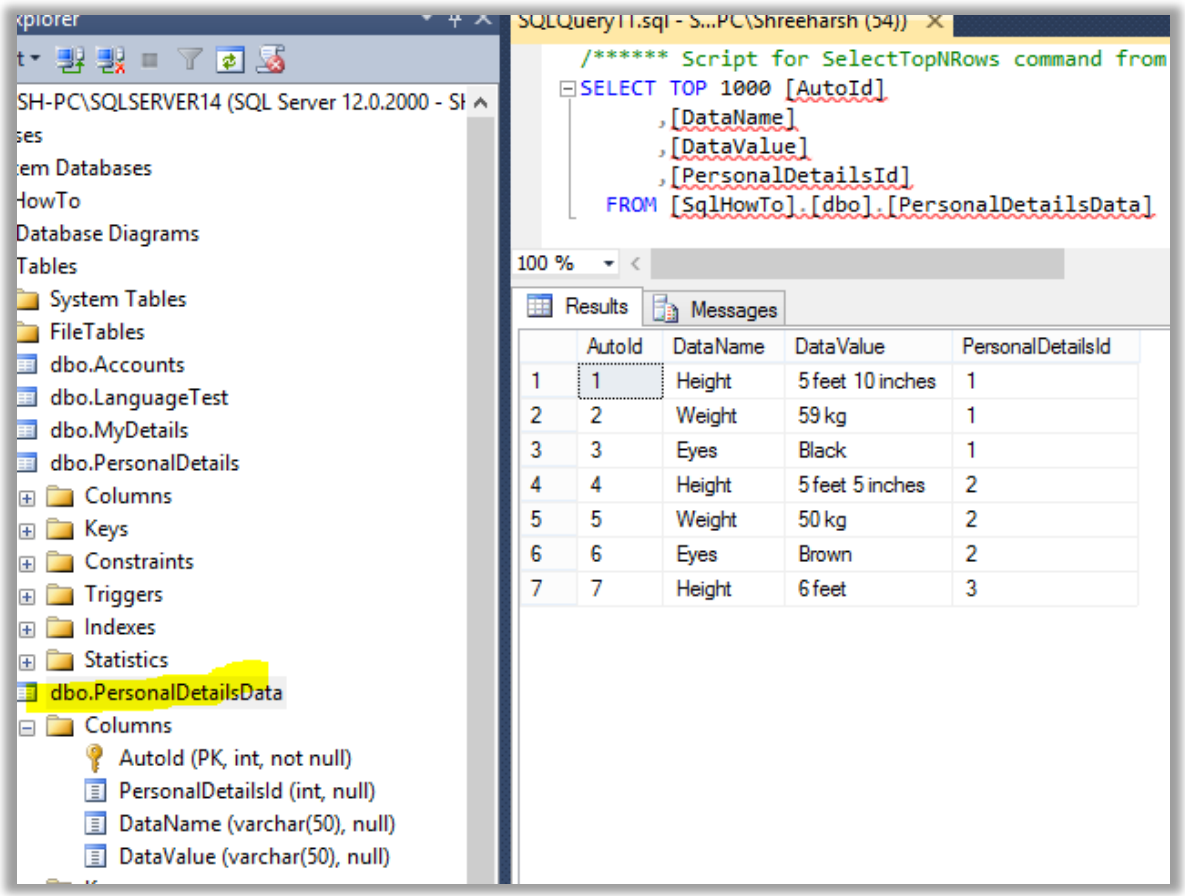


For more wild card use in the SQL query, read wild card related points demonstrated in this eBook.

57. How to convert rows into columns of the database tables in SQL Server?

Many a times, our database table structure are not in proper ways where our data are stored into rows and columns format where rows are only data.

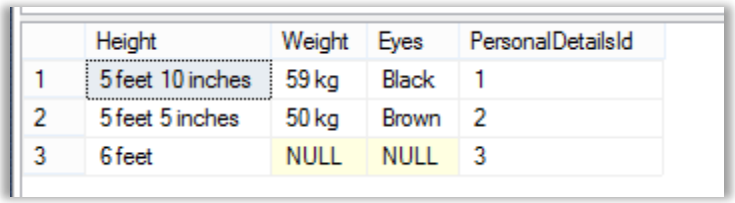
Look at this database table design.



In PersonalDetailsData table, we are storing specific attributes of a person into DataName rows and its value in DataValue rows. To identify the person for which these data are stored, we have a PersonalDetailsId column that is the foreign key of primary table PersonalDetails.

Using SELECT record from database table gives data in above format that is very tough to present and understand to the end user.

The solution of this problem is to convert the unique DataName values into columns and give its values as DataValue in each rows like bellow



	Height	Weight	Eyes	PersonalDetailsId
1	5 feet 10 inches	59 kg	Black	1
2	5 feet 5 inches	50 kg	Brown	2
3	6 feet	NULL	NULL	3

To do this, we need to use PIVOT clause like below.

```
SELECT Height, [Weight], Eyes, PersonalDetailsId
    FROM
        (
            SELECT DataName, DataValue, PersonalDetailsId,
            'param'+ cast(ROW_NUMBER() OVER
            (
                PARTITION BY PersonalDetailsId, DataName, DataValue
                ORDER BY DataName) as varchar(50)) PD
            FROM
            PersonalDetailsData
        ) PDD
    PIVOT
    (
        max(DataValue)
        FOR
        DataName
        IN
        (Height, [Weight], Eyes)
    ) piv
```

In the above query, we are selecting the Height, [Weight], Eyes and PersonalDetailsId from subquery formed with the help of PARTITION BY clause.

The first set of selected statement adds a columns starting with 'param…' by partitioning the columns of the PersonalDetailsData that has alias PD.

Now the next set of highlighted statement select the DataValue for each DataName written into PersonalDetailsData table.

All this is shown in below picture. The first result is the raw data of PersonaDetailsData table and the 2nd result set is converted data.

The second result set is easy to understand and present to the end user.

SQLQuery7.sql - SH...C\Shreeharsh (53))*

```
/****** Script for SelectTopNRows command from SSMS  ******
SELECT TOP 1000 [AutoId]
      ,[PersonalDetailsId]
      ,[DataName]
      ,[DataValue]
  FROM [SqlHowTo].[dbo].[PersonalDetailsData]
-- ****************
  SELECT Height, [Weight], Eyes, PersonalDetailsId
    FROM
        (
            SELECT DataName, DataValue, PersonalDetailsId,
            'param'+ cast(ROW_NUMBER() OVER
            (
                PARTITION BY PersonalDetailsId, DataName,
                ORDER BY DataName) as varchar(50)) PD
            FROM
            PersonalDetailsData
        ) PDD
    PIVOT
    (
        max(DataValue)
        FOR
        DataName
        IN
        (Height, [Weight], Eyes)
    ) piv
```

100 %

Results | Messages

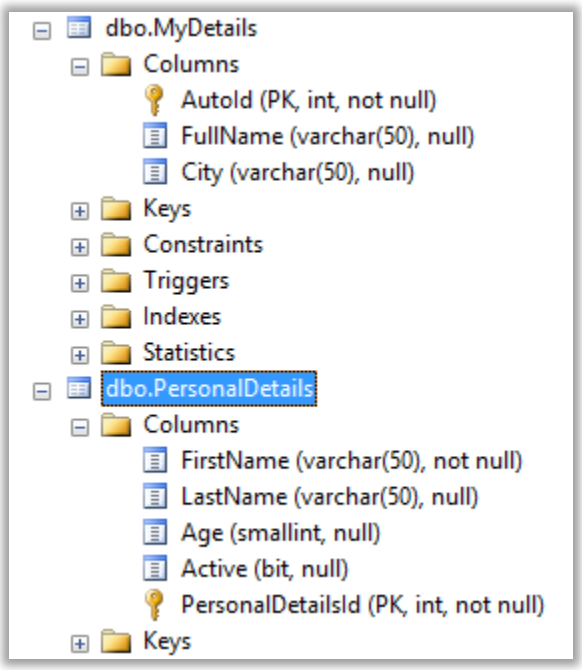
	AutoId	PersonalDetailsId	DataName	DataValue
1	1	1	Height	5 feet 10 inches
2	2	1	Weight	59 kg
3	3	1	Eyes	Black
4	4	2	Height	5 feet 5 inches
5	5	2	Weight	50 kg
6	6	2	Eyes	Brown
7	7	3	Height	6 feet

	Height	Weight	Eyes	PersonalDetailsId
1	5 feet 10 inches	59 kg	Black	1
2	5 feet 5 inches	50 kg	Brown	2
3	6 feet	NULL	NULL	3

58. How to transfer data from one table to another in SQL Server?

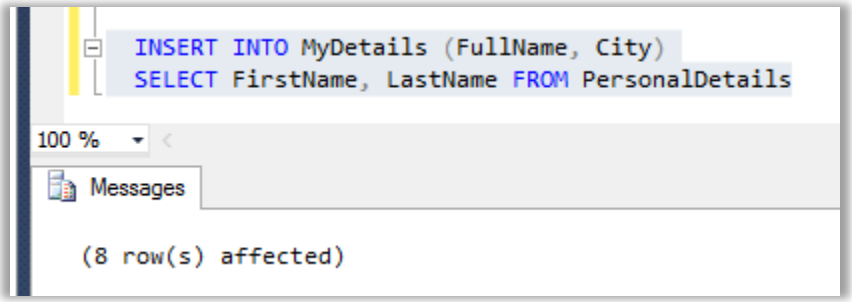
Transferring data from one table to another table, we can use INSERT INTO statement provided the data type of both table columns are same.

In this example, we have following data structure of both tables and we shall try to insert FirstName, LastName column data of PersonalDetails into FullName and City of MyAccounts (Do not worry, the data is not making sense, however our demonstration purpose will be solved ☺).



Execute following query in the Query window

```
INSERT INTO MyDetails (FullName, City)
SELECT FirstName, LastName FROM PersonalDetails
```



As the data type of both columns from both database tables are same so it will select FirstName and LastName from PersonalDetails table and insert into MyDetails table.

For conditional insertion, we can put WHERE clause in the SELECT statement above.

```
INSERT INTO MyDetails (FullName, City)
SELECT FirstName, LastName FROM PersonalDetails
WHERE FirstName LIKE 'S%'
```

Above query will select only those records from the PersonalDetails table whose FirstName starts with “S” and insert into MyDetails database table.

59. How to delete duplicate rows from the database table in SQL Server?

To delete duplicate rows from SQL Server database tables, we can use below SQL Statements. In this case we are creating a CTE (Common Table Expression - explained in following points) with ROW_NUMBER based on the value of FullName and City from MyDetails table. The duplicate of FullName and City count will be stored into RecordCount column (dynamic column). After that we are deleting the record from the CTE whose RecordCouint is greater than 1 (ie. Duplicate rows found).

```
WITH PersonalDetailsCTE AS
      (
            SELECT ROW_NUMBER() OVER (PARTITION BY FullName, City ORDER BY
AutoId) AS RecordCount FROM MyDetails
      )
   DELETE FROM PersonalDetailsCTE WHERE RecordCount > 1
```

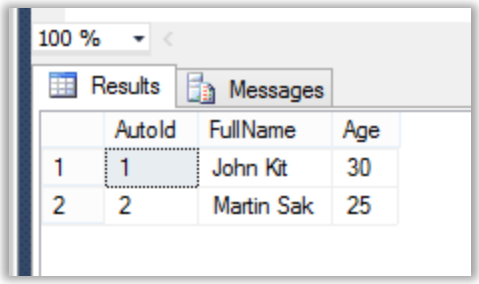
Running the above code blocks shows the number rows affected, ie. Deleted.

60. How to write query to join tables form two different databases in SQL Server?

To join tables from two different databases, we can use fully qualified names of the tables in this format [databasename].[owner].[tablename]

```
SELECT ac.AutoId, ac.FullName, pd.Age
FROM [IHaveBeenRenamed].[dbo].Accounts ac
INNER JOIN
[SqlHowTo].[dbo].PersonalDetails pd ON ac.AutoId = pd.PersonalDetailsId
```

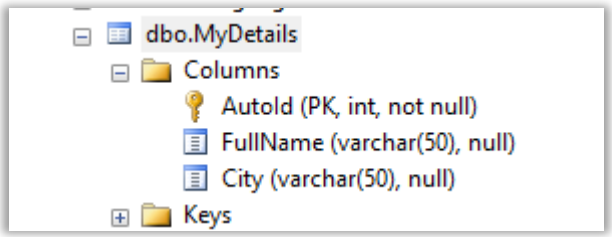
In the above query, we are joining Accounts table of IHaveBeenRenamed database table and PersonalDetails of SqlHowTo table on AutoId of Accounts table and PersonalDetailsId of PersonalDetails table.



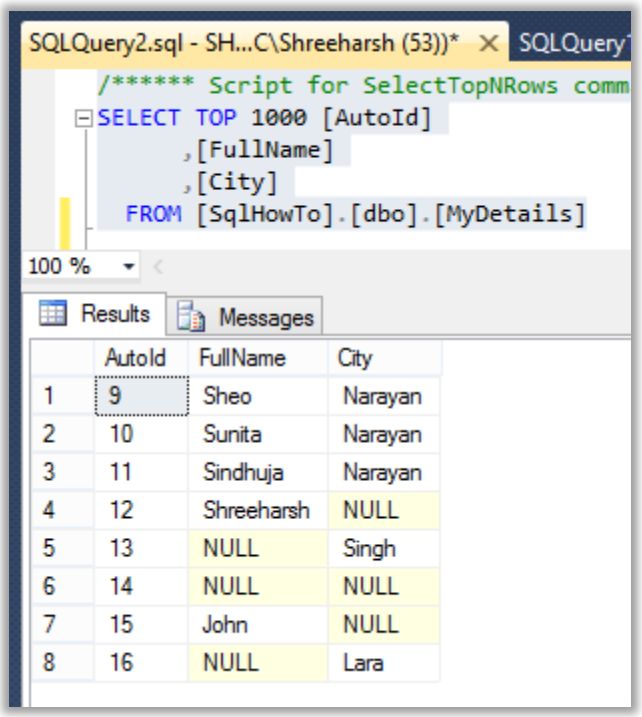
To know how to join two databases that are physically on different servers, read this post of DotNetFunda.com - http://www.dotnetfunda.com/forums/show/3006/linking-2-servers-and-copying-table-data-from-server1-to-server2

61. How to get a default value for a nullable fields in SQL Server?

To demonstrate how to get a default value for a nullable fields in SQL Server, we have taken below MyDetails table as an example.



Above table has following records

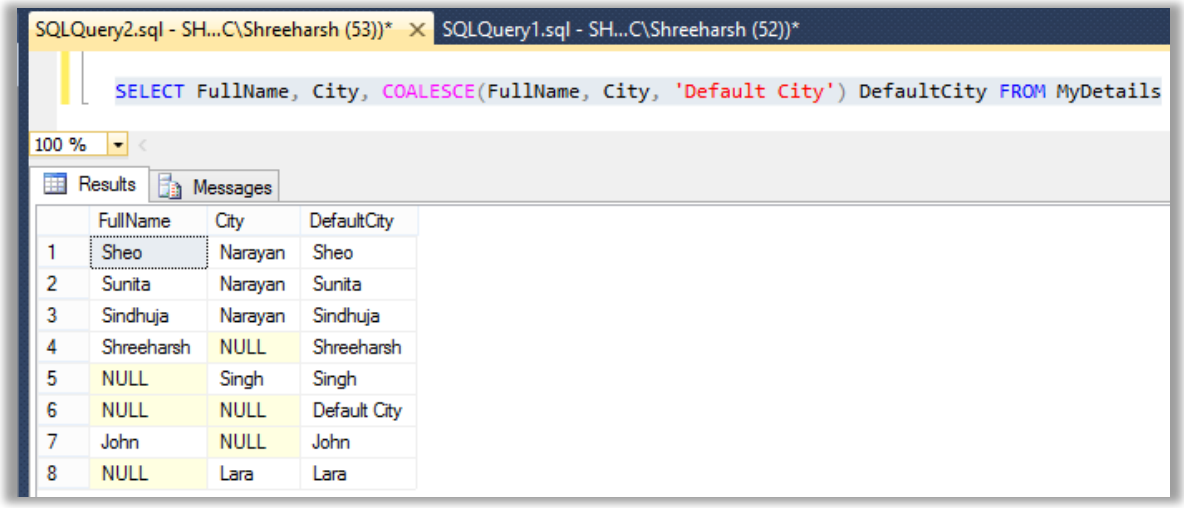


	AutoId	FullName	City
1	9	Sheo	Narayan
2	10	Sunita	Narayan
3	11	Sindhuja	Narayan
4	12	Shreeharsh	NULL
5	13	NULL	Singh
6	14	NULL	NULL
7	15	John	NULL
8	16	NULL	Lara

Now, let's assume that we have to get not null value of either FullName, City or if both columns have null value then a Default value then we can use COALESCE function.

```
SELECT FullName, City, COALESCE(FullName, City, 'Default City') DefaultCity FROM
MyDetails
```

In the above code snippet, notice the COALESCE function. The first parameter is FullName, second parameter is City and third parameter is the hard coded value "Default City".



	FullName	City	DefaultCity
1	Sheo	Narayan	Sheo
2	Sunita	Narayan	Sunita
3	Sindhuja	Narayan	Sindhuja
4	Shreeharsh	NULL	Shreeharsh
5	NULL	Singh	Singh
6	NULL	NULL	Default City
7	John	NULL	John
8	NULL	Lara	Lara

In above case, non null value of FullName and City will be taken as the value of [Default City] column. In case both FullName and City is null then “Default City” value is set as the value of [Default City] column.

Built in functions

There are many built in functions available in SQL Server that is used to manipulate date, time, numbers, string etc.

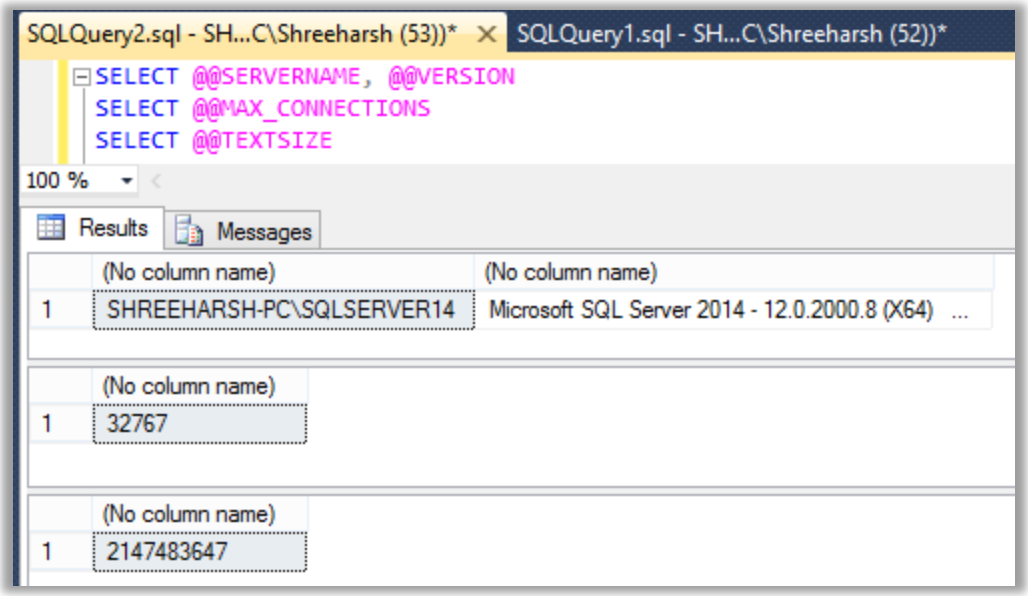
62. How to know the installed SQL Server name and version?

To get server name @@SERVERNAME and to get the version of SQL Server installed @@VERSION function can be used.

```
SELECT @@SERVERNAME, @@VERSION
```

Similarly, we can also use MAX connection allowed and max text size function like below

```
SELECT @@MAX_CONNECTIONS
SELECT @@TEXTSIZE
```



Aggregate functions

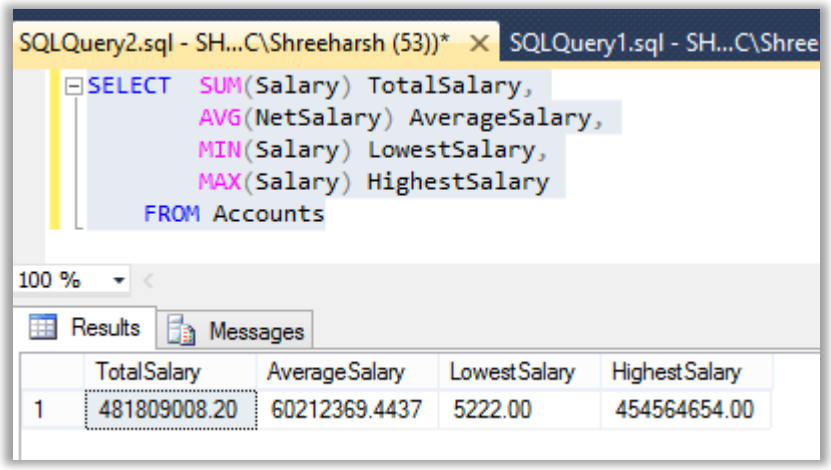
Aggregate functions are functions that work on a particular column of the rows and returns a single value.

63. How to user aggregate functions like AVG, MIN, MAX, SUM etc in SQL Server?

As AVG, MIN, MAX, and SUM all these functions work on numeric data so we need to pass numeric data type of columns to these functions that returns a single value.

```
SELECT  SUM(Salary) TotalSalary,
              AVG(NetSalary) AverageSalary,
              MIN(Salary) LowestSalary,
              MAX(Salary) HighestSalary
       FROM Accounts
```

In above query, we are getting the sum of Salary, average of NetSalary, the lowest of the salary and highest of the salary.



We can also filter the data and get these values based on the filtered records.

```
SELECT  SUM(Salary) TotalSalary,
              AVG(NetSalary) AverageSalary,
              MIN(Salary) LowestSalary,
              MAX(Salary) HighestSalary
       FROM Accounts
WHERE AutoId > 3
```

This returns sum, average, min and max of only those records whose AutoId value is greater than 3.

String related

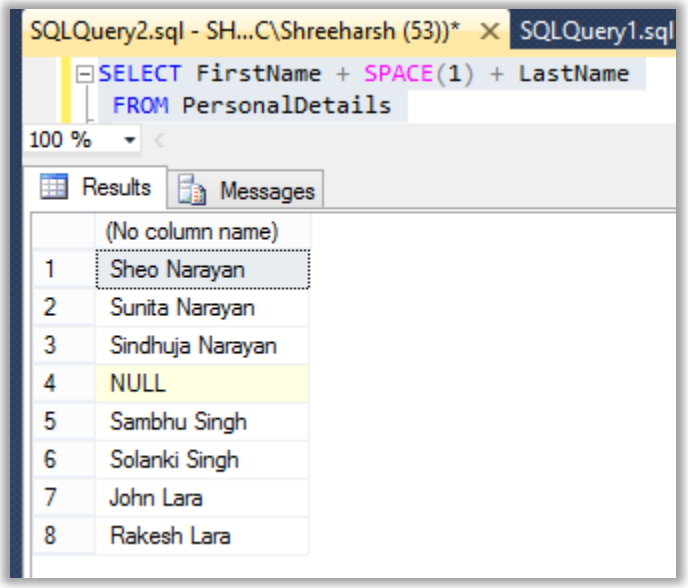
String related functions work on string data type columns.

64. How to concatenate two string types columns separated by a string in SQL Server?

To concatenate two string type columns separated by space, we can use space function.

```
SELECT FirstName + SPACE(1) + LastName
 FROM PersonalDetails
```

Notice the SPACE(1) function in between FirstName and LastName. As the parameter value passed in SPACE function is 1 so there will be one blank space in between FirstName and LastName column values.

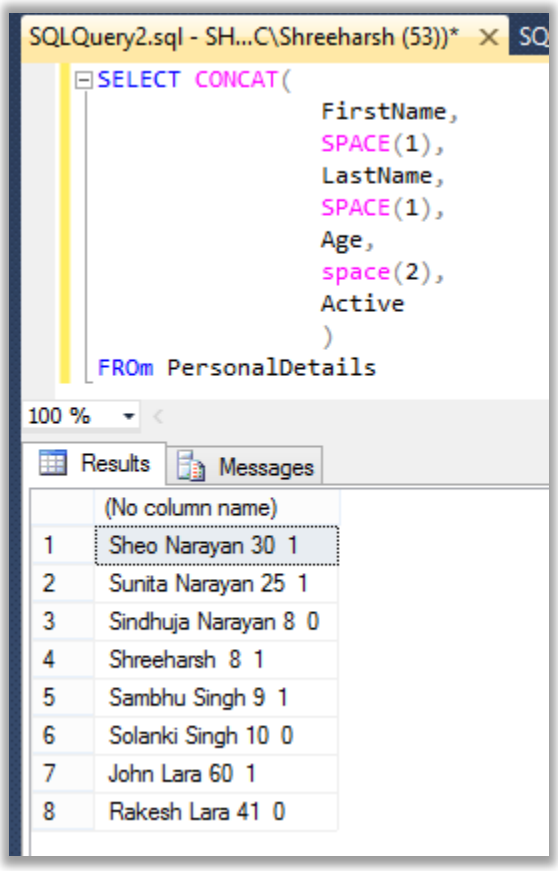


65. How to concatenate more than one columns in SQL Server?

To concatenate more than one columns into a single value, we can user CONCAT function like below. The good thing is that even if columns are of different types, it can be concatenated. Like we can concatenate string with integer, datetime etc.

```
SELECT CONCAT(
                        FirstName,
                        SPACE(1),
                        LastName,
                        SPACE(1),
                        Age,
                        space(2),
                        Active
                        )
FROm PersonalDetails
```

Here, we are concatenating FirstName a space separated by LastName one space separated by Age and two space separated by Active field from the PersonalDetails table.



Date Time related

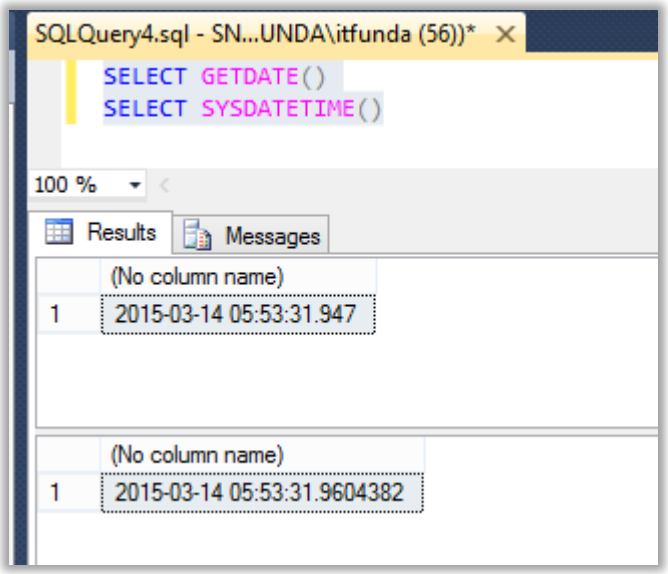
DateTime related functions are used to work with date and times data types.

66. How to get current Date and time in SQL Server?

To get current date and time in SQL Server, we can use GETDATE() or SYSDATETIME() functions.

```
SELECT GETDATE()
SELECT SYSDATETIME()
```

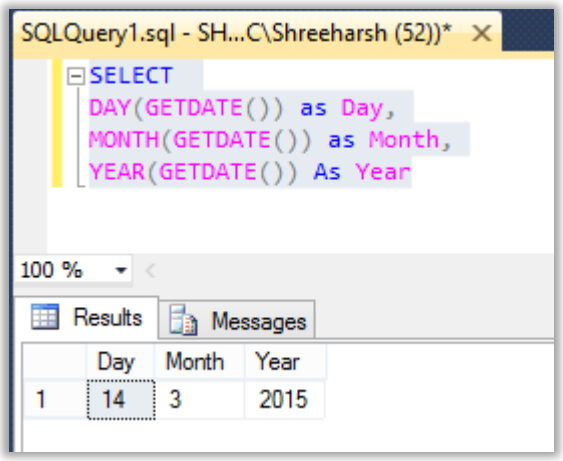
The first GETDATE() function gives the current date and time, the second SYSDATETIME gives more precise value.



67. How to return day, month, and year in the SQL Server database?

To return day, month and year of the date we can use DAY, MONTH and YEAR function by passing the current date. Similarly, to get day, month and year of other date simply pass other date as parameter to these functions.

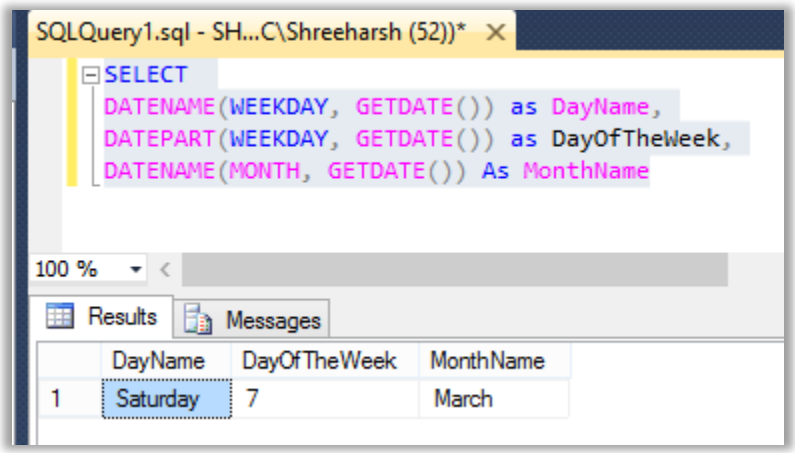
```
SELECT
DAY(GETDATE()) as Day,
MONTH(GETDATE()) as Month,
YEAR(GETDATE()) As Year
```



68. How to return Day name of the week, day of the week, and month name in SQL Server?

We can use DATENAME, DATEPART functions by passing respective parameters to return the day name of the week, day of the week and month names.

```
SELECT
DATENAME(WEEKDAY, GETDATE()) as DayName,
DATEPART(WEEKDAY, GETDATE()) as DayOfTheWeek,
DATENAME(MONTH, GETDATE()) As MonthName
```



Following are valid parameters value that can be passed to DATEPART function.

DATEPART(first parameter value)	DATEPART(first parameter value abbrevidations)
Year	yy, yyyy
Quarter	q or qq
Month	mm, m
Week	wk, ww
Day	D, dd
Weekday	dw
Dayofyear	Dy, y
Hour	Hh
Minute	Mi, n
Second	Ss, s

Notice that the same above result can be retrieved by passing abbreviations instead of full datepart parameter value.

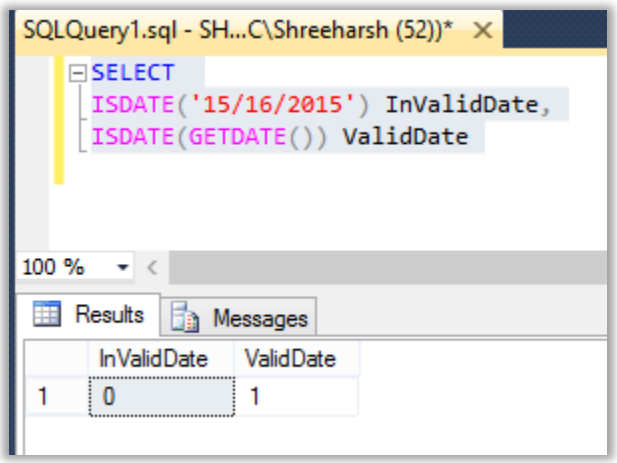
```
SELECT
DATENAME(dw, GETDATE()) as DayName,
DATEPART(dw, GETDATE()) as DayOfTheWeek,
DATENAME(m, GETDATE()) As MonthName
```

Above query will give the same result as the previous one.

69. How to check for validate date in SQL Server?

To check for a valid date, we can use IsValid function by passing the date string or a date value.

```
SELECT
ISDATE('15/16/2015') InValidDate,
ISDATE(GETDATE()) ValidDate
```

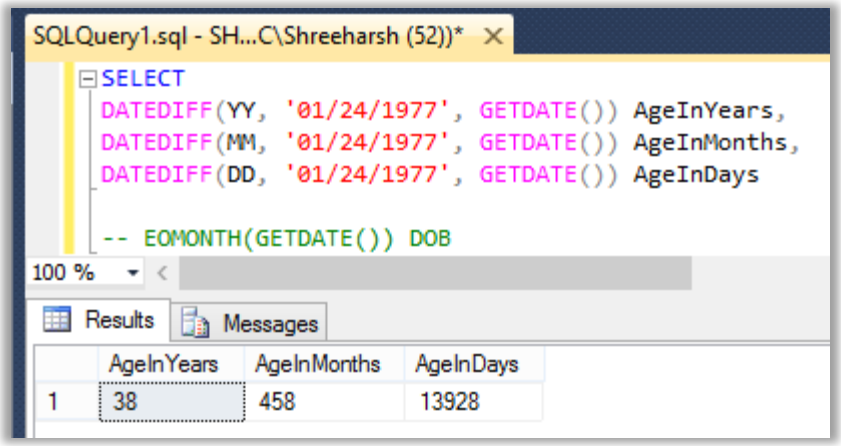


70. How to get the difference between two dates in SQL Server?

To get the difference between two dates, we can use DATEDIFF function.

```
SELECT
DATEDIFF(YY, '01/24/1977', GETDATE()) AgeInYears,
DATEDIFF(MM, '01/24/1977', GETDATE()) AgeInMonths,
DATEDIFF(DD, '01/24/1977', GETDATE()) AgeInDays
```

Passing first parameter as different interval values, we can get year, month and days. Here, the SQL query will return difference of two dates in year, month an days respectively.



71. How to get Age in year, month and date format in SQL Server?

To get age in Year, month and date, we need to manipulate the given date with DATEDIFF and other helping functions.

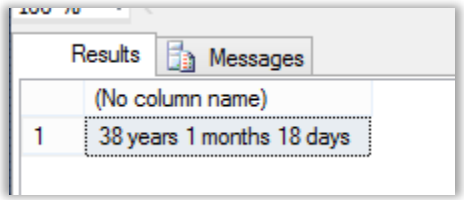
```
DECLARE @givenDate datetime
DECLARE @tempDate datetime
DECLARE @years int, @months int, @days int

SELECT @givenDate = '01/24/1977'

SELECT @tempDate = @givenDate

-- get year
SELECT @years = DATEDIFF(yy, @tempDate, GETDATE()) -
              CASE
                     WHEN
                            (MONTH(@givenDate) > MONTH(GETDATE()))
                                   OR
                            (MONTH(@givenDate) = MONTH(GETDATE())
                                          AND
                                          DAY(@givenDate) > DAY(GETDATE()))
                     THEN 1
                     ELSE 0
              END

SELECT @tempDate = DATEADD(yy, @years, @tempDate)

-- get months
SELECT @months = DATEDIFF(m, @tempDate, GETDATE()) -
              CASE
                     WHEN
                            DAY(@givenDate) > DAY(GETDATE())
                     THEN 1
                     ELSE 0
              END
```

```
SELECT @tempDate = DATEADD(m, @months, @tempDate)

-- get days
SELECT @days = DATEDIFF(d, @tempDate, GETDATE())

-- output the result
SELECT  CAST(@years as varchar(5)) + ' years ' +
              CAST(@months as varchar(3)) + ' months ' +
              CAST(@days as varchar(3)) + ' days'
```

In the above code snippet, we are declaring few variables to hold given date, temporary date to calculate on and calculated date, months and year.

First, we are calculating the year portion of the age by subtracting the year from 1 or 0 based on whether the current month is greater than given month or not.

Next, we are calculating months and the days and at last we are casting years, months and days values into varchar and giving as output.



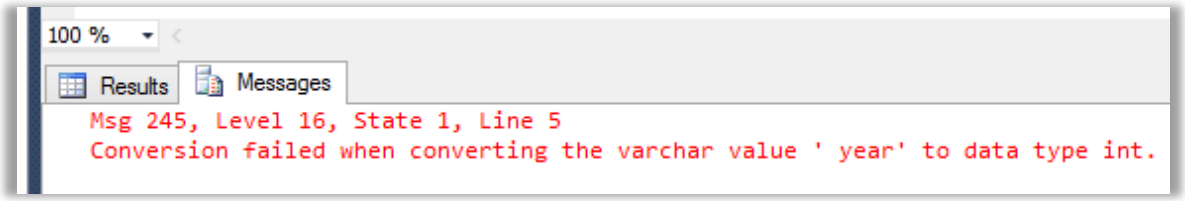
Conversion related

72. How to convert the data type of the SQL Server variables?

To convert the data types of a variable, we can use CAST function. CAST function is only used to change the data type of the value.

```
DECLARE @year int

SELECT @year = 1975

SELECT @year + ' year'
```

In the above code snippet, we are declaring @year variable of int type. If we want to append a string to the variable in the output, it throws error.



This error comes because @year is of integer type and we are trying add string into it. Remember that integer can be only added to ant numeric type not string. If we try to add two string, it gets concatenated.

Replacing the last line to below code will work

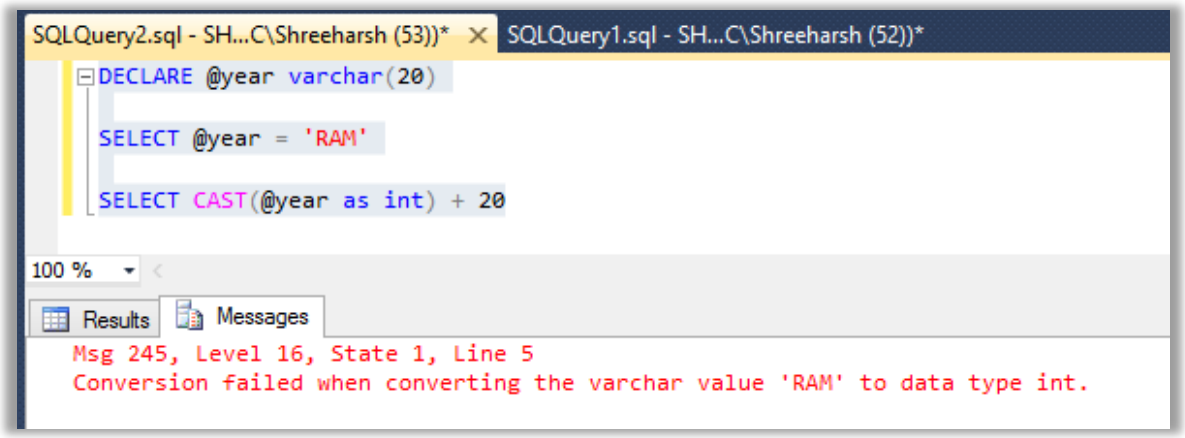
```
SELECT CAST(@year as varchar(5)) + ' year'
```

Here, we are converting the @year variable to varchar type and then we are trying to add ' year' string that will work as both are of string type and it will easily get concatenated.

Instead of varchar, we can use other data types like decimal, float, single etc depending on whether it is convertible or not. The thumb rule is any numeric data types can be converted into string type. However a string may or may not get converted into numeric data types.

```
DECLARE @year varchar(20)

SELECT @year = '1975'

SELECT CAST(@year as int) + 20
```

In above case, we are converting string to integer and adding 20, so the result would be 1995. Here, the value of @year is '1975' string that is a perfect valid string so it can be converted into integer. However, if we change the value of @year to some other alphabets, it throws error.



73. How to convert a column value from one data type to another in SQL Server?

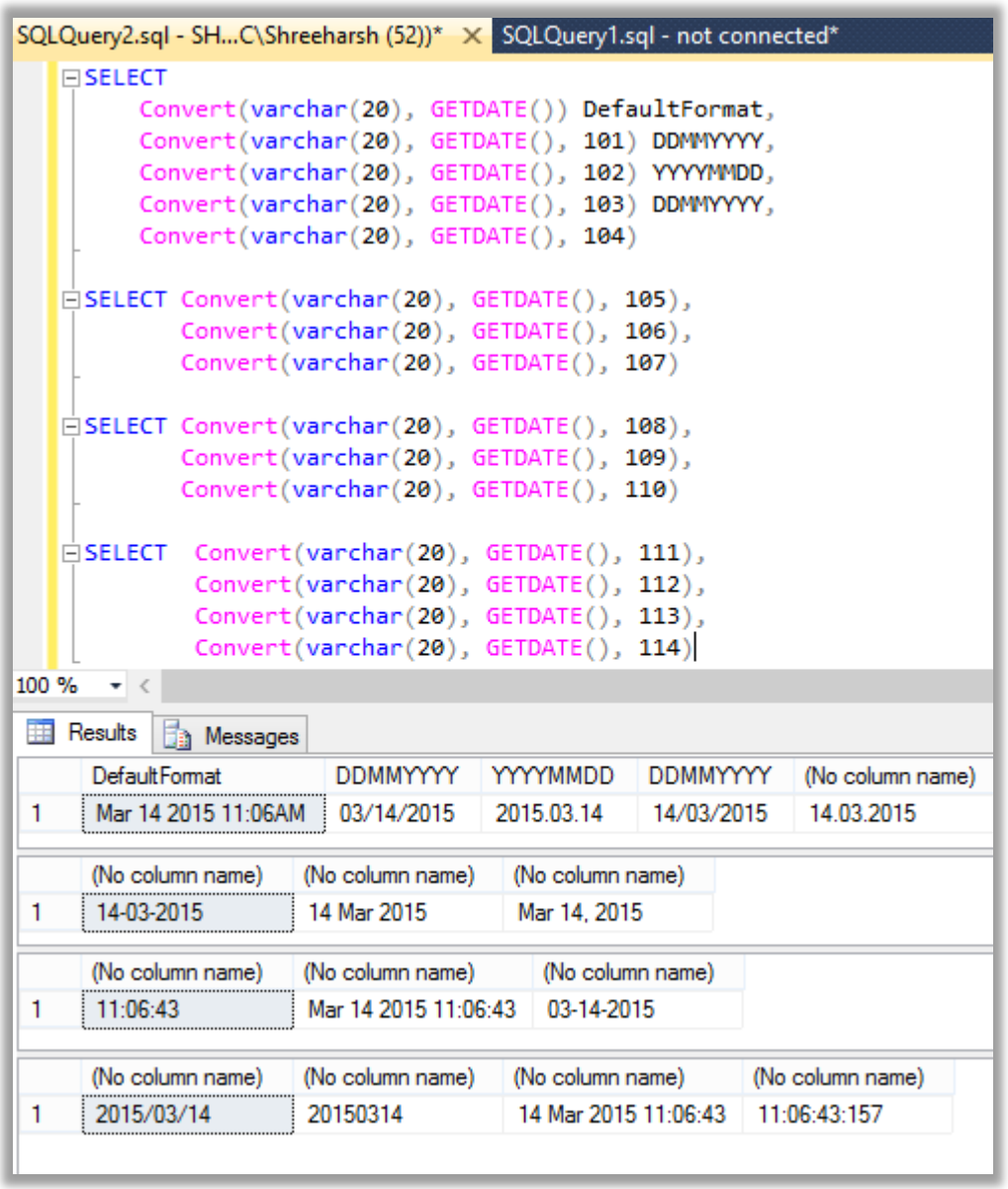
To convert a column value from one data type to another, we use CONVERT function. This accepts three parameters

i. The target data type to convert to
ii. The value to convert to
iii. The style number to convert to (optional)

The main difference between CAST and CONVERT is that CONVERT after converting the data type, also apply the formatting number based on value we pass.

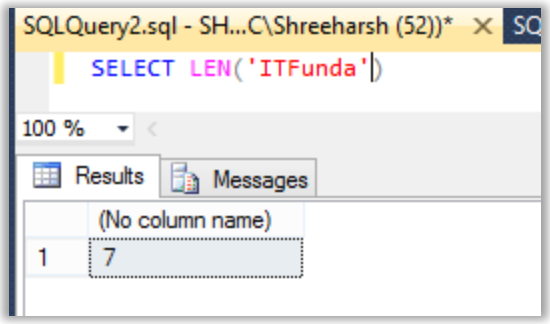
```
SELECT
       Convert(varchar(20), GETDATE()) DefaultFormat,
       Convert(varchar(20), GETDATE(), 101) DDMMYYYY,
       Convert(varchar(20), GETDATE(), 102) YYYYMMDD,
       Convert(varchar(20), GETDATE(), 103) DDMMYYYY,
       Convert(varchar(20), GETDATE(), 104)

SELECT Convert(varchar(20), GETDATE(), 105),
          Convert(varchar(20), GETDATE(), 106),
          Convert(varchar(20), GETDATE(), 107)

SELECT Convert(varchar(20), GETDATE(), 108),
          Convert(varchar(20), GETDATE(), 109),
          Convert(varchar(20), GETDATE(), 110)

SELECT  Convert(varchar(20), GETDATE(), 111),
              Convert(varchar(20), GETDATE(), 112),
              Convert(varchar(20), GETDATE(), 113),
              Convert(varchar(20), GETDATE(), 114)
```

Notice the 3rd parameter of each CONVERT function and notice the result below.



74. How to get the length of the string in SQL Server?

To get the length of the string in SQL Server, we user LEN function by passing the string as parameter.



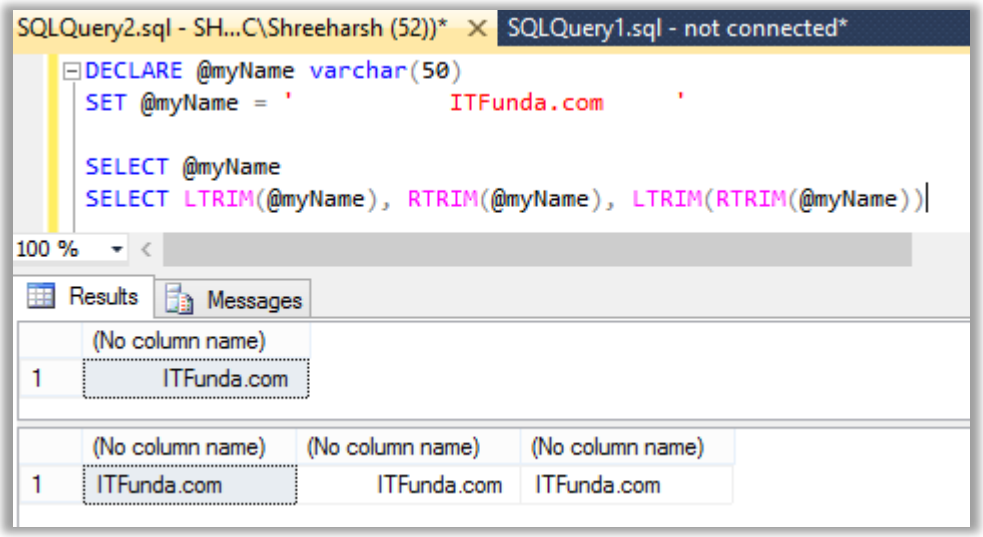
75. How to trim unnecessary space from the string in SQL Server?

To trim unnecessary blank spaces from the string in SQL Server, we use LTRIM or RTRIM functions.

i. LTRIM – removes the left side spaces, if any
ii. RTRIM – removes the right side spaces, if any
    a. To remove all spaces, we can use both LTRIM and RTRIM functions.

```
DECLARE @myName varchar(50)
SET @myName = '           ITFunda.com     '

SELECT @myName
SELECT LTRIM(@myName), RTRIM(@myName), LTRIM(RTRIM(@myName))
```

Notice the output of above code snippet below.



In first case, the output is coming along with left side and right side blank space.

In the 2nd result set

i. The first column value is after removing all left side spaces from the string
ii. The second column value is after removing all right side spaces from the string
iii. The third column removes spaces from right side and then left side and gives the actual string without any blank space either side.

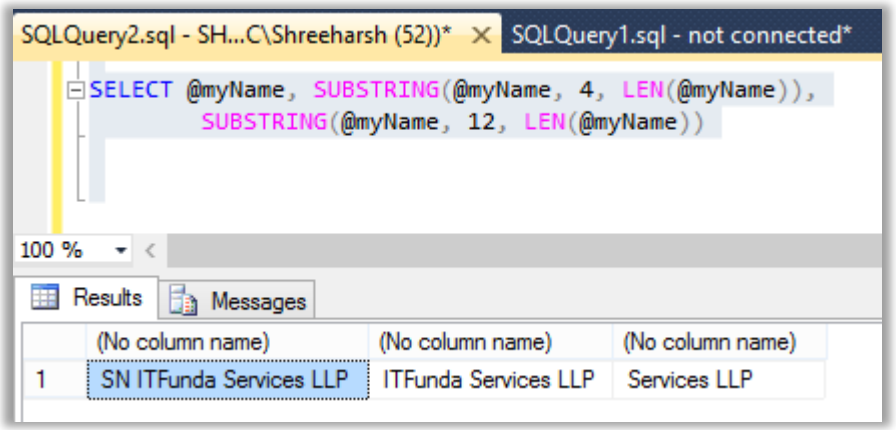
76. How to get a part of string from a sentence in SQL Server?

To get part of string from a sentence, we can use SUBSTRING function. It accepts three parameter

i. First parameter is the string to get substring from
ii. Second parameter is the length of characters from where to start getting the string
iii. Third parameter is the total length of characters we need to get starting from 2nd parameter value.

```
DECLARE @myName varchar(50)
SET @myName = 'SN ITFunda Services LLP'

SELECT @myName, SUBSTRING(@myName, 4, LEN(@myName)),
                SUBSTRING(@myName, 12, LEN(@myName))
```

The first column will give us the complete string, the second will give us all the string starting from 4th position, the third column will give us string starting from 12th position in the whole string.



User defined functions (UDFs)

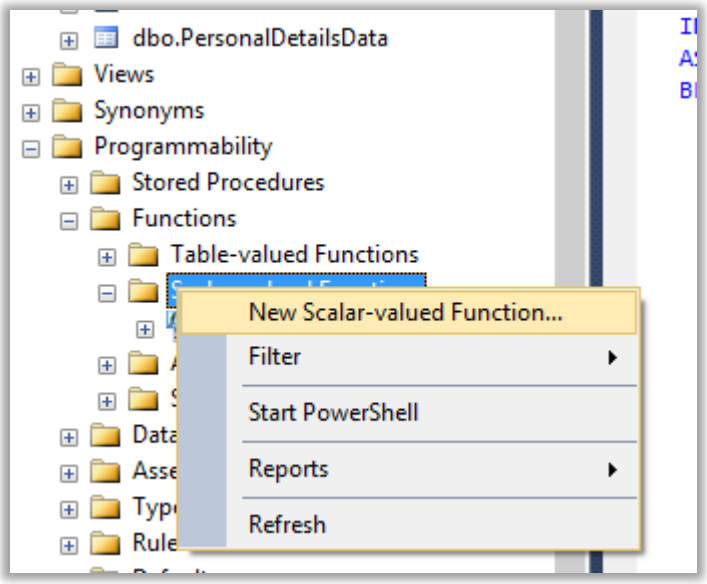
SQL Server user-defined functions are set of SQL statements that accept parameters, perform an action, and return the result. The return value can either be a single scalar value or a result set.

There are two types of user defined functions in SQL Server

i. Scalar function – the user defined function that returns a single value
ii. Table-Valued function – the user defined function that returns tabular data

77. How to create a User defined scalar function in SQL Server?

To create a scalar function, right click the Scalar-valued Functions option from Programmability > Functions.



This opens up a function template in the query window. Now change placeholders for name, data type, variables and return variable etc.

```
CRETAE FUNCTION [dbo].[GetFullName]
(
	-- Add the parameters for the function here
	@PersonalDetailsId int
)
RETURNS varchar(50)
AS
BEGIN
	-- Declare the return variable here
	DECLARE @FullName varchar(50)

	-- Add the T-SQL statements to compute the return value here
	SELECT @FullName = FirstName + ' ' + LastName FROM PersonalDetails WHERE
PersonalDetailsId = @PersonalDetailsId

	-- Return the result of the function
	RETURN @FullName

END
```

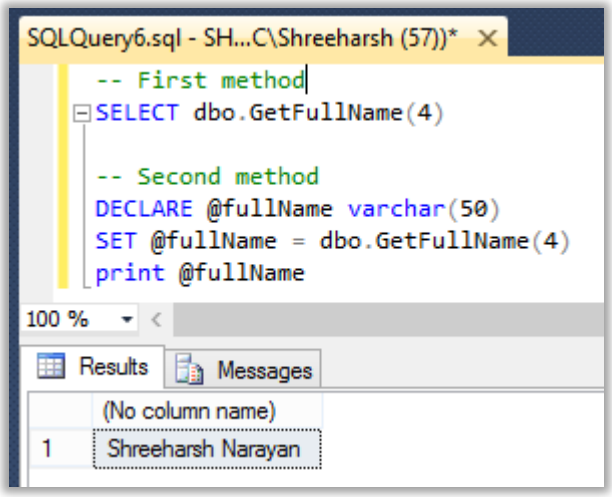
In above case, we are creating a function named “GetFullName” with @PersonalDetailsId as parameter. The function returns a data that is of “varchar(50)”.

Next, we are declaring a @FullName variable and setting its value in the SELECT statement from the PersonalDetails table based on @PersonalDetailsId parameter value passed in.

Then returning the @FullName value.

To call this function, we can use either SELECT statement or set its value into a variable like below

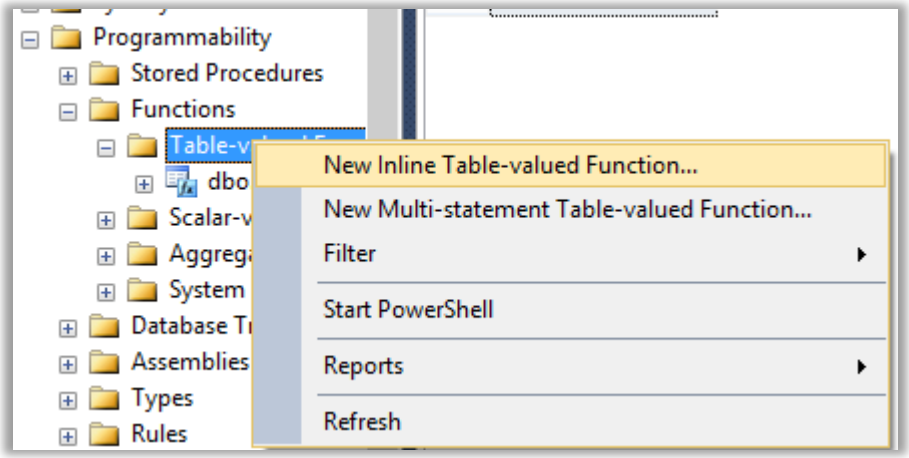
```
-- First method
SELECT dbo.GetFullName(4)

-- Second method
DECLARE @fullName varchar(50)
SET @fullName = dbo.GetFullName(4)
print @fullName
```



The value of @fullName will be printed in the Message tab in the result panel.

78. How to create a Table-valued function in SQL Srever?

As explained above, a table-valued function returns a tabular data. To create this, we can right click the table-valued Functions option under Programmability > Functions and choose “New Inline Table-valued Function….”.



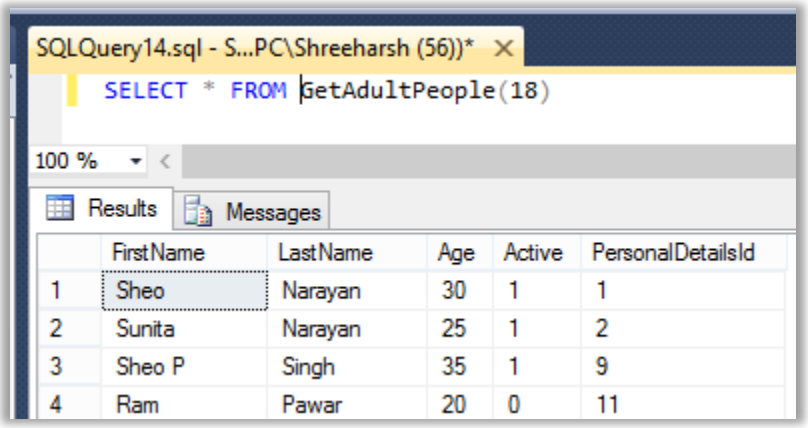
This will create a new query window with table-valued user defined function template. Replace the placeholders for name, data types etc. as per need.

```
CRETAE FUNCTION [dbo].[GetAdultPeople]
(
	@age int
)
RETURNS TABLE
AS
RETURN
(
	-- Add the SELECT statement with parameter references here
	SELECT * FROM PersonalDetails WHERE Age >= @age
)
```

In above case, the function name is GetAdultPeople with @age parameter. It is returning all details from PersonalDetails table whose age is greater than @age parameter value.

To call this function, call as if this is a physical table (as this returns a tabular structure data).

```
SELECT * FROM dbo.GetAdultPeople(18)
```



	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1
2	Sunita	Narayan	25	1	2
3	Sheo P	Singh	35	1	9
4	Ram	Pawar	20	0	11

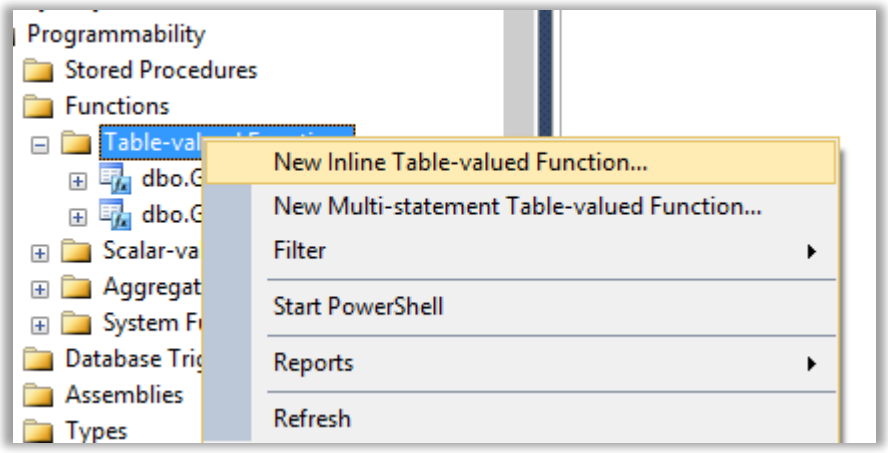
Important:

We can also ignore “dbo” (DataBase Owner) word however it is suggested to use that as prefixed with the function name. If we have created this function by logging in with another user, we should prefix the function name with that username and call it.

Also, conditional (IF ELSE) statement is not permitted in “New Inline Table-valued Function….”, if we want to perform more complex operations in functions in SQL Server, we can use “Multi-statement Table-valued function …”.

79. How to create Multi-statement Table-valued function in SQL Server?

To create multi-statement table-valued function, right click Table-valued Functions from Programmability > Functions



This will open up a template for Multi-statement Table-valued Function, replace different placeholders for name, data type etc.

```
ALTER FUNCTION [dbo].[GetAdultRecords]
(
	@ageType varchar(10)
)
RETURNS @myRecords TABLE
(
	fullName varchar(50) not null,
	salary money
)
AS
BEGIN
	-- Fill the table variable with the rows for your result set

	IF (@ageType = 'minor')
		BEGIN
			INSERT INTO @myRecords
			SELECT pd.FirstName + ' ' + pd.LastName fullName, ac.Salary
			FROM PersonalDetails pd JOIN Accounts ac ON
			pd.PersonalDetailsId = ac.PersonalDetailsId
			WHERE pd.Age < 18
		END
	ELSE
		BEGIN
			INSERT INTO @myRecords
			SELECT pd.FirstName + ' ' + pd.LastName fullName, ac.Salary
			FROM PersonalDetails pd JOIN Accounts ac ON
			pd.PersonalDetailsId = ac.PersonalDetailsId
			WHERE pd.Age >= 18
		END
	RETURN ;
END
```

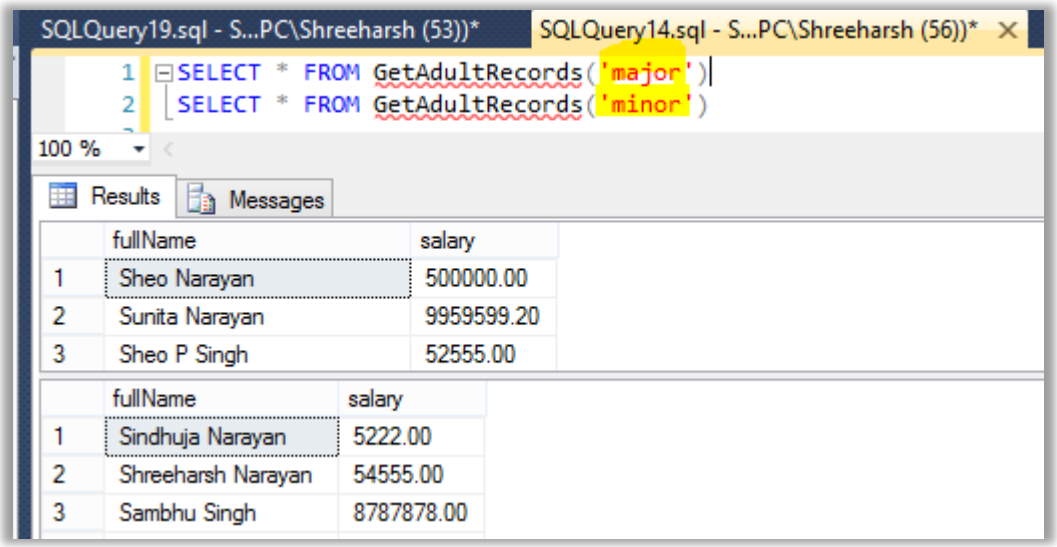
In the above case, we are getting the input parameter of GetAdultRecords function as @ageType. The tabular data (@myRecords) this function returns contains fullName and salary column (virtual table created in this function).

In the following lines of code, based on @ageType value we are getting combination of FirstName and LastName as FullName from PersonalDetails and Salary from Accounts table and inserting into virtual table @myRecords and the same is being returned.

To call this function, use this as a table in the SELECT statement.

```
SELECT * FROM GetAdultRecords('major')
SELECT * FROM GetAdultRecords('minor')
```

This will give desired result in the result window.



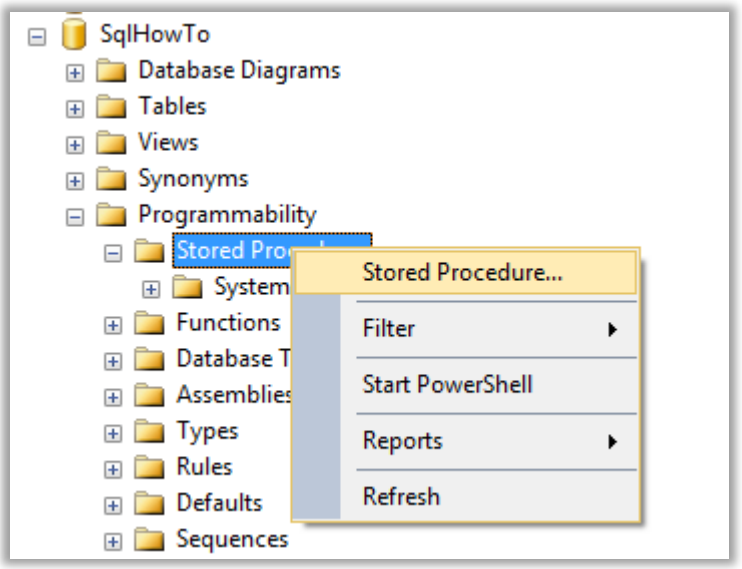
Stored Procedure

A stored procedure is a group of SQL statements compiled as one so that it can be reused again and again.

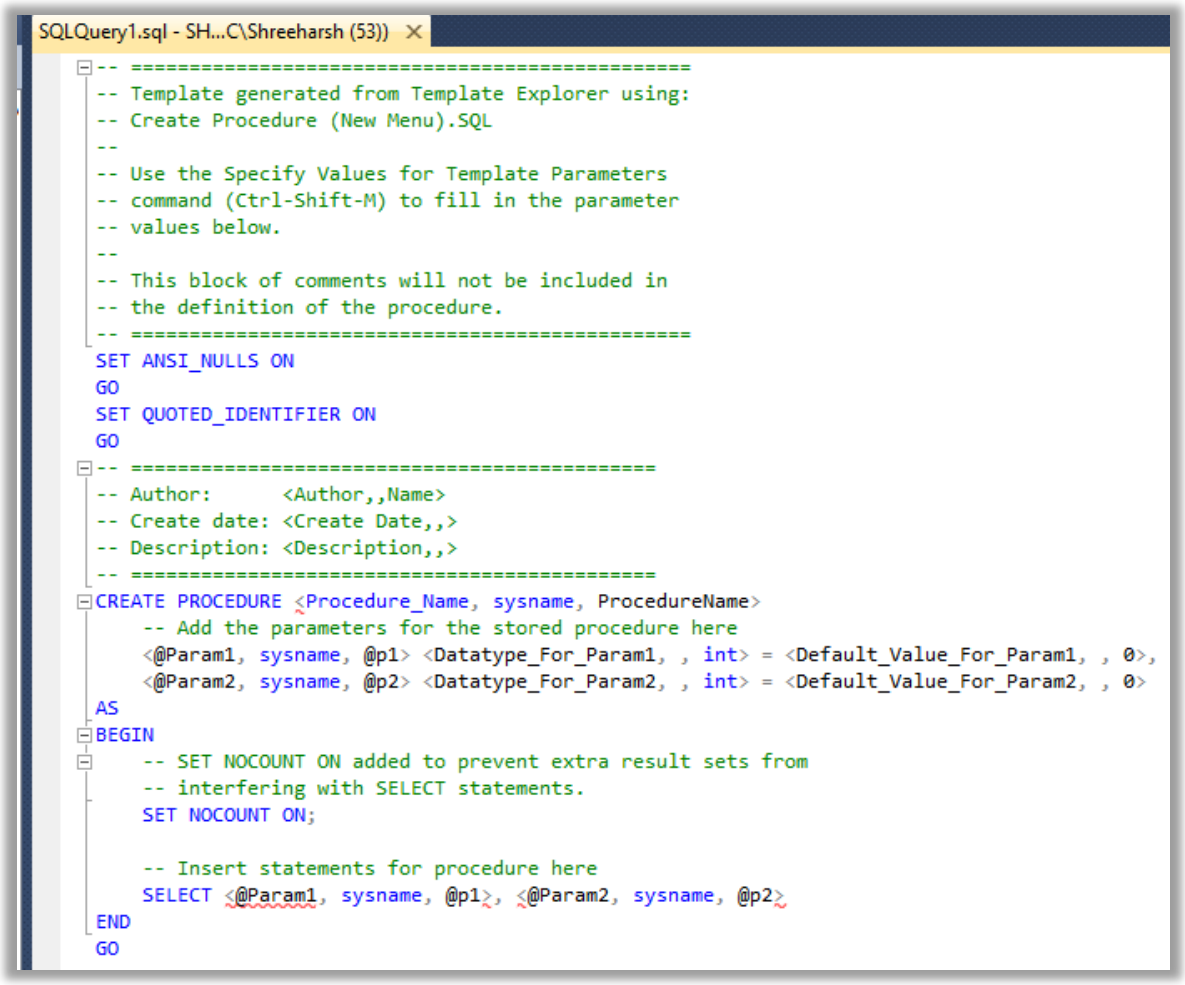
The difference between User defined functions (UDFs) and stored procedure is that, a Function must return a value however, a stored procedure may or may not return a value.

80. How to create stored procedure that accepts parameters to fetch data from database?

To create a stored procedure, expand the Programmability folder of the database and right click the Stored Procedures folder and choose Stored Procedure….



This will open up query window in right side with a template of stored procedure that looks like below



Now modify it as per our own need.

In this case, we have changed the name to LoadPersonalDetails, added a @Age of integer type parameter. Parameter is optional in the stored procedure and more than one parameters can be added separated by comma (,). The parameter always follow by the data types.

Now write the SQL statements based on what should be the purpose of the stored procedure.

In this case, we are going to retrieve the data from PersonalDetails table where Age is greater than the age passed as input parameter.

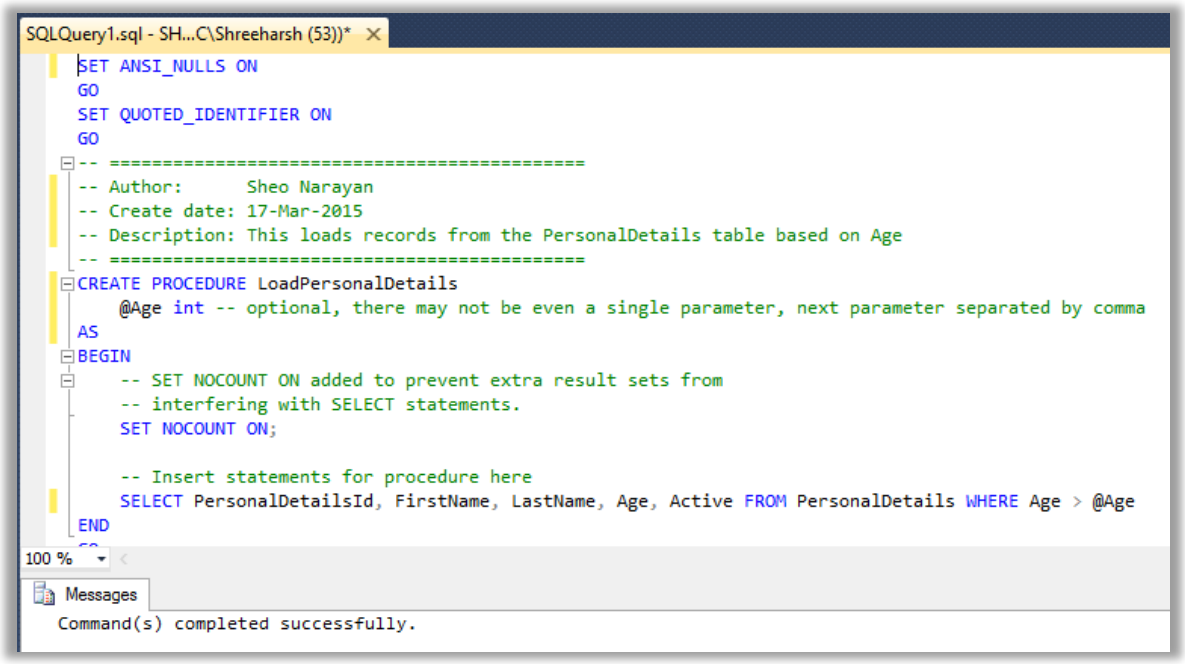
It is always recommended to write the Author, Create date and Description (purpose) of the stored procedure in the comment area at top so that it can be tracked down later on during modification, bug fixes etc. However, it is not mandatory.

My stored procedure code looks like below.

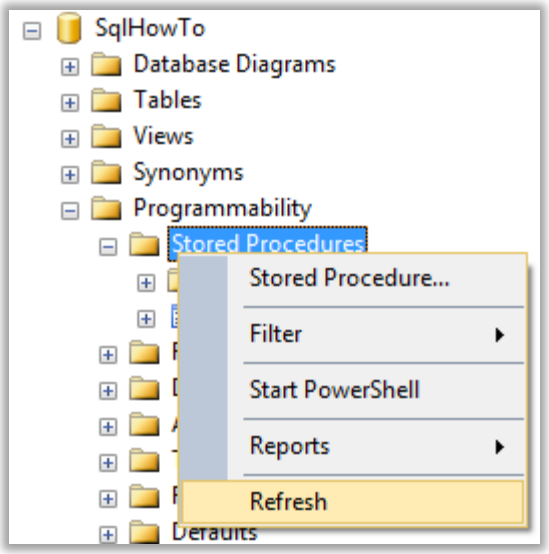
```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
-- =============================================
-- Author:           Sheo Narayan
-- Create date: 17-Mar-2015
-- Description:      This loads records from the PersonalDetails table based on Age
-- =============================================
ALTER PROCEDURE [dbo].[LoadPersonalDetails]
       @Age int -- optional, there may not be even a single parameter, next
parameter separated by comma
AS
BEGIN
       -- SET NOCOUNT ON added to prevent extra result sets from
       -- interfering with SELECT statements.
       SET NOCOUNT ON;

    -- Insert statements for procedure here
       SELECT PersonalDetailsId, FirstName, LastName, Age, Active FROM
PersonalDetails WHERE Age > @Age
END
```

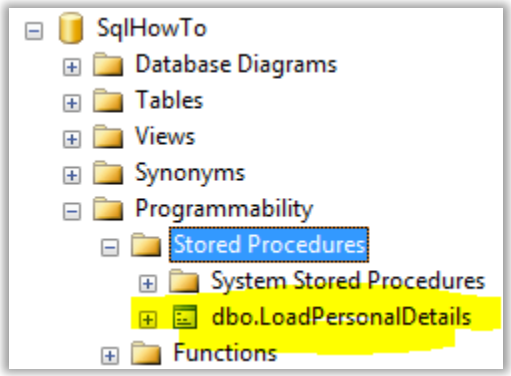
Now remove the template generator code and press F5 or Execute icon from the top – left of the toolbar and it should shows a success message as shown in the below picture.



Now, go back to the Stored Procedures folder and right click and choose Refresh.



That will show the stored procedure we just created.



81. How to create parameter less stored procedure to return data from Sql Server database?

To create a stored procedure, follow the same step as we followed above point, now alter the default stored procedure template like this

```
CREATE PROCEDURE GetAllPersonalDetails

AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

    -- Insert statements for procedure here
	SELECT * FROM PersonalDetails
END
```

Here, we have removed any parameter name, replaced the SELECT statement with selecting all columns and all records (*) from the PersonalDetails database table.

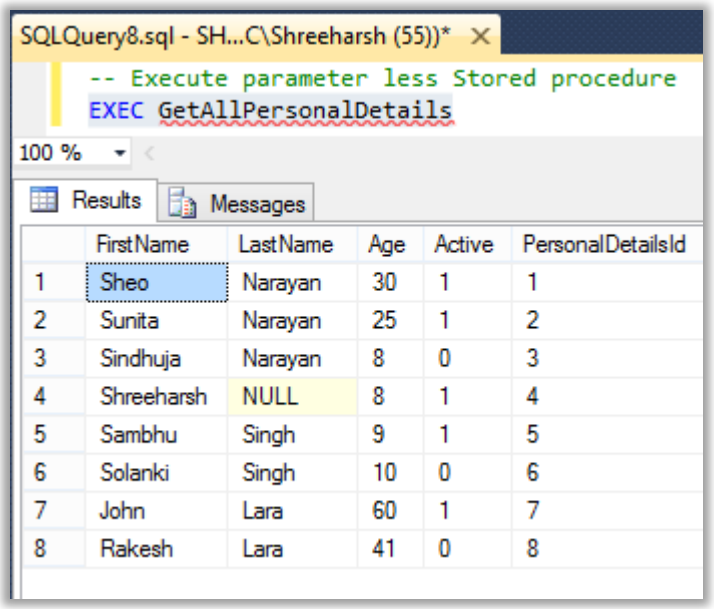
82. How to execute a stored procedure from the query window in SQL Server?

To execute a stored procedure from the query window, we can use EXEC statements.

Executing parameter less stored procedure is very simple, use EXEC with stored procedure name.

```
-- Execute parameter less Stored procedure
EXEC GetAllPersonalDetails
```

Notice that the database dropdown must have the database selected in which the stored procedure exists.

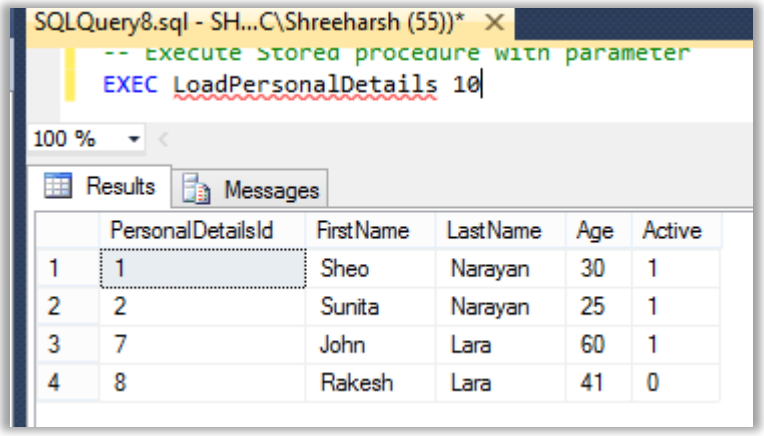


	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1
2	Sunita	Narayan	25	1	2
3	Sindhuja	Narayan	8	0	3
4	Shreeharsh	NULL	8	1	4
5	Sambhu	Singh	9	1	5
6	Solanki	Singh	10	0	6
7	John	Lara	60	1	7
8	Rakesh	Lara	41	0	8

Executing stored procedure with parameter is just passing the parameter value followed by the name of the stored procedure.

```
-- Execute Stored procedure with parameter
EXEC LoadPersonalDetails 10
```

As this stored procedure, created above takes Age as parameter, we have passed its value as 10 that returns all records from the PersonalDetials table having age more than 10. If we have more than one parameter in the stored procedure, we pass those parameters values separated by comma.



	PersonalDetailsId	FirstName	LastName	Age	Active
1	1	Sheo	Narayan	30	1
2	2	Sunita	Narayan	25	1
3	7	John	Lara	60	1
4	8	Rakesh	Lara	41	0

83. How to create a stored procedure to insert record into SQL Server database?

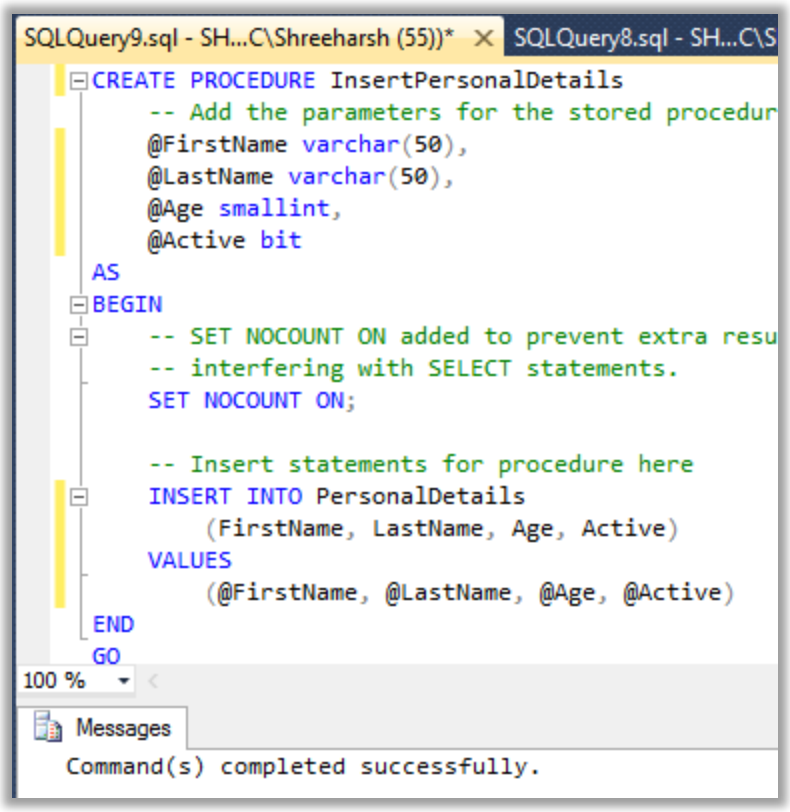
Create a stored procedure and update the default template code with below

```
CREATE PROCEDURE InsertPersonalDetails
	-- Add the parameters for the stored procedure here
	@FirstName varchar(50),
	@LastName varchar(50),
	@Age smallint,
	@Active bit
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

    -- Insert statements for procedure here
	INSERT INTO PersonalDetails
		(FirstName, LastName, Age, Active)
	VALUES
		(@FirstName, @LastName, @Age, @Active)
END
```

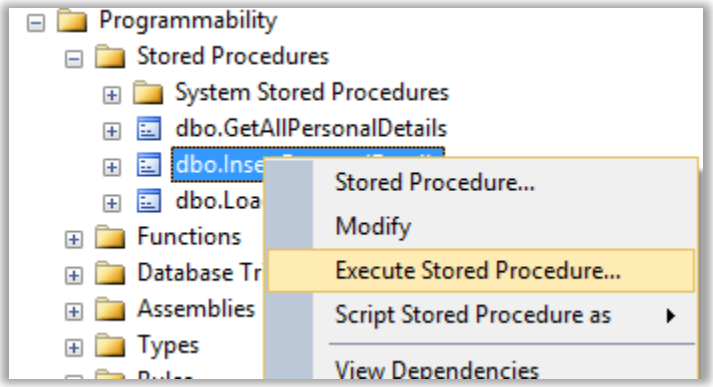
Here, we have four parameters into the stored procedure that is separated by comma. Notice the data type and size of the parameters. It must match with the field type specified in the database table column.

Notice the INSERT into statement. After writing the table name, the column names should be written under bracket and separated by comma and then VALUES and again the input parameters coming in to this stored procedure for respective column names of the table.

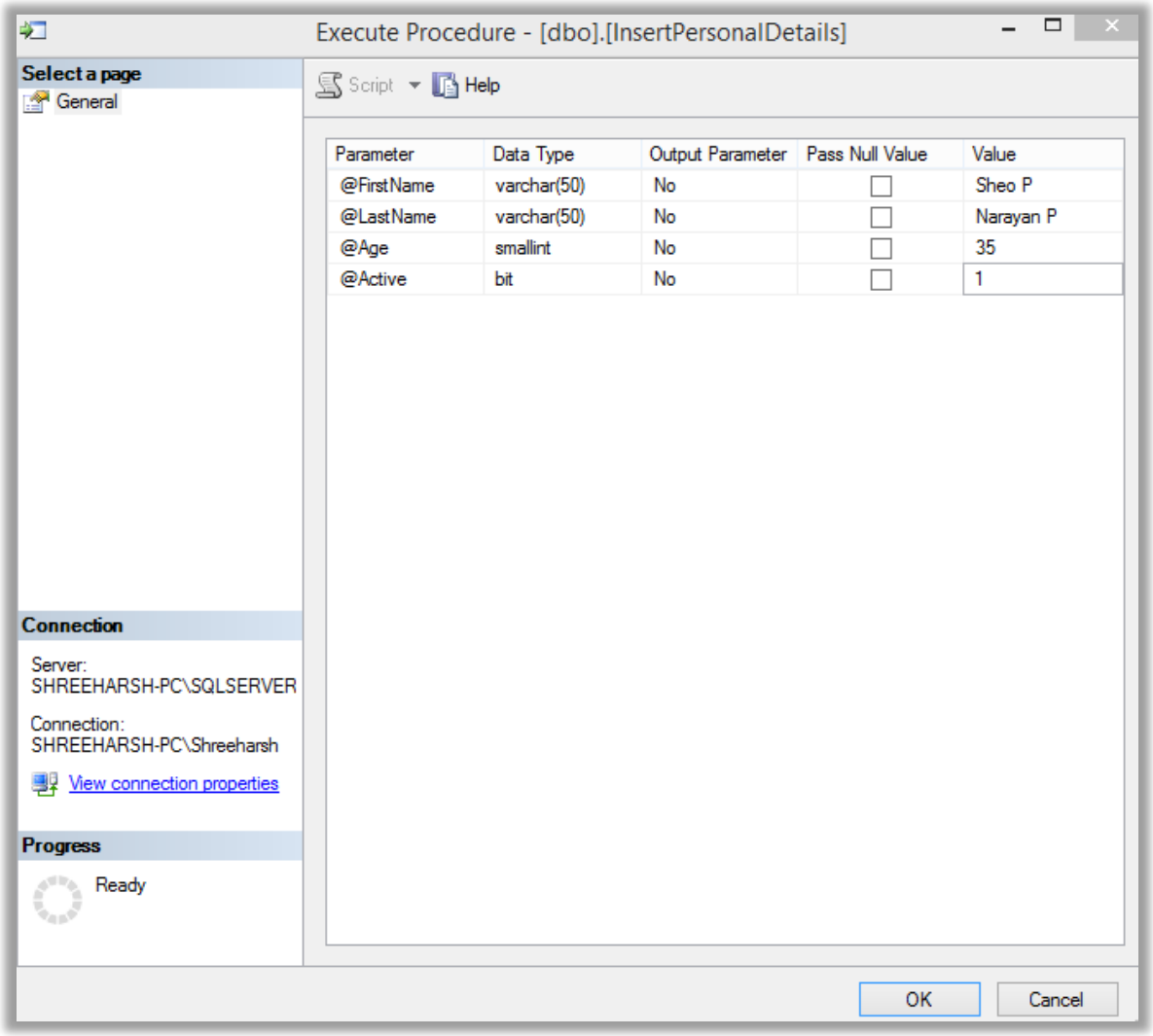
Now executing the above SQL statements will create a new stored procedure in the database.



To execute this stored procedure, we can either use EXEC statement as explained above or right click the stored procedure and choose Execute Stored Procedure… option.

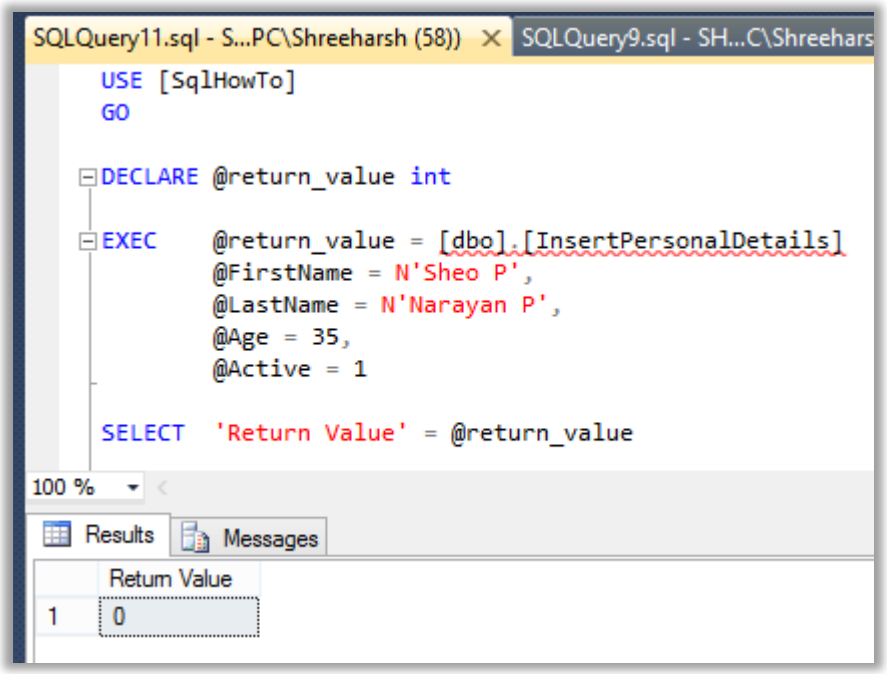


This will bring Execute Procedure dialog box with equal number of rows and value textbox as the stored procedure parameters. We need to write necessary data for the respective parameter value box and click OK button.



Clicking OK button opens up a new query window with same EXEC statement that we have used in previous topics. Notice the EXEC statement and parameters value specified. This dialog box does the same thing that we had done manually by writing EXEC statement.

The Return Value is 0 as this stored procedure is not returning any value.



84. How to create stored procedure to update record in the SQL Server database?

To write Update stored procedure, follow the same procedure, simply write UPDATE sql statement inside the stored procedure.

```
CREATE PROCEDURE UpdatePersonalDetails
	-- Add the parameters for the stored procedure here
	@PersonalDetailsId int,
	@FirstName varchar(50),
	@LastName varchar(50),
	@Age smallint,
	@Active bit
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

    -- Insert statements for procedure here
	UPDATE PersonalDetails SET FirstName = @FirstName, LastName = @LastName,
		Age = @Age, Active = @Active WHERE PersonalDetailsId =
@PersonalDetailsId
END
```

Notice the above stored procedure code, this is almost similar to the Insert stored procedure. We have just added a new parameter named “PersonalDetailsId” that is being used in the WHERE clause of the UPDATE statement.

Calling the stored procedure is same as calling INSERT stored procedure.

85. How to create stored procedure to delete a record from the SQL Server database?

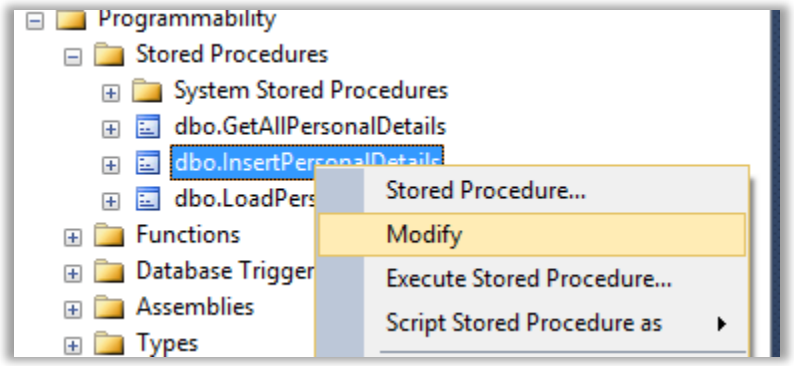
To create delete stored procedure, replace the UPDATE statement with DELETE statement and change the parameters accordingly.

```
CREATE PROCEDURE DeletePersonalDetails
       -- Add the parameters for the stored procedure here
       @PersonalDetailsId int
AS
BEGIN
       -- SET NOCOUNT ON added to prevent extra result sets from
       -- interfering with SELECT statements.
       SET NOCOUNT ON;

    -- Insert statements for procedure here
       DELETE PersonalDetails WHERE PersonalDetailsId = @PersonalDetailsId
END
```

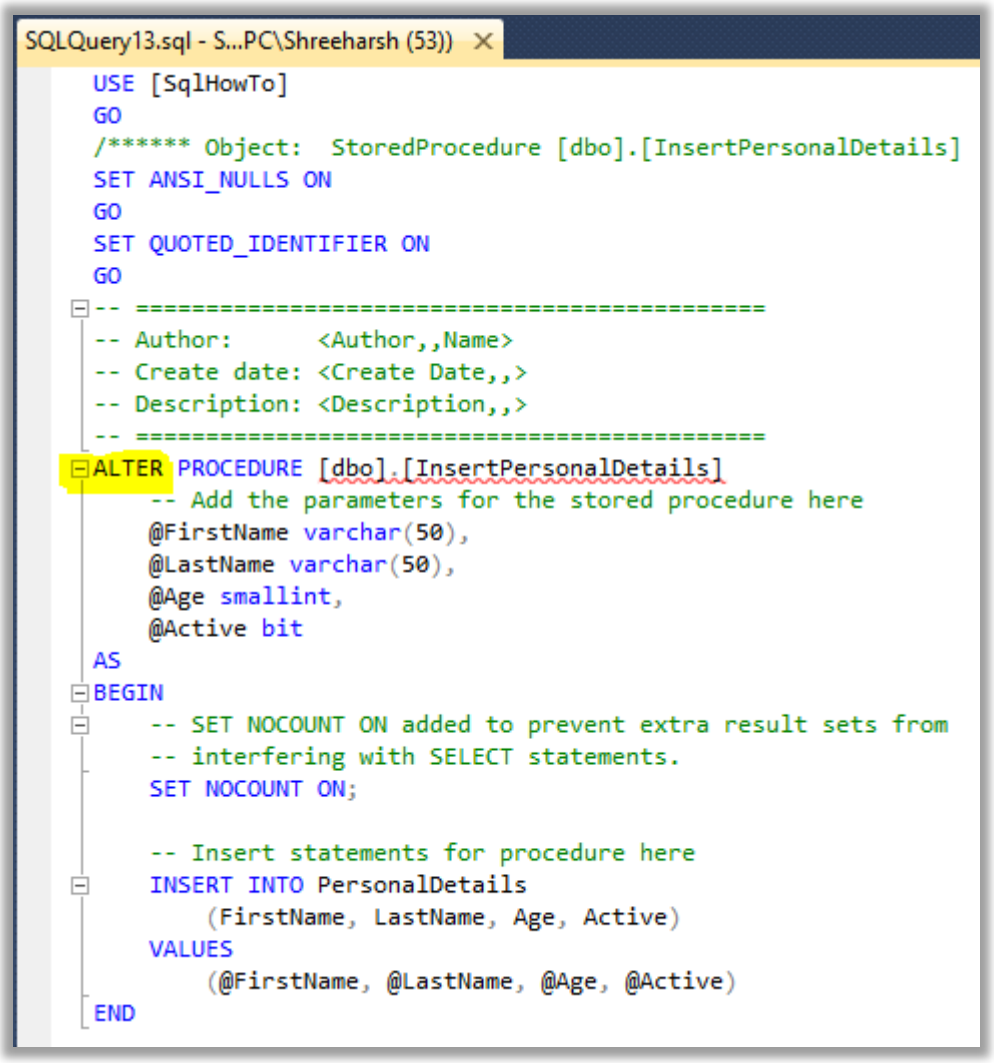
Notice that we have only one parameter in the above stored procedure as we need only that to find out the record having PersonalDetailsId and delete it.

86. How to alter / modify a stored procedure in SQL Server?

To alter the stored procedure, right click it and select Modify that brings the stored procedure in new query window with ALTER statement.



Now, alter the stored procedure (altering means addition / deletion / modifying the parameters and its type, altering the SQL statements etc.) and press Execute icon from the toolbar or hit F5 key from the keyboard that will modify the stored procedure.



87. How to create stored procedure to insert record and return identity column value?

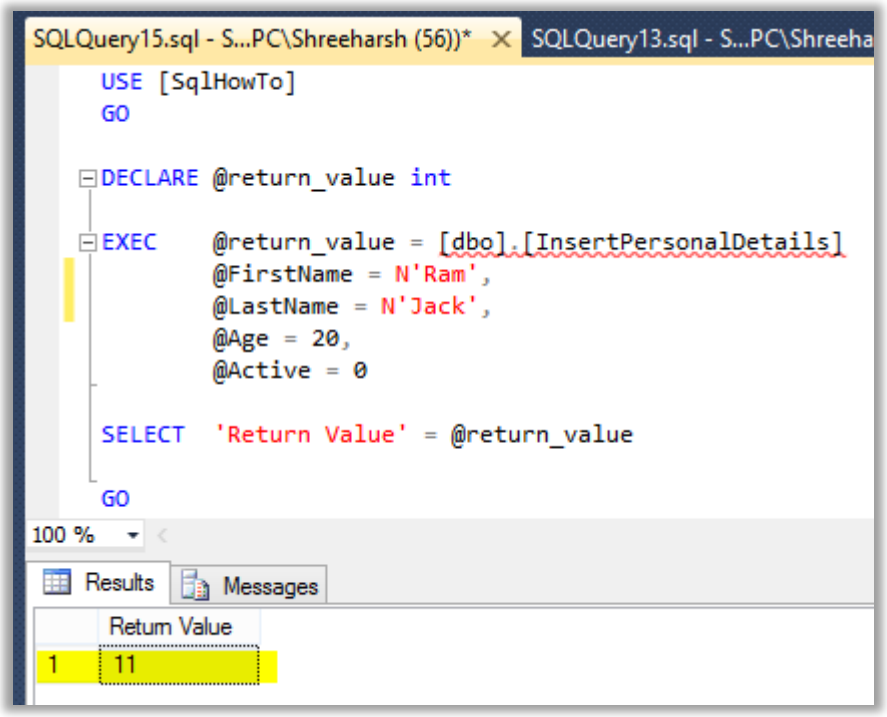
To return the identity column value for just inserted record, we can use SCOPE_IDENTITY() method. To see how it works, modify the InsertPersonalDetails stored procedure like below

```
ALTER PROCEDURE [dbo].[InsertPersonalDetails]
	-- Add the parameters for the stored procedure here
	@FirstName varchar(50),
	@LastName varchar(50),
	@Age smallint,
	@Active bit
AS
BEGIN
```

```
        -- SET NOCOUNT ON added to prevent extra result sets from
        -- interfering with SELECT statements.
        SET NOCOUNT ON;

    -- Insert statements for procedure here
        INSERT INTO PersonalDetails
               (FirstName, LastName, Age, Active)
        VALUES
               (@FirstName, @LastName, @Age, @Active)

        return SCOPE_IDENTITY()
END
```

Notice the highlighted line of code where we are using RETURN keyword with the SCOPE_IDENTITY() method. When we execute this stored procedure, after inserting the record, it returns the PersonalDetialsId value of last inserted record.

Now running the modified stored procedure gives a return value like below



Important

SCOPE_IDENTITY() method only returns the Identity (auto increment) column value. If the primary key value is of different type, it will not return desired data.

88. How to create stored procedure to return paginated data (custom paginations) from SQL Server database?

To get the paginated data from stored procedure, we need to pass at least three parameters if the paginated data doesn't need any filtration.

i. startRowIndex – start fetching records from which row number
ii. pageSize – number of records to fetch
iii. totalCount – total count of records into the database table that will be used to render the page number links on the web page so that user will be able to navigate to different pages.

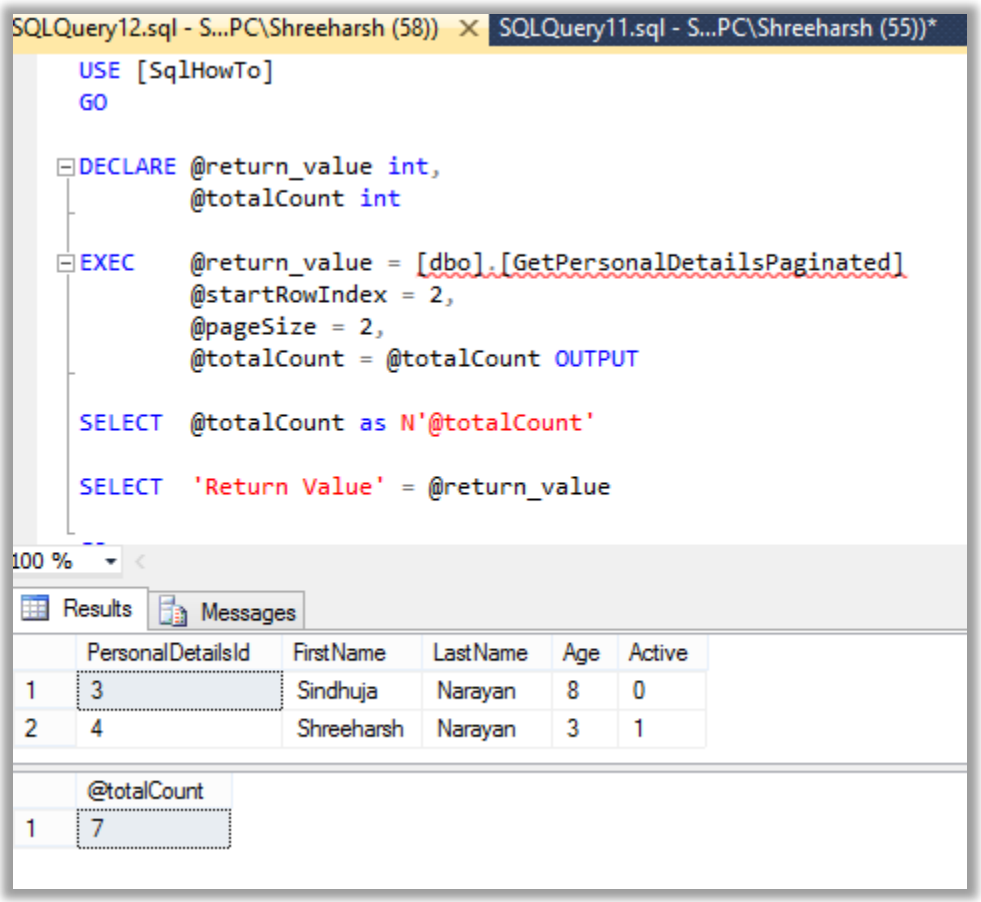
```
CREATE PROCEDURE GetPersonalDetailsPaginated
	-- Add the parameters for the stored procedure here
	@startRowIndex int,
	@pageSize int,
	@totalCount int OUT
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

	-- Get the paginated records
	SELECT PersonalDetailsId, FirstName, LastName, Age, Active FROM
PersonalDetails
		-- keep WHERE clause if any
	ORDER BY PersonalDetailsId
	OFFSET @startRowindex ROWS FETCH NEXT @pageSize ROWS ONLY

	-- get the total count of the records
	SELECT @totalCount = COUNT(PersonalDetailsId) FROM PersonalDetails
		-- keep the same WHERE clause as above
```

Note that we have @startRowIndex and @pageSize as normal parameters (input parameter) and @totalCount as OUT (output parameter) whose value should be set inside the stored procedure.

Write above SQL code into the query window and execute it that will create the stored procedure. In this stored procedure, we are first selecting PersonalDetails records ordered by PersonalDetailsId and then using OFFSET (skip following number of records) the value of @startRowIndex and the FETCH NEXT @pageSize rows from the PersonalDetials table.

The next SELECT statement is setting the value of the @totalCount.

Now when we Execute Stored procedure by right clicking the stored procedure name, we see that the first result set shows as the paginated record and the second result as the totalCount value that is nothing but the total count of the records in the PersonalDetails database table.



89. How to create stored procedure that accepts optional parameter in SQL Server?

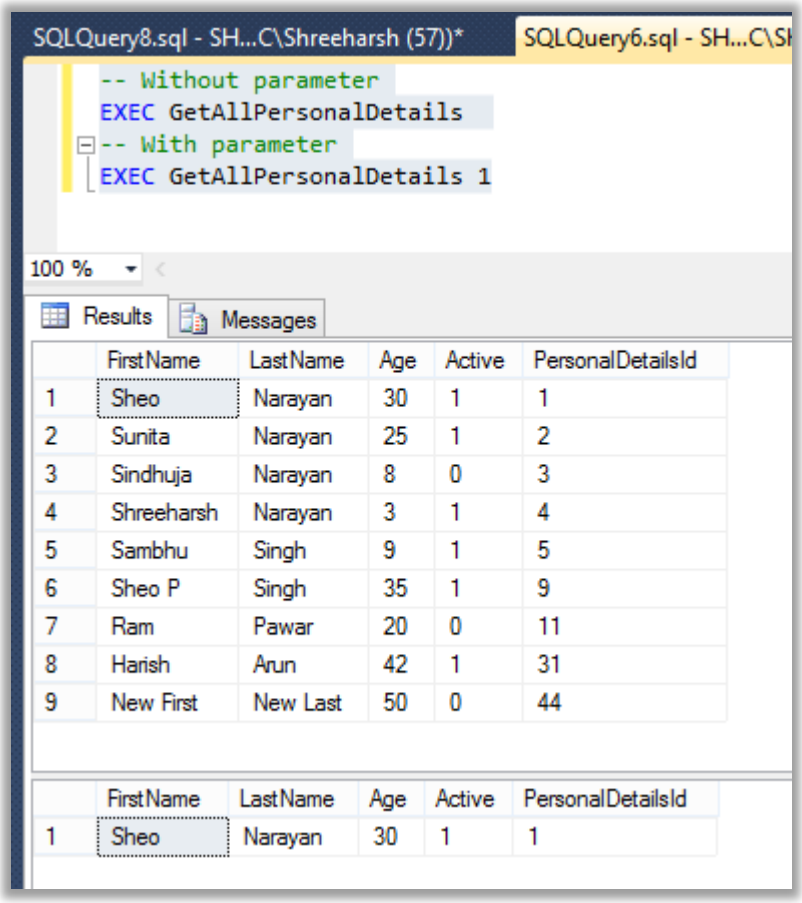
To create optional parameter in stored procedure, we set the parameter value to NULL.

```
CREATE PROCEDURE [dbo].[GetAllPersonalDetails]
	@personalDetailsId int = null
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

    -- Insert statements for procedure here
	IF @personalDetailsId is null
		BEGIN
			SELECT * FROM PersonalDetails
		END
	ELSE
		BEGIN
			SELECT * FROM PersonalDetails
```

```
                        WHERE PersonalDetailsId = @personalDetailsId
            END

END
```

The above stored procedure may or may not be passed @personalDetailsId parameter value and it can be called in any of the following way

```
-- Without parameter
EXEC GetAllPersonalDetails
-- With parameter
EXEC GetAllPersonalDetails 1
```



	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1
2	Sunita	Narayan	25	1	2
3	Sindhuja	Narayan	8	0	3
4	Shreeharsh	Narayan	3	1	4
5	Sambhu	Singh	9	1	5
6	Sheo P	Singh	35	1	9
7	Ram	Pawar	20	0	11
8	Harish	Arun	42	1	31
9	New First	New Last	50	0	44

	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sheo	Narayan	30	1	1

Transactions

90. How to use transaction in stored procedure in SQL Server?

Simply put transaction is used to ensure that either all SQL statements gets executed successfully or no one gets executed successfully. If we have more than one SQL statements in execute in the stored procedure and we want to rollback any changes done by any one of the SQL statements in case an error occurred because of one of the SQL statements, we can use transaction in stored procedure.

Below is the stored procedure that is trying to insert a record into PersonalDetails and Accounts table using two INSERT statement. Our scenario should be that If any of the INSERT statements fails to execute, no record should be inserted into any of the table.

```
CREATE PROCEDURE [dbo].[InsertPersonalDetailsAndAccount]
	-- Add the parameters for the stored procedure here
	@FirstName varchar(50),
	@LastName varchar(50),
	@Age smallint,
	@Active bit,
	@Salary money,
	@PPFDeduction money
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

	BEGIN TRAN
		BEGIN TRY
			-- Insert into PersonalDetails table first
			INSERT INTO PersonalDetails
				(FirstName, LastName, Age, Active)
			VALUES
				(@FirstName, @LastName, @Age, @Active)


			DECLARE @pdId int
			SET @pdId = SCOPE_IDENTITY()
			-- now insert into Accounts table
			INSERT INTO Accounts
				(Salary, PPFDeduction, PersonalDetailsId)
			VALUES
				(@Salary, @PPFDeduction, @pdId)

		-- if not error, commit the transcation
		COMMIT TRANSACTION
	END TRY
	BEGIN CATCH
		-- if error, roll back any chanegs done by any of the sql statements
		ROLLBACK TRANSACTION
	END CATCH

END
```

Notice the BEGIN TRAN statement that is creating a transaction scope. After that we are using BEGIN TRY statement where we are going to keep our INSERT statements that may throw errors. After both INSERT statements, we are calling COMMIT TRANSACTION statements to notify that everything is

alright and the data can be saved into the database permanently. If any error occurs in any of the INSERT statements inside the BEGIN TRY block, the BEGIN CATCH block executes that calls ROLLBACK TRANSACTION. This will rollback any changes done in the database because of these two INSERT statements inside the TRY BLOCK.

Error Handling

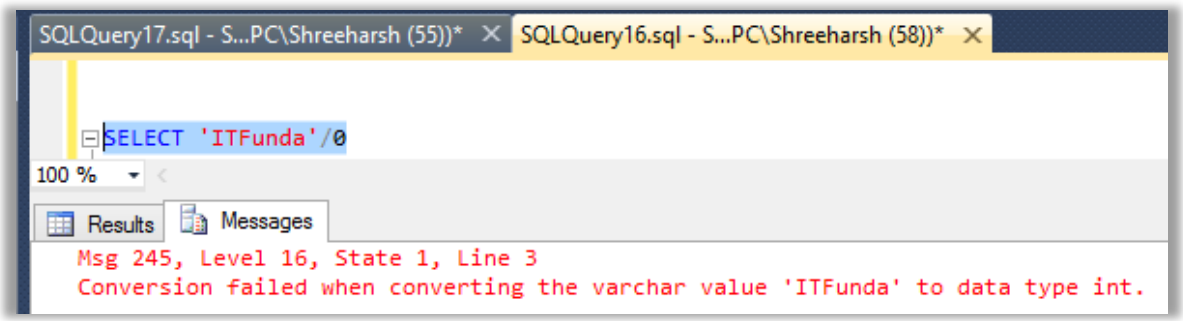
91. How to handle error in stored procedures in SQL Server?

To handle error in stored procedure of SQL Server, we use BEGING TRY – END TRY and BEGIN CATCH and EDN CATCH statements.

Normally, if we perform any invalid SQL operations, it throws error like in below case

```
SELECT 'ITFunda'/0
```

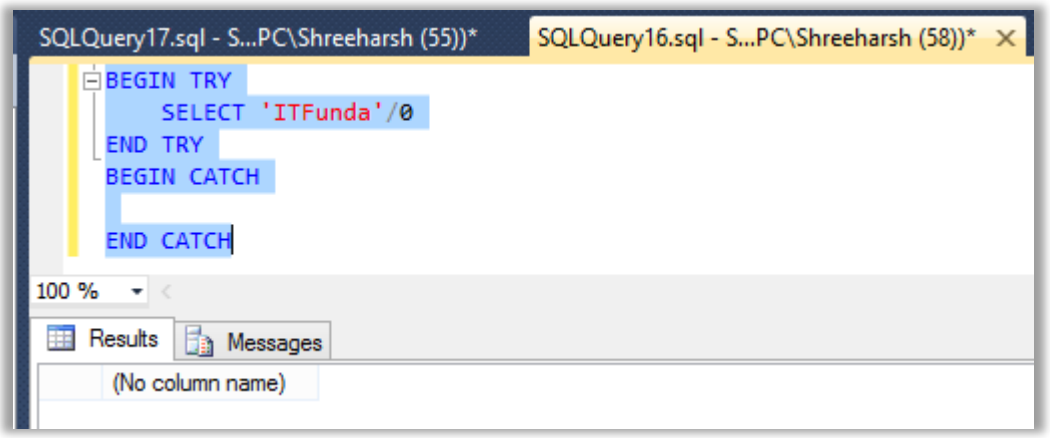
Here, it is not possible divide any string by 0, so SQL Server is trying to convert it to integer however 'ITFunda' couldn't get converted and it throws error.



Instead of raw error coming in, we want to ignore this error, we can wrap the above statements into BEGIN and END TRY like this

```
BEGIN TRY
       SELECT 'ITFunda'/0
END TRY
BEGIN CATCH

END CATCH
```

Here, the SELECT statement will throw error that is being caught in the BEGIN CATCH block, but this block is not doing anything but suppressing the error. So the output is showing nothing without any error.

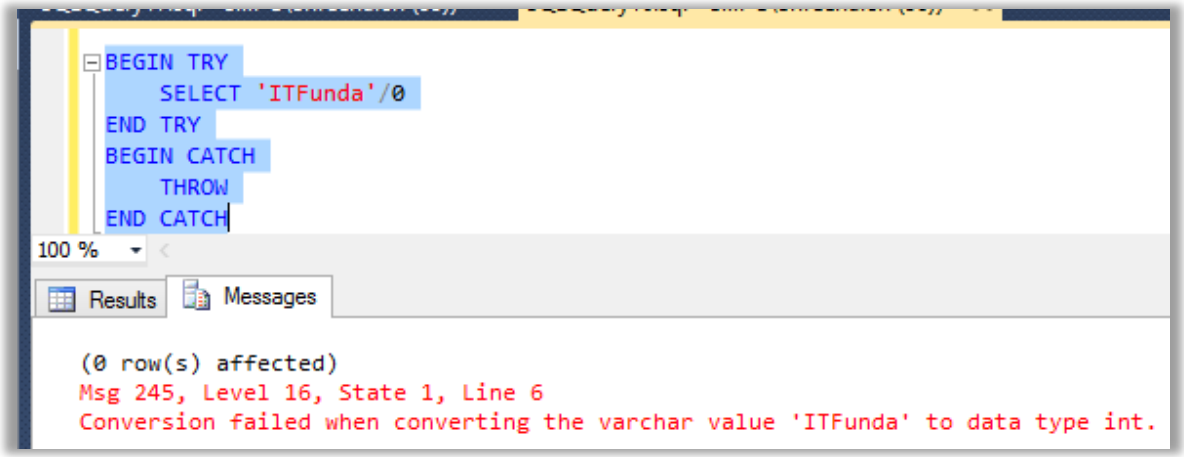


92. How to catch and throw error in SQL Server?

To throw error, we use THROW statement.

```
BEGIN TRY
	SELECT 'ITFunda'/0
END TRY
BEGIN CATCH
	THROW
END CATCH
```

Here, notice the BEGIN CATCH block that uses THROW statement to throw the error.



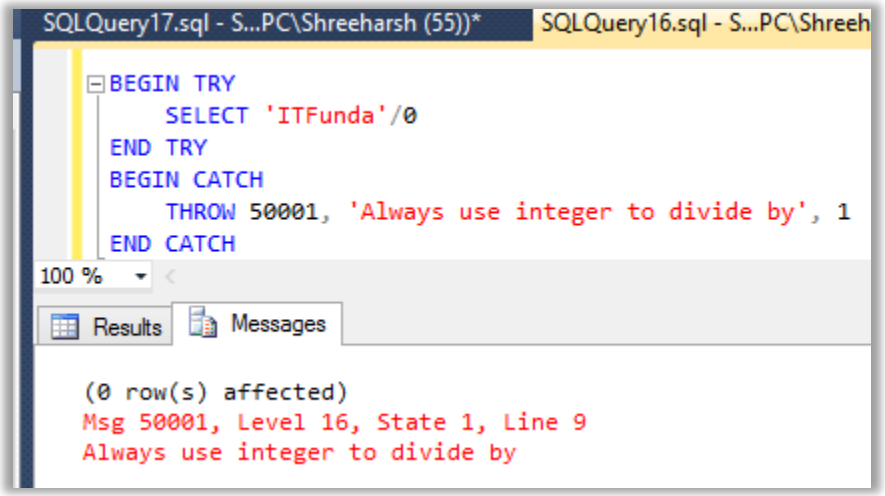
93. How to return a custom error message from SQL Server in case of error?

To return custom error message in case of error, we pass necessary parameter to the THROW statement.

```
--- SELECT 'ITFunda'/0

BEGIN TRY
	SELECT 'ITFunda'/0
END TRY
BEGIN CATCH
	THROW 50001, 'Always use integer to divide by', 1
END CATCH
```

Throw has three parameters,

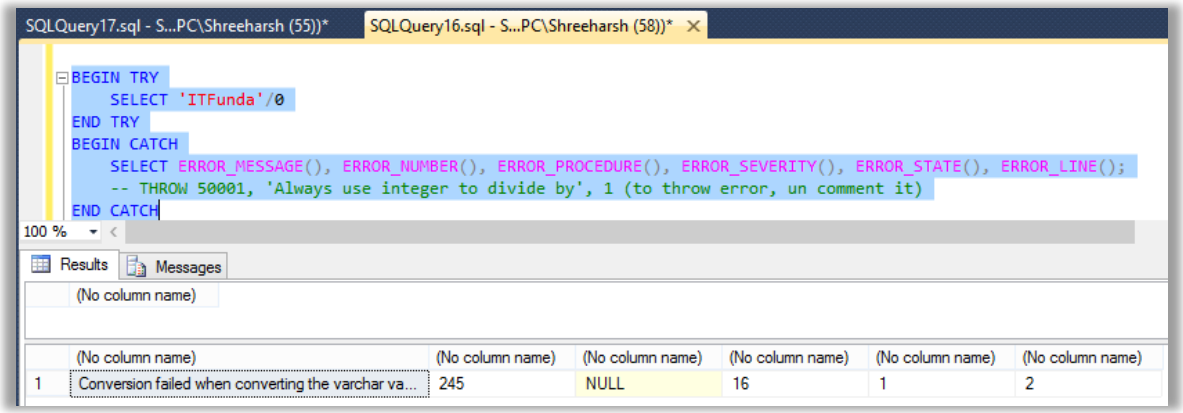
i. ErrorNumber – must be greater than 50000 and less than 2147483647
ii. ErrorMessage – the error message to throw
iii. State – generally in between 0 to 255, to associate the state of the error message.



94. How to get more details about an error occurred in SQL Server?

To get more error related values in the CATCH block, try to use error methods, used in the SELECT statement inside the BEGIN CATCH block.

```
BEGIN TRY
	SELECT 'ITFunda'/0
END TRY
BEGIN CATCH
	SELECT ERROR_MESSAGE(), ERROR_NUMBER(), ERROR_PROCEDURE(),
ERROR_SEVERITY(), ERROR_STATE(), ERROR_LINE();
	-- THROW 50001, 'Always use integer to divide by', 1 (to throw error, un
comment it)
END CATCH
```



Loops and Conditions

Loops and conditions are used to loop through certain number of times and conditions are used to check for a variable value.

95. How to use IF condition in SQL Server?

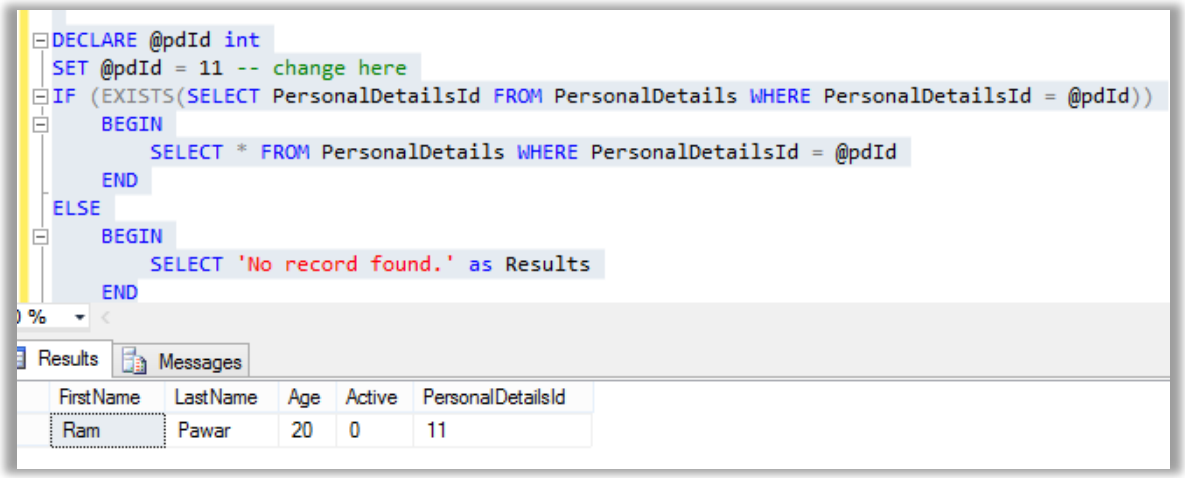
If condition works in SQL server in the same way it works in other programming language.

```
DECLARE @pdId int
SET @pdId = 11 -- change here
IF (EXISTS(SELECT PersonalDetailsId FROM PersonalDetails WHERE PersonalDetailsId =
@pdId))
	BEGIN
		SELECT * FROM PersonalDetails WHERE PersonalDetailsId = @pdId
	END
ELSE
	BEGIN
		SELECT 'No record found.' as Results
	END
```

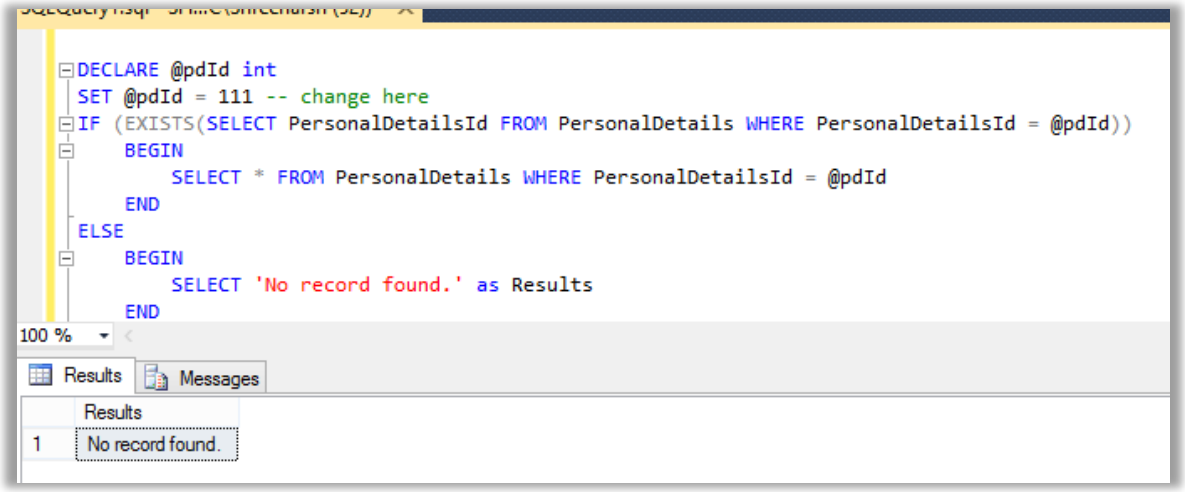
Look at above SQL code, we have first declared @pdId variable and setting its value as 11.

In the next like we are checking for using EXISTS in-built function with parameter as the record from PersonalDetails whose PersonalDetailsId as @pdId. If the record is found, EXISTS returns true and the first IF block executes and gives all columns from PersonalDetails otherwise 'No record found' message is show.

Records found



Records not found

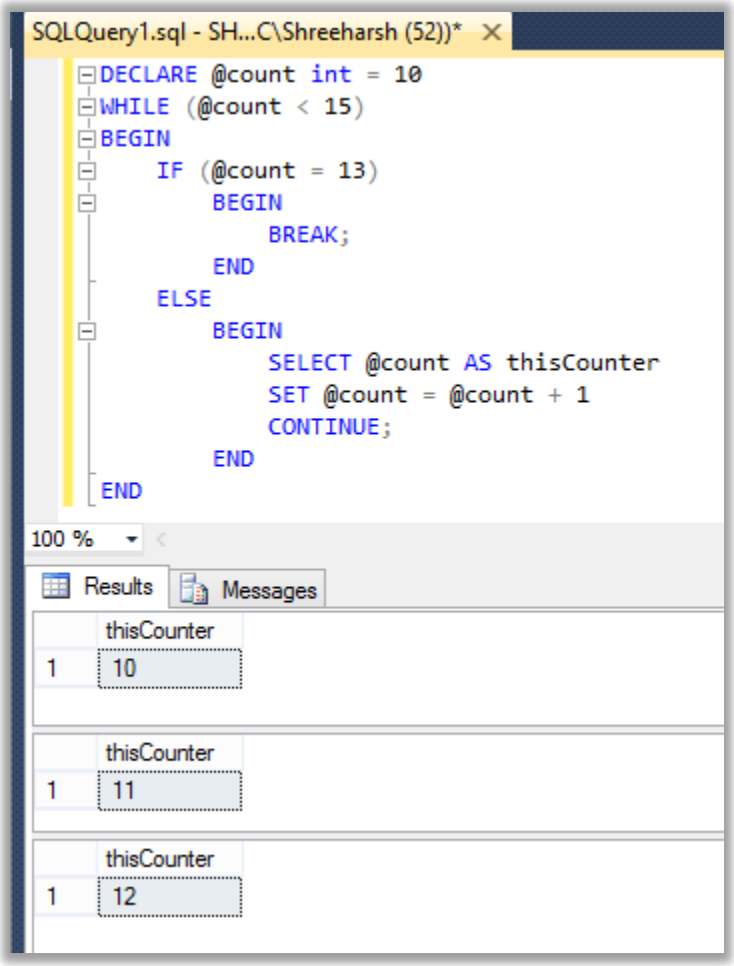


96. How to use WHILE loop in SQL Server?

Similar to IF condition, WHILE loop is also used as in other programming language.

```
DECLARE @count int = 10
WHILE (@count < 15)
BEGIN
	IF (@count = 13)
		BEGIN
			BREAK;
		END
	ELSE
		BEGIN
			SELECT @count AS thisCounter
			SET @count = @count + 1
			CONTINUE;
		END
END
```

In the above code snippet, the WHILE loop is marked to run 5 times however when the value of @count is reaching 13, we are breaking the loop using BREAK statement otherwise CONTINUE statement iterate to the next iteration of the loop.

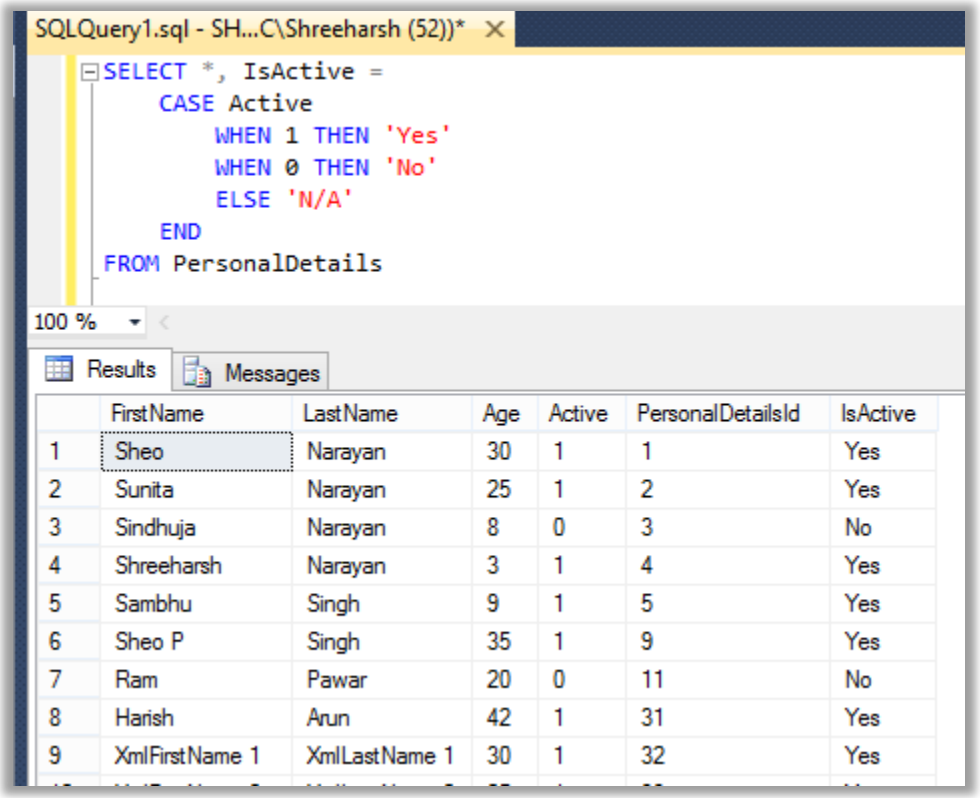


97. How to use CASE statement to check a value of the column in SQL Server?

SWITCH case is used to evaluate several conditions and returns one of multiple possible result.

```
SELECT *, IsActive =
       CASE Active
              WHEN 1 THEN 'Yes'
              WHEN 0 THEN 'No'
              ELSE 'N/A'
       END
FROM PersonalDetails
```

In above case, we are trying to get all column value of the PersonalDetails and adding another column named “IsActive” and when Active column value of the database table is 1 (true) then setting “IsActive” value to ‘Yes’, when 0 (false) then ‘No’ else setting it to ‘N/A’.

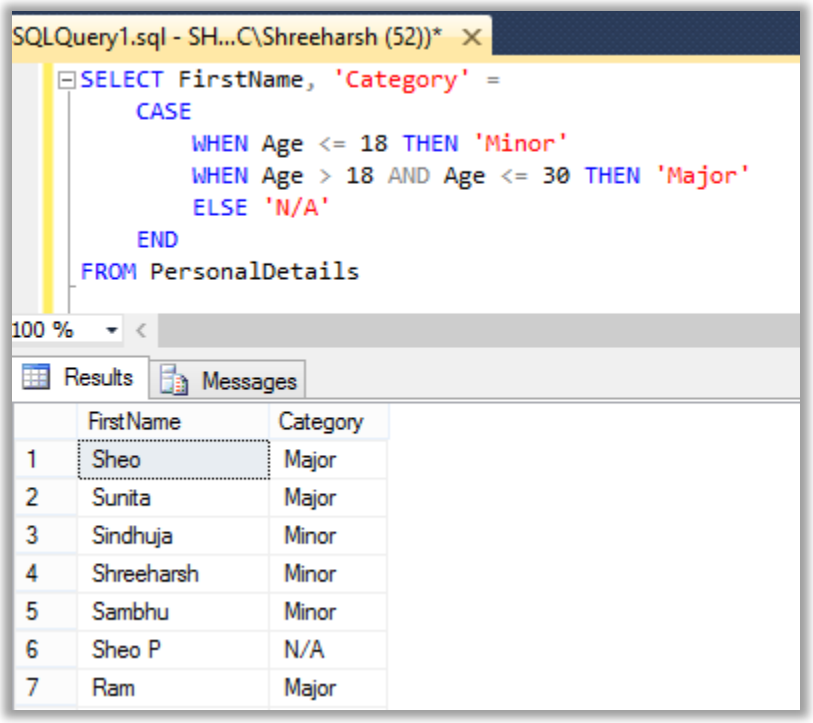


	FirstName	LastName	Age	Active	PersonalDetailsId	IsActive
1	Sheo	Narayan	30	1	1	Yes
2	Sunita	Narayan	25	1	2	Yes
3	Sindhuja	Narayan	8	0	3	No
4	Shreeharsh	Narayan	3	1	4	Yes
5	Sambhu	Singh	9	1	5	Yes
6	Sheo P	Singh	35	1	9	Yes
7	Ram	Pawar	20	0	11	No
8	Harish	Arun	42	1	31	Yes
9	XmlFirstName 1	XmlLastName 1	30	1	32	Yes

98. How to use CASE statement to check for searched condition in SQL Server?

```
SELECT FirstName, 'Category' =
       CASE
              WHEN Age <= 18 THEN 'Minor'
              WHEN Age > 18 AND Age <= 30 THEN 'Major'
              ELSE 'N/A'
       END
FROM PersonalDetails
```

In above case, we are checking for value of Age column, WHEN Age <= 18 then setting ‘Category’ column value to ‘Minor’, when the age is in between 19 to 30 then setting ‘Category’ value to ‘Major’ else ‘N/A’.

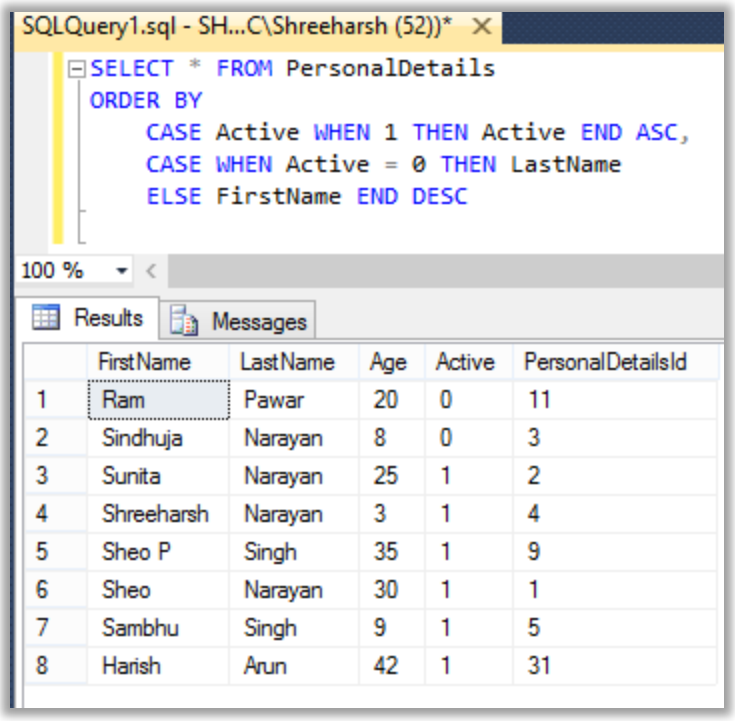


99. How to use CASE statement with ORDER BY clause in SQL Server?

To do this, we use the CASE after ORDER BY and then checks for column value.

```
SELECT * FROM PersonalDetails
ORDER BY
       CASE Active WHEN 1 THEN Active END ASC,
       CASE WHEN Active = 0 THEN LastName
       ELSE FirstName END DESC
```

In above case, all records having Active = 1 is sorted on “Active ASC” order. All records having Active = 0 is sorted on ‘LastName DESC’ else ‘FirstName DESC’ order.

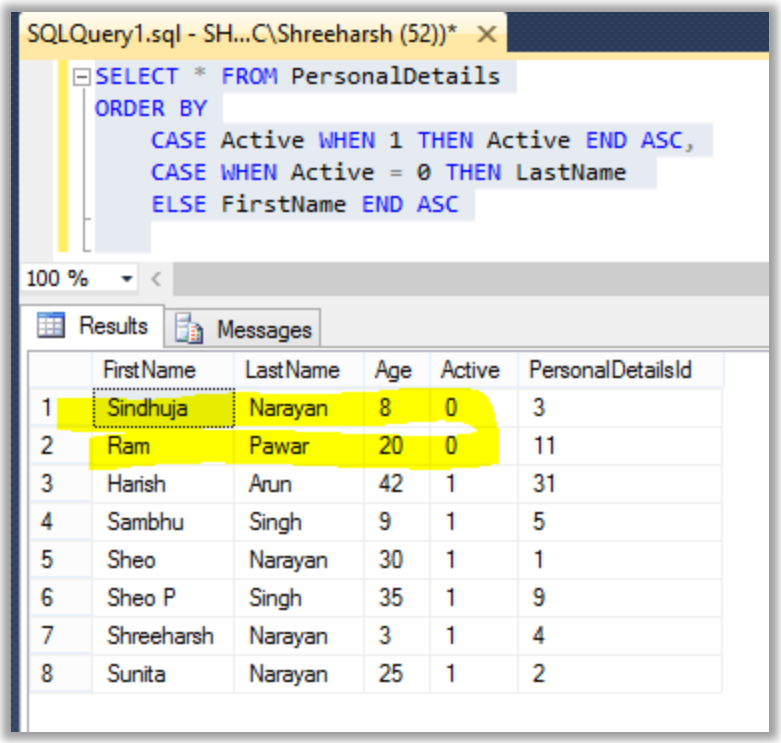


	FirstName	LastName	Age	Active	PersonalDetailsId
1	Ram	Pawar	20	0	11
2	Sindhuja	Narayan	8	0	3
3	Sunita	Narayan	25	1	2
4	Shreeharsh	Narayan	3	1	4
5	Sheo P	Singh	35	1	9
6	Sheo	Narayan	30	1	1
7	Sambhu	Singh	9	1	5
8	Harish	Arun	42	1	31

Now look at the below statements, where we have just changed the last word from DESC to ASC.

```
SELECT * FROM PersonalDetails
ORDER BY
       CASE Active WHEN 1 THEN Active END ASC,
       CASE WHEN Active = 0 THEN LastName
       ELSE FirstName END ASC
```

And the order of records having Active = 0 has changed. Notice the highlighted records in below picture.

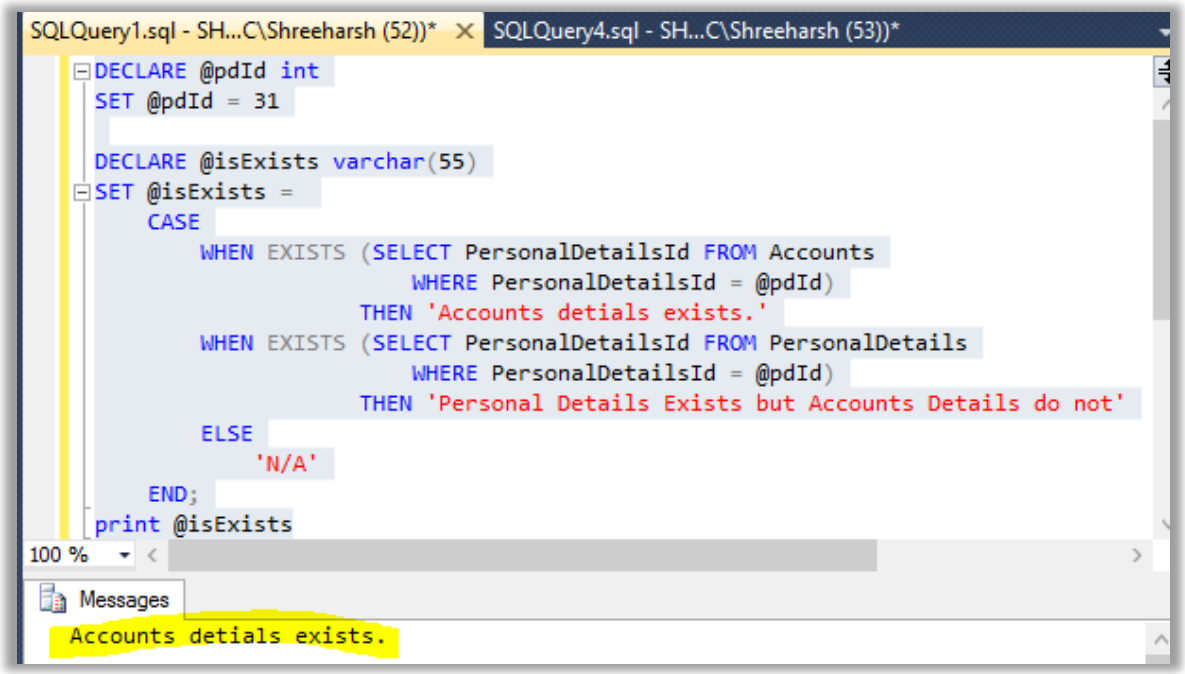


	FirstName	LastName	Age	Active	PersonalDetailsId
1	Sindhuja	Narayan	8	0	3
2	Ram	Pawar	20	0	11
3	Harish	Arun	42	1	31
4	Sambhu	Singh	9	1	5
5	Sheo	Narayan	30	1	1
6	Sheo P	Singh	35	1	9
7	Shreeharsh	Narayan	3	1	4
8	Sunita	Narayan	25	1	2

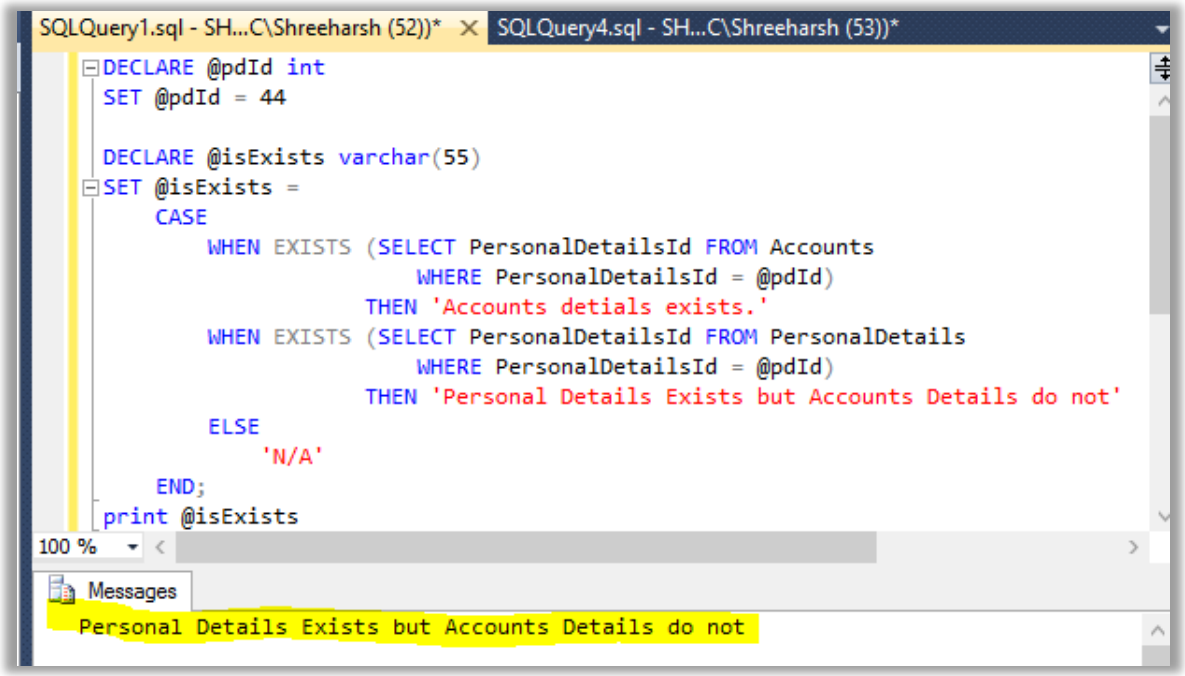
100. How to use CASE statement to set a variable value in SQL Server?

```
DECLARE @pdId int
SET @pdId = 31

DECLARE @isExists varchar(55)
SET @isExists =
	CASE
		WHEN EXISTS (SELECT PersonalDetailsId FROM Accounts
					WHERE PersonalDetailsId = @pdId)
				THEN 'Accounts detials exists.'
		WHEN EXISTS (SELECT PersonalDetailsId FROM PersonalDetails
					WHERE PersonalDetailsId = @pdId)
				THEN 'Personal Details Exists but Accounts
Details do not'
		ELSE
			'N/A'
	END;
print @isExists
```

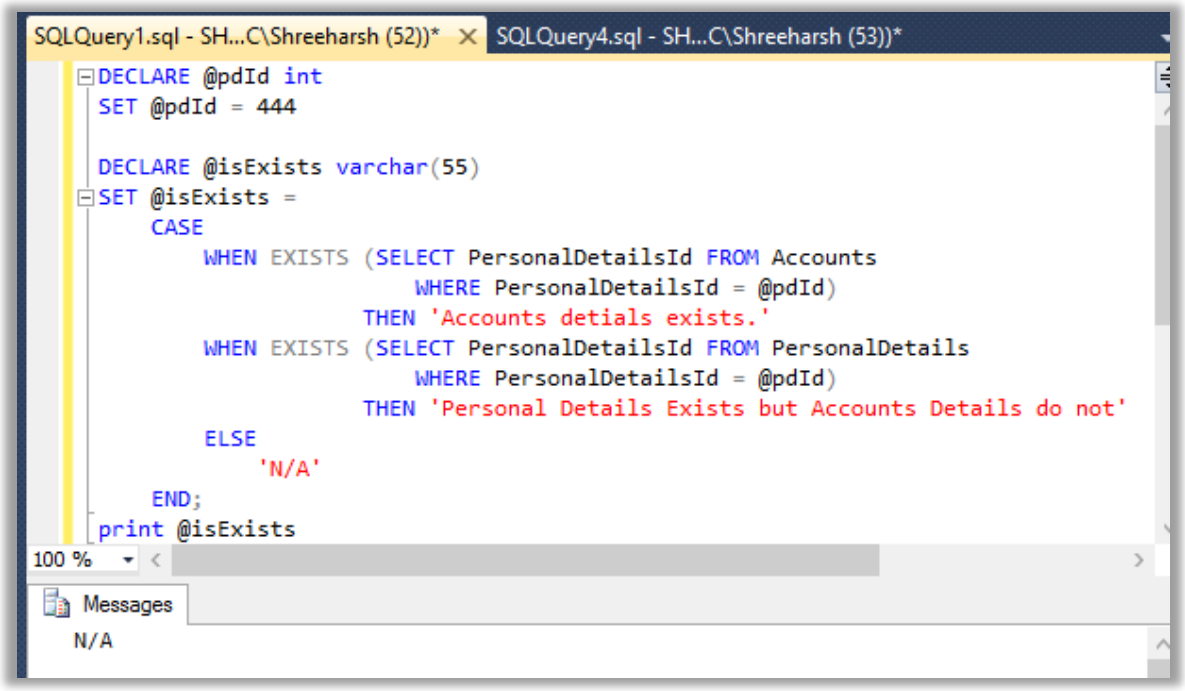
In above case, the value of @pdId is 31 that exists in the Accounts table and the value of the @isExists variable is set to 'Accounts details exists'.



When the value of @pdId is 44 that doesn't exists in the Accounts table but it exists in PersonalDetails table, the value of @isExists is set to 'Personal Details Exists but Accounts Details do not'.



If the @pdId is set to something that neither exists in the PersonalDetails table and Accounts table, @isExists value is set to 'N/A'.



Common Table Expression (CTE)

Common table expression is a temporary result set that we define during the execution of the SQL statements. The CTE is a run time object whose scope is limited to the execution of the query just after the CTE is defined. Generally, it is used to replace views, perform recursion, creating multiple reference of the single table data or hold temporary data.

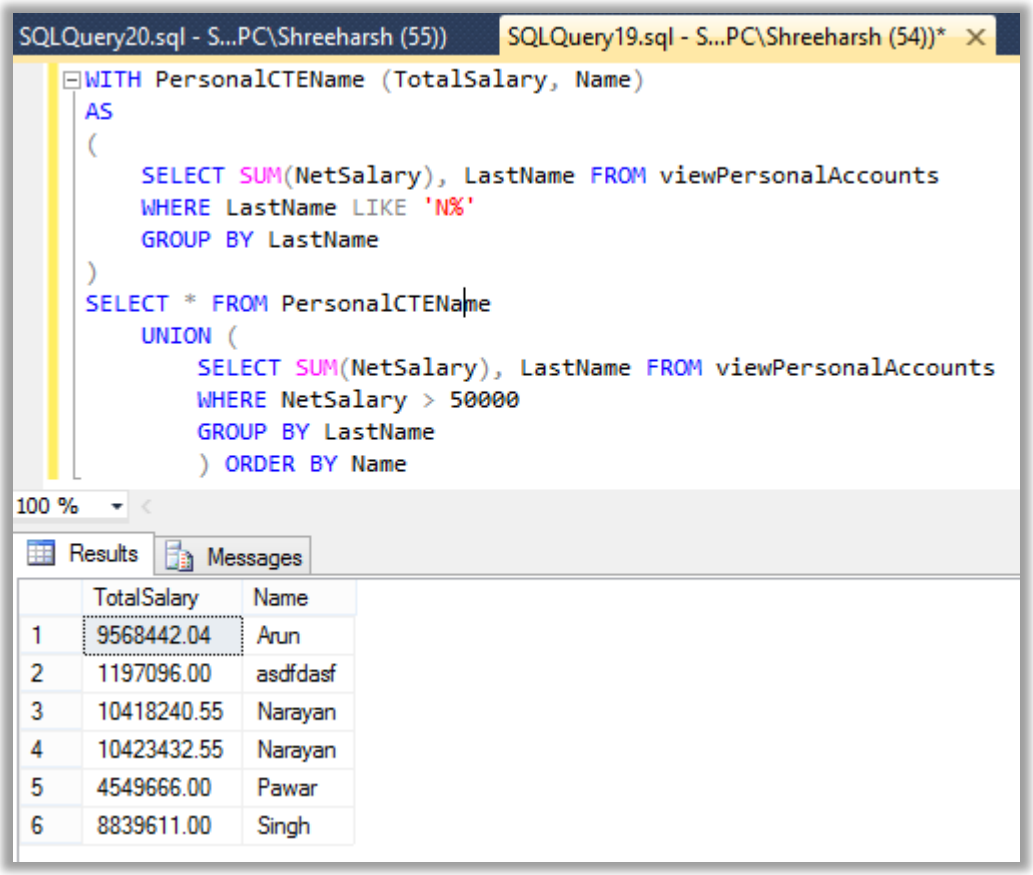
101. How to create a CTE (Common Table Expression) and use it?

To create CTE, use highlighted approach. Read below SQL statements carefully

```
WITH PersonalCTEName (TotalSalary, Name)
AS
(
       SELECT SUM(NetSalary), LastName FROM viewPersonalAccounts
       WHERE LastName LIKE 'N%'
       GROUP BY LastName
)
SELECT * FROM PersonalCTEName
       UNION (
              SELECT SUM(NetSalary), LastName FROM viewPersonalAccounts
              WHERE NetSalary > 50000
              GROUP BY LastName
              ) ORDER BY Name
```

The CTE definition starts with "WITH" keyword followed by the name of the CTE and then the column that this CTE will return. Under bracket, we write the SQL statements to return the data from this CTE (this SQL statements can be any valid SQL statements).

Here, the first highlighted part is the CTE defined whose data is being fetched from the following SELECT statement (SELECT * FROM PersonalCTEName) and we are using UNION clause to append the data from next query (get sum of records whose NetSalary is more than 50000 grouped by LastName) and then the combination of record from CTE and UNION block result is being sorted by Name.



Working with XML in SQL Server

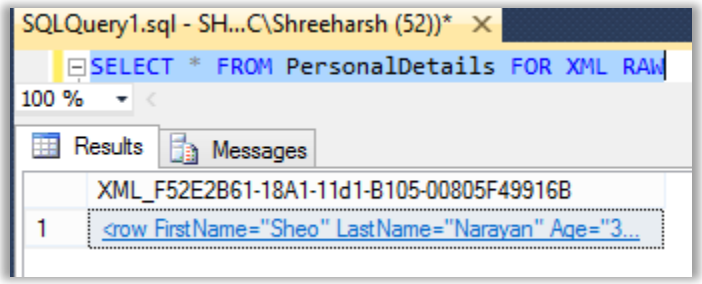
SQL Server is well capable to storing XML data, retrieving normal table data in XML or searching XML data. In this section, we shall learn all of them.

102. How to retrieve data in XML format from SQL Server?

To retrieve data in XML format from SQL Server database, we can use FOR XML <options> clause.

```
SELECT * FROM PersonalDetails FOR XML RAW
```

Notice the last three words in the above query. We have a normal SELECT statement and in the last of the statement, we are suffixing FOR XML RAW that will return the data from PersonalDetails table in Raw xml format.



The XML data

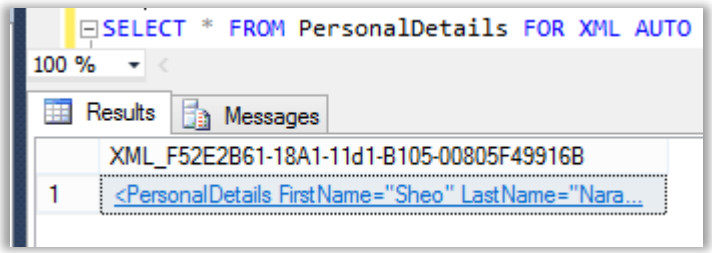
```
<row FirstName="Sheo" LastName="Narayan" Age="30" Active="1" PersonalDetailsId="1"
/>
<row FirstName="Sunita" LastName="Narayan" Age="25" Active="1"
PersonalDetailsId="2" />
<row FirstName="Sindhuja" LastName="Narayan" Age="8" Active="0"
PersonalDetailsId="3" />
<row FirstName="Shreeharsh" LastName="Narayan" Age="3" Active="1"
PersonalDetailsId="4" />
<row FirstName="Sambhu" LastName="Singh" Age="9" Active="1" PersonalDetailsId="5"
/>
<row FirstName="Sheo P" LastName="Singh" Age="35" Active="1" PersonalDetailsId="9"
/>
<row FirstName="Ram" LastName="Pawar" Age="20" Active="0" PersonalDetailsId="11"
/>
<row FirstName="Harish" LastName="Arun" Age="42" Active="1" PersonalDetailsId="31"
/>
```

Notice the node name here, it is row that simply represent each row of the database table.

When we change the option to AUTO

```
SELECT * FROM PersonalDetails FOR XML AUTO
```

The same query returns almost same XML as in the previous case however the XML node name changes to the database table name.



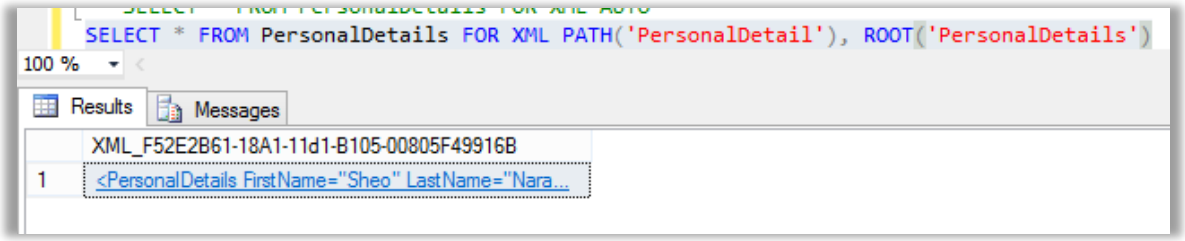
```
<PersonalDetails FirstName="Sheo" LastName="Narayan" Age="30" Active="1"
PersonalDetailsId="1" />
<PersonalDetails FirstName="Sunita" LastName="Narayan" Age="25" Active="1"
PersonalDetailsId="2" />
<PersonalDetails FirstName="Sindhuja" LastName="Narayan" Age="8" Active="0"
PersonalDetailsId="3" />
<PersonalDetails FirstName="Shreeharsh" LastName="Narayan" Age="3" Active="1"
PersonalDetailsId="4" />
<PersonalDetails FirstName="Sambhu" LastName="Singh" Age="9" Active="1"
PersonalDetailsId="5" />
<PersonalDetails FirstName="Sheo P" LastName="Singh" Age="35" Active="1"
PersonalDetailsId="9" />
<PersonalDetails FirstName="Ram" LastName="Pawar" Age="20" Active="0"
PersonalDetailsId="11" />
<PersonalDetails FirstName="Harish" LastName="Arun" Age="42" Active="1"
PersonalDetailsId="31" />
```

Notice the node name “PersonalDetails” ie. the name of the database table.

In case, we want a custom parent node and child node of the XML, we can specify PATH and ROOT parameter to the last like this

```
SELECT * FROM PersonalDetails FOR XML PATH('PersonalDetail'),
ROOT('PersonalDetails')
```

Here, we have specified PATH parameter value as “PersonalDetial” and ROOT as “PersonalDetials” and the same will get applied in the resultant XML.



We can specify any string with PATH and ROOT, not necessarily in line with the database table names.

```
<PersonalDetails>
  <PersonalDetail>
    <FirstName>Sheo</FirstName>
    <LastName>Narayan</LastName>
    <Age>30</Age>
    <Active>1</Active>
    <PersonalDetailsId>1</PersonalDetailsId>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>Sunita</FirstName>
    <LastName>Narayan</LastName>
    <Age>25</Age>
    <Active>1</Active>
    <PersonalDetailsId>2</PersonalDetailsId>
  </PersonalDetail>
```

```
  <PersonalDetail>
    <FirstName>Sindhuja</FirstName>
    <LastName>Narayan</LastName>
    <Age>8</Age>
    <Active>0</Active>
    <PersonalDetailsId>3</PersonalDetailsId>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>Shreeharsh</FirstName>
    <LastName>Narayan</LastName>
    <Age>3</Age>
    <Active>1</Active>
    <PersonalDetailsId>4</PersonalDetailsId>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>Sambhu</FirstName>
    <LastName>Singh</LastName>
    <Age>9</Age>
    <Active>1</Active>
    <PersonalDetailsId>5</PersonalDetailsId>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>Sheo P</FirstName>
    <LastName>Singh</LastName>
    <Age>35</Age>
    <Active>1</Active>
    <PersonalDetailsId>9</PersonalDetailsId>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>Ram</FirstName>
    <LastName>Pawar</LastName>
    <Age>20</Age>
    <Active>0</Active>
    <PersonalDetailsId>11</PersonalDetailsId>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>Harish</FirstName>
    <LastName>Arun</LastName>
    <Age>42</Age>
    <Active>1</Active>
    <PersonalDetailsId>31</PersonalDetailsId>
  </PersonalDetail>
</PersonalDetails>
```

103. How to insert records from XML to SQL Server database table?

To insert records from XML data into SQL Server database table, we use XML data type as input parameter.

Create a stored procedure like below with Xml as input parameter type.

```
CREATE PROCEDURE InsertPersonalDetailsFromXML
       -- Add the parameters for the stored procedure here
       @xmlData XML
```

```
AS
BEGIN
	-- SET NOCOUNT ON added to prevent extra result sets from
	-- interfering with SELECT statements.
	SET NOCOUNT ON;

    -- Insert statements for procedure here

	INSERT INTO PersonalDetails
	(FirstName, LastName, Age, Active)

	SELECT x.value('FirstName[1]', 'varchar(50)') AS FirstName,
	    x.value('LastName[1]', 'varchar(50)') AS LastName,
		    x.value('Age[1]', 'int') AS Age,
		    x.value('Active[1]', 'bit') AS Active
	FROM @xmlData.nodes('//PersonalDetail') XmlData(x)
END
```

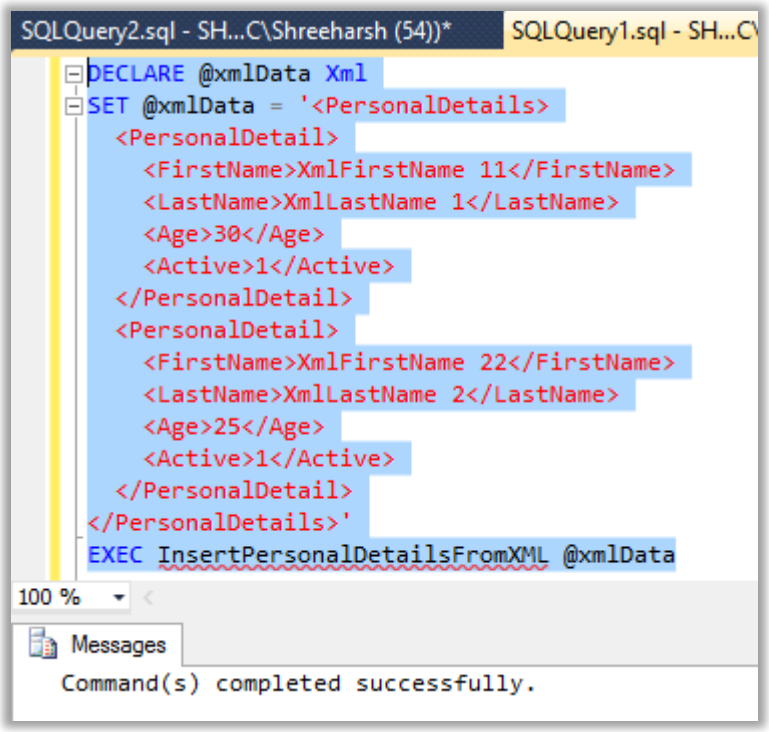
Notice that the input parameter named @xmlData is of XML type. Next, we are using INSERT statement and selecting the value of each node of “PersonalDetail” to insert into PersonalDetails table.

To execute above stored procedure, call following script

```
DECLARE @xmlData Xml
SET @xmlData = '<PersonalDetails>
  <PersonalDetail>
    <FirstName>XmlFirstName 11</FirstName>
    <LastName>XmlLastName 1</LastName>
    <Age>30</Age>
    <Active>1</Active>
  </PersonalDetail>
  <PersonalDetail>
    <FirstName>XmlFirstName 22</FirstName>
    <LastName>XmlLastName 2</LastName>
    <Age>25</Age>
    <Active>1</Active>
  </PersonalDetail>
</PersonalDetails>'

EXEC InsertPersonalDetailsFromXML @xmlData
```

Note that the @xmlData is of Xml type, here even if we change the variable type to varchar it will work as its string value is a valid Xml.

Above code snippet inserts two records into PersonalDetails database table.



104. How to temporarily hold data into table variable in SQL Server?

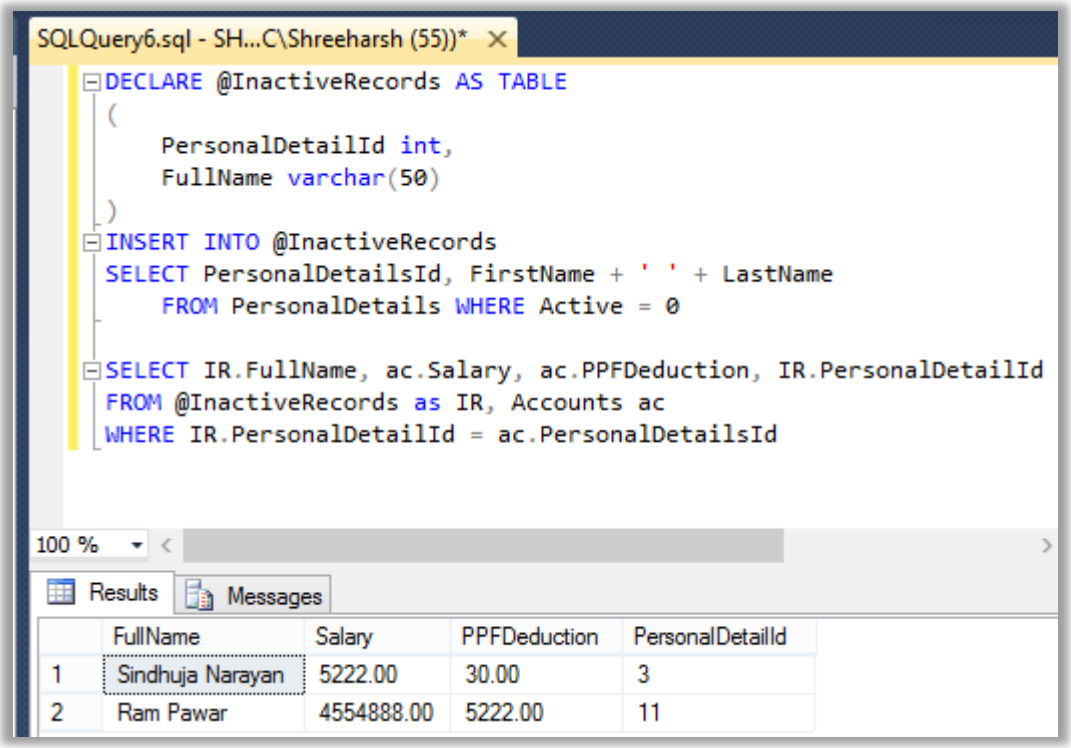
While working with bigger and complex query, we might need to hold some data temporarily to use it or perform some query on. In these scenario, we can use Temporary table.

```
DECLARE @InactiveRecords AS TABLE
(
	PersonalDetailId int,
	FullName varchar(50)
)
INSERT INTO @InactiveRecords
SELECT PersonalDetailsId, FirstName + ' ' + LastName
	FROM PersonalDetails WHERE Active = 0

SELECT IR.FullName, ac.Salary, ac.PPFDeduction, IR.PersonalDetailId
FROM @InactiveRecords as IR, Accounts ac
WHERE IR.PersonalDetailId = ac.PersonalDetailsId
```

In the above code snippet, we have declared a temporary table named "@InactiveRecords" with PersonalDetailsId and FullName column. Next, we are filling this table with inactive data from PersonalDetails table and then using this temp table to join with Accounts and getting other details and listing it.

Important:

It is mandatory to use alias with the temporary table while referencing in the SELECT query.



Remember that as this is a table variable, so the name must be prefixed with '@' character. The scope of this variable exists only within session or stored procedure where it is declared.

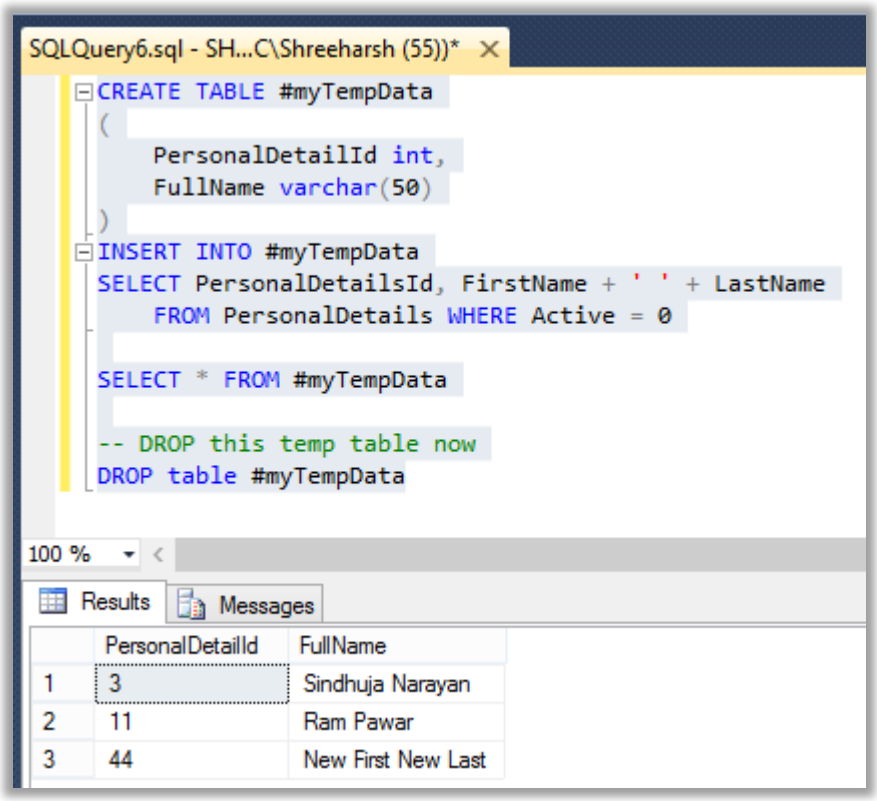
105. How to temporarily hold data into temporary table in SQL Server?

Temporary table is similar to table variable however, temporary table is not created in memory but it gets created physically in the database and it remains unless the current session ends or it is dropped explicitly.

```
CREATE TABLE #myTempData
(
	PersonalDetailId int,
	FullName varchar(50)
)
INSERT INTO #myTempData
SELECT PersonalDetailsId, FirstName + ' ' + LastName
	FROM PersonalDetails WHERE Active = 0

SELECT * FROM #myTempData

-- DROP this temp table now
DROP table #myTempData
```

Temporary table name is prefixed with '#' character.

In above case, we have create a table named "#myTempData' with PersonalDetailId and FullName fields and inserting record from PersonalDetails table then selecting its records and when we are intended to not use this temp table any more, we have dropped it.



Triggers

Triggers are special kind of stored procedure that executes only when the database table is modified using Data Manipulation Language (DML) SQL Statements (Either INSERT, UPDATE or DELETE).

There are following types of triggers

i. AFTER – executes after DML SQL statements
ii. INSTEAD OF – this executes instead of actual DML statement

106. How to create AFTER INSERT triggers in SQL Server?

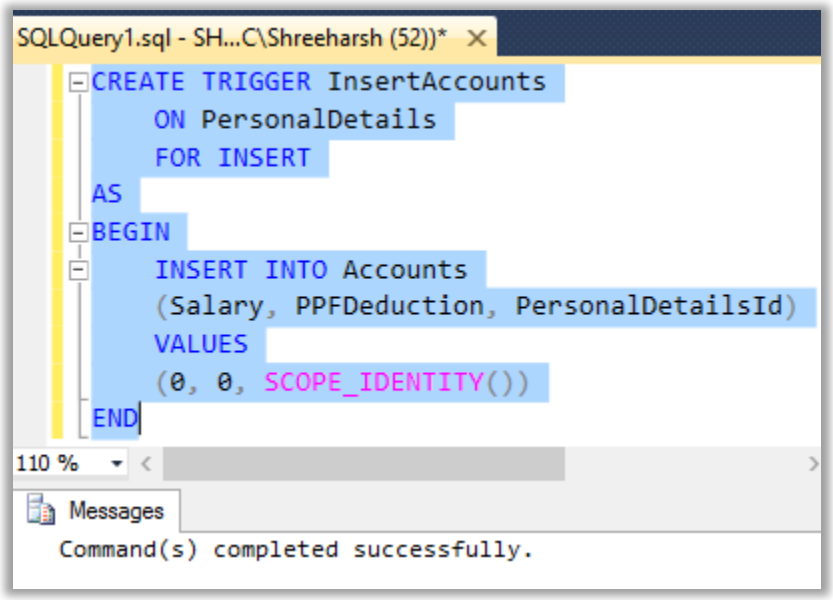
AFTER INSERT trigger executes after a record is inserted into the database. Open a new query window and write below statements

```
CREATE TRIGGER InsertAccounts
       ON PersonalDetails
```

```
        AFTER INSERT
AS
BEGIN
        DECLARE @id int
        SELECT @id = PersonalDetailsId FROM inserted

        INSERT INTO Accounts
        (Salary, PPFDeduction, PersonalDetailsId)
        VALUES
        (0, 0, @id)
END
```

Notice the above code snippet, the name of the trigger is InsertAccounts and created on PersonalDetails table. The type of this trigger is “AFTER INSERT”.

Inside the trigger, we have declared a variable and trying to get the PersonalDetailsId primary key value of the inserted table (in this case PersonalDetials – the highlighted word “inserted” is a logical table that gets created with the record data that was inserted, the scope of this table is limited to the scope of the trigger.) and the same is being used to insert a record into Accounts table as foreign key and default values of Salary and PPFDeduction).



To test, whether this Trigger is working, try to insert a new record into PersonalDetails table.

```
INSERT INTO PersonalDetails
VALUES
('T FirstName 1 ', 'TLastName 1 ', 31, 1)
```

This insert a record into PersonalDetails table and a record into Accounts table (two records).

```
INSERT INTO PersonalDetails
VALUES
('T FirstName 1 ', 'TLastName 1 ', 31, 1)
```

110 %

Messages

```
(1 row(s) affected)

(1 row(s) affected)
```

Result Window

100 %

Results | Messages

	FirstName	LastName	Age	Active	PersonalDetailsId
1	T FirstName 1	TLastName 1	31	1	54

	AutoId	Salary	PPFDeduction	NetSalary	PersonalDetailsId
1	39	0.00	0.00	0.00	54

To delete any trigger created, simply right click the trigger name and choose Delete and click OK on the dialog box. It can also be done using DROP TRIGGER <trigger name> statement running in the query window.

107. How to create AFTER UPDATE triggers in SQL Server?

After update trigger is created in the same way as we created AFTER INSERT trigger, we can simply replace the INSERT word to UPDATE and change respective SQL statements.

```
CREATE TRIGGER [dbo].[UpdateAccounts]
       ON [dbo].[PersonalDetails]
       AFTER UPDATE
AS
BEGIN
       DECLARE @id int
       SELECT @id = PersonalDetailsId FROM inserted

       IF (NOT EXISTS(SELECT AutoId FROM Accounts
                           WHERE PersonalDetailsId = @id))
              BEGIN
                     INSERT INTO Accounts
                     (Salary, PPFDeduction, PersonalDetailsId)
                     VALUES
                     (0, 0, @id)
              END
```

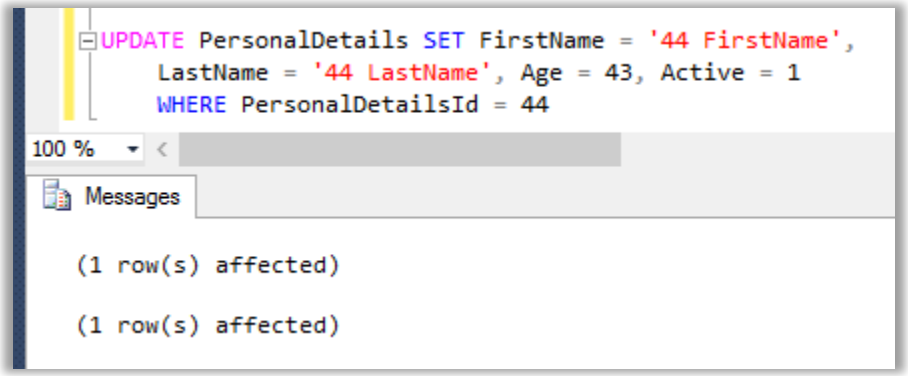
```
END
```

(Few people use FOR instead of AFTER in the above query – notice the highlighted word. Both are same.)

In the above query, we are creating a UpdateAccounts trigger that will execute only when PersonalDetails record will get updated.

In this trigger, we are checking if a corresponding record exists into the foreign key table Accounts for the current record that is being updated in PersonalDetails table. If corresponding record doesn't exists into Accounts table, it adds a record with default value.

To trigger the UpdateAccounts trigger, just created we need to fire an UPDATE statement.



108. How to create AFTER DELETE triggers in SQL Server?

Same as UPDATE, we do delete also. The only difference is in case of INSERT or UPDATE, the logical table name gets created with INSERTED or DELETED record is "inserted" and in case of DELETE trigger the logical table name gets created is "deleted" with the record data that gets deleted.

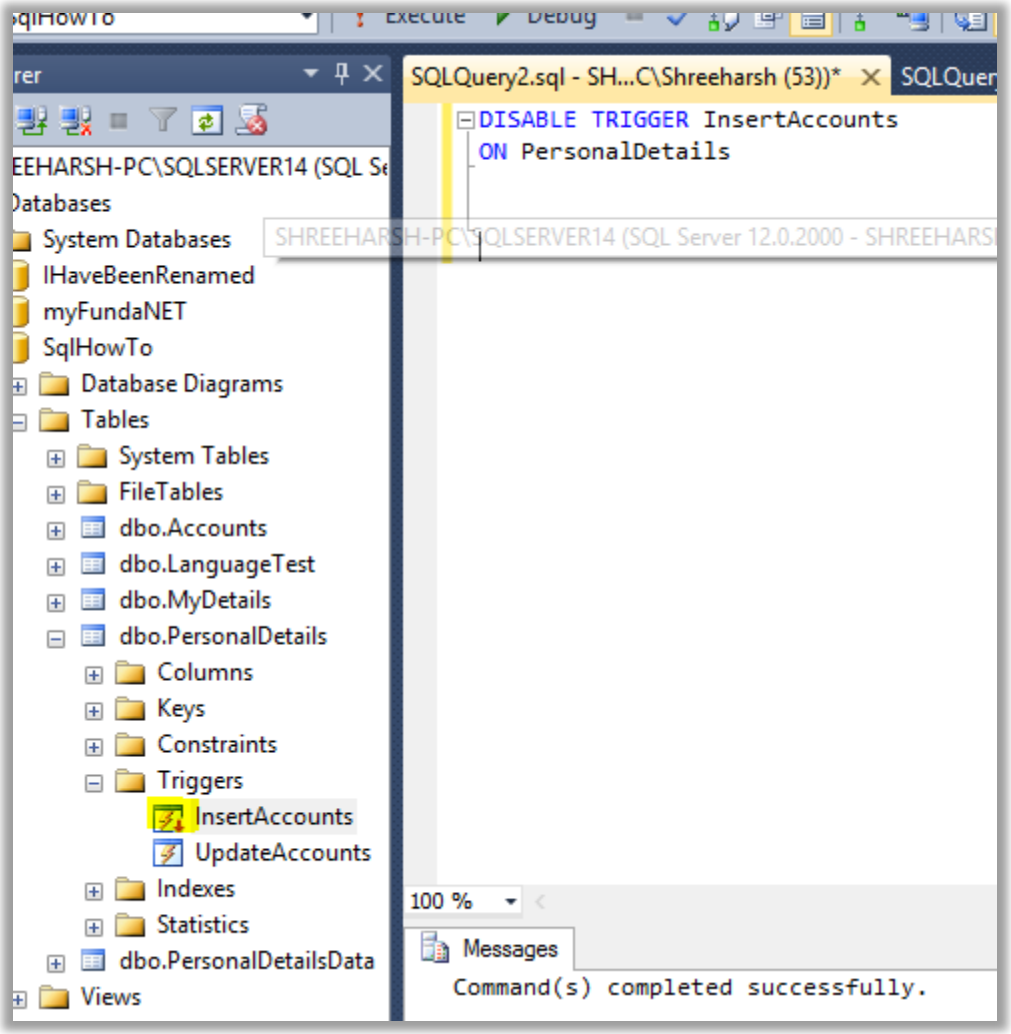
This trigger fires when a record gets deleted from the table on which it is created.

109. How to disable a trigger in SQL Server?

To disable a trigger, open the query window and DISABLE TRIGGER statement and execute it.

```
DISABLE TRIGGER InsertAccounts
ON PersonalDetails
```

This will disable the trigger named "InsertAccounts" that is on PersonalDetails table.



After disabling the trigger, refresh the Triggers folder and see the icon change between this trigger and other triggers.

Now, when a new record is inserted into PersonalDetails table, this trigger won't fire.

110. How to enable trigger in SQL Server?

To enable trigger that was disabled, we use ENABLE TRIGGER statement.

```
ENABLE TRIGGER InsertAccounts
ON PersonalDetails
```

Above lines of code will enable InsertAccounts table created on PersonalDetails table.

Executing above SQL statement will enable the query and inserting a record into PersonalDetails table will start firing InsertAccounts table as well.

111. How to create INSTEAD OF trigger in SQL Server?

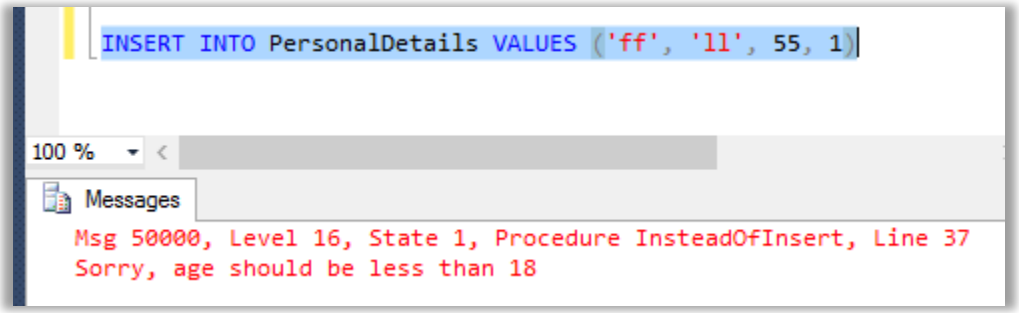
INSTEAD OF trigger executes in place of INSERT/UPDATE/DELETE statements executed against the table.

Open a query window and write below query.

```
CREATE TRIGGER InsteadOfInsert
ON PersonalDetails
INSTEAD OF INSERT
AS
BEGIN
	DECLARE @age int
	SELECT @age = Age FROM inserted

	BEGIN TRY
		IF (@age > 18)
			BEGIN
				RAISERROR ('Sorry, age should be less than 18', 16, 1)
			END
		ELSE
			BEGIN
				INSERT INTO PersonalDetails
				SELECT FirstName, LastName, Age, Active
				FROM inserted
			END
	END TRY
	BEGIN CATCH
		THROW;
	END CATCH
END
```

This will create a InsteadOfInsert trigger that will be called instead of the INSERT statement called on PersonalDetails table. Ie. When we execute INSERT statement on PersonalDetails table, in place of that this trigger will be called.

In above case, we are getting the Age of the record being inserted from the logical table (the data that is being set into the standard INSERT statement) and the same @age is being checked, if the age is greater than 18, it raise error otherwise inserts the record into the database with the help of "inserted" logical table.



Thanks for reading.

Important request: If you are facing any other problems in SQL Server that have not been covered here, do let us know so that we can solve it and update this eBook to make it more useful.

Visit

http://www.ITFunda.com

for more study materials, online training and job support related with .NET Technologies.

For any feedback, contact us

Phone: +91-40-4222-2291

Mobile: +91-768-088-9888

Email: support@itfunda.com

Website: http://www.ITFunda.com

Skype: FundaSupport