Lab Manual

CS614 – Data Warehousing

Lab 1

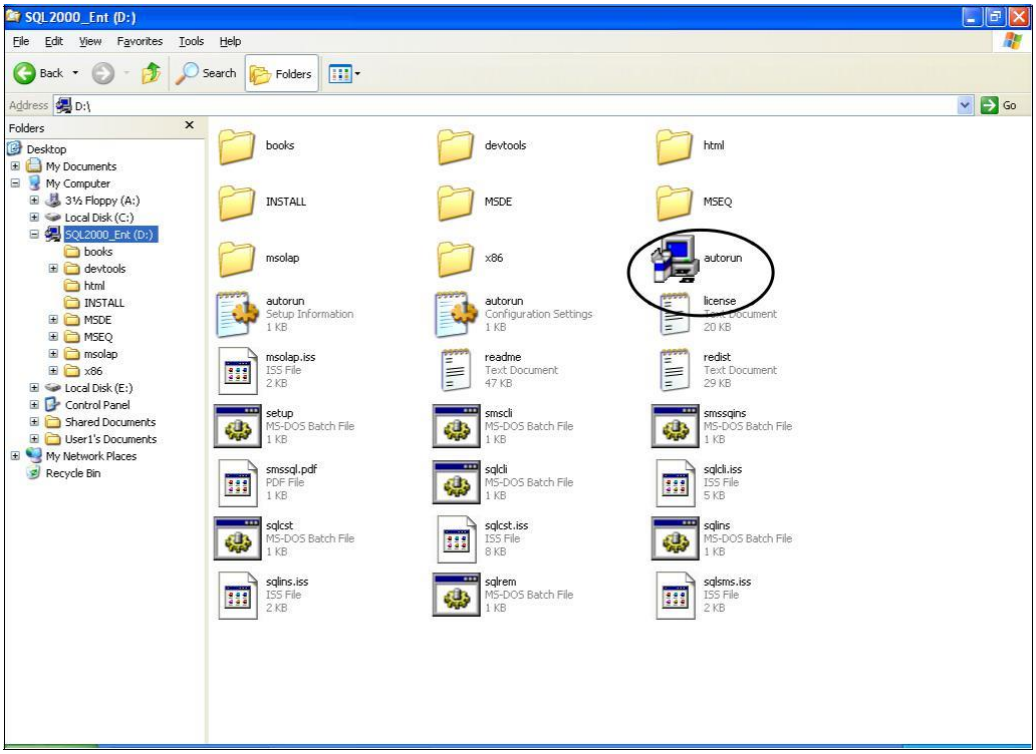
Installation of Microsoft SQL Server 2000

Important Instruction

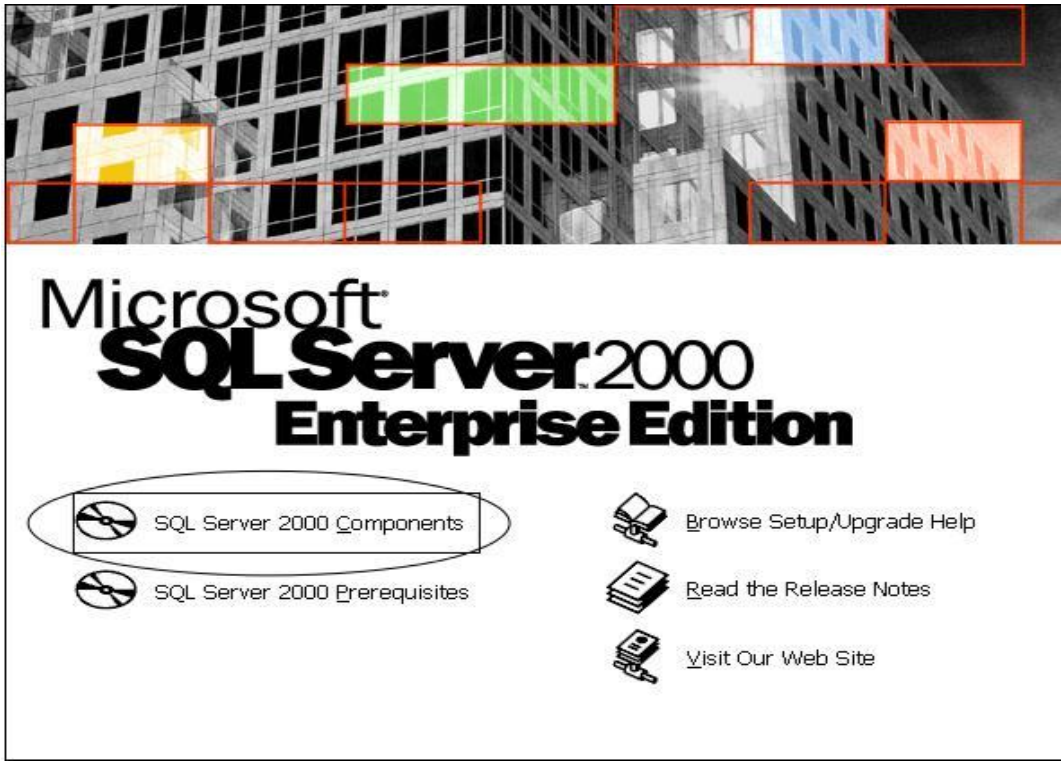
- ✓ User must have Administrators permissions in order to install Windows SQL Server 2000.
- ✓ Windows Server 2000 with SP3 or higher must be installed on the system.
- ✓ Close all programs running on computer before installation of SQL Server 2000.

Installation of SQL Server 2000 Step by Step:

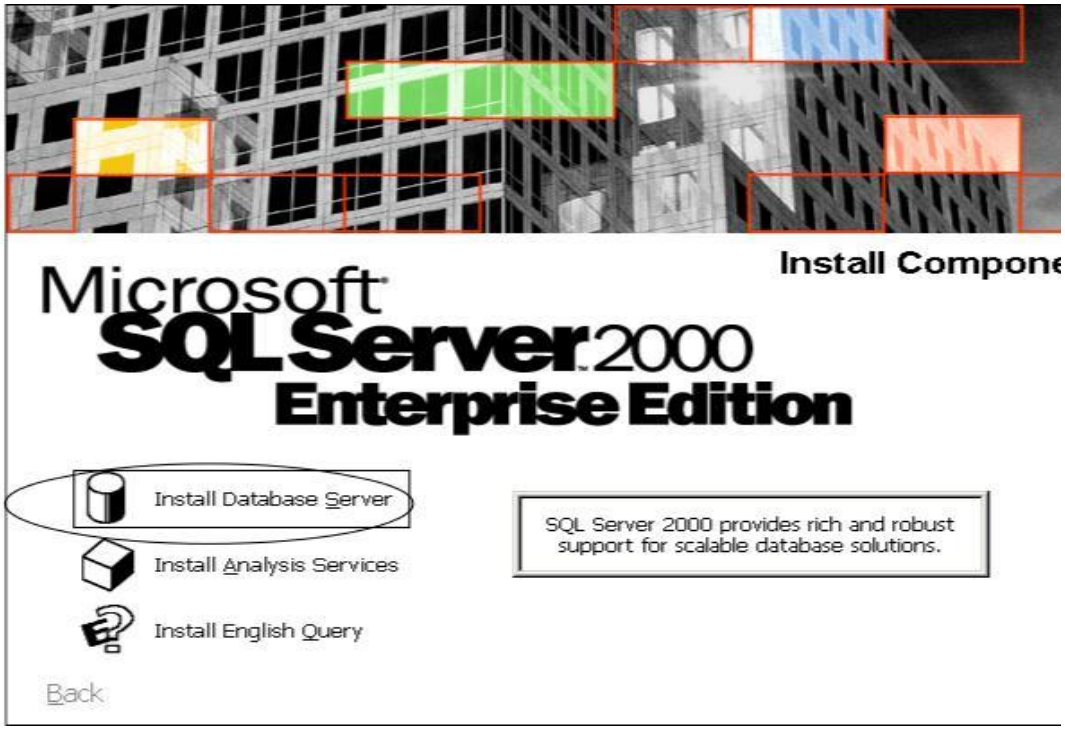
1. Open the folder/Drive containing set up of SQL Server 2000 and double click on “autorun”.



2. At the Microsoft SQL Server 2000 Enterprise Edition screen, press the “SQL Server 2000 Components” button.

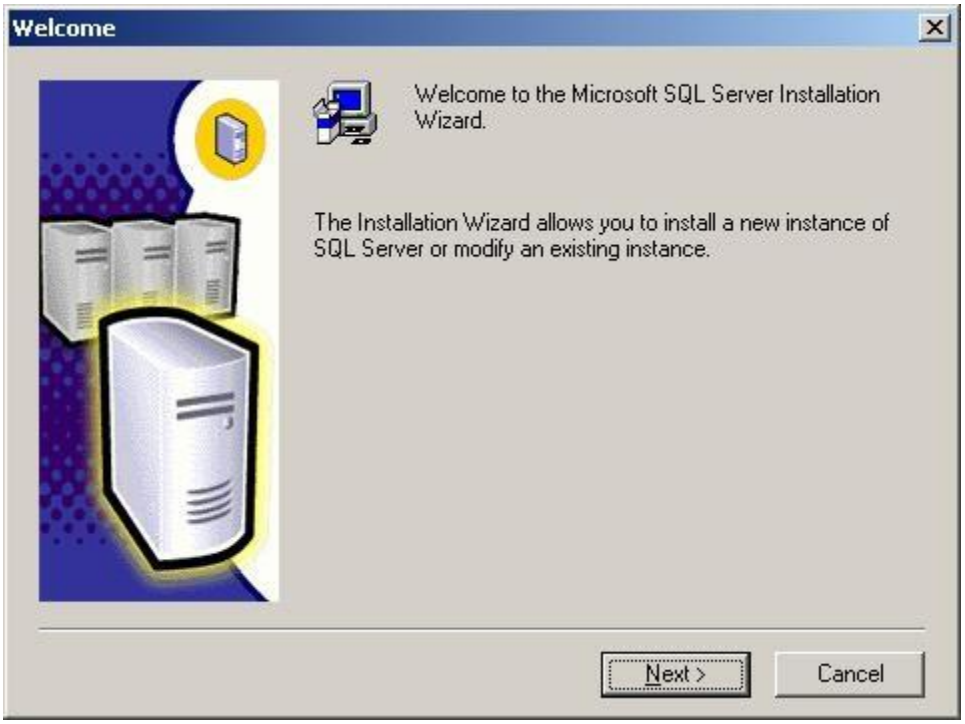


3. At the Microsoft SQL Server 2000 Enterprise Edition screen, press the “Install Database Server” button



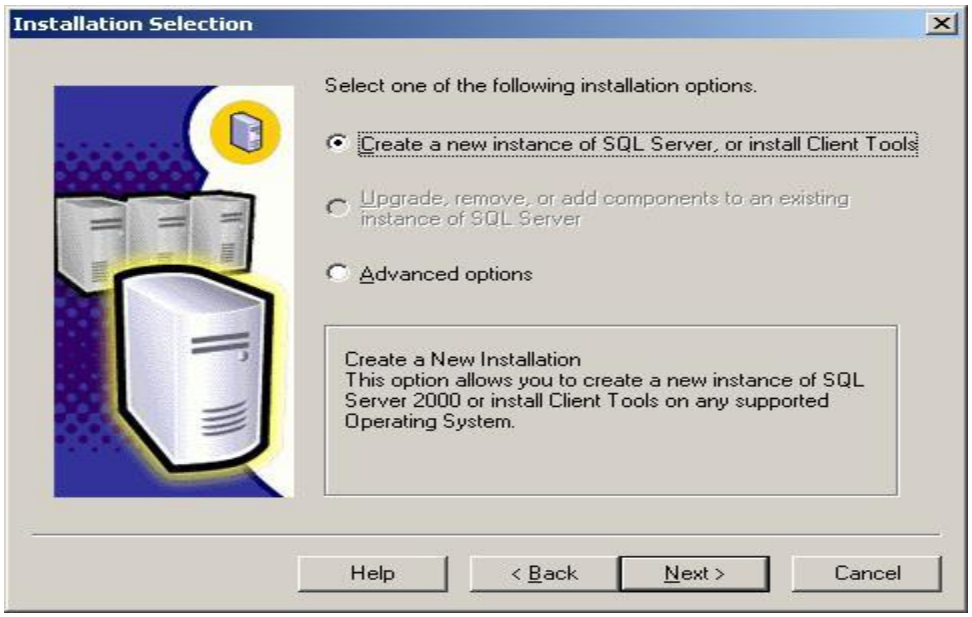
4. If Windows Server 2003 is installed you will see the SQL Server 2000 window. Press the Continue button.

5. You will be brought to the Welcome window to the Microsoft SQL Server Installation Wizard. Here press the Next button.



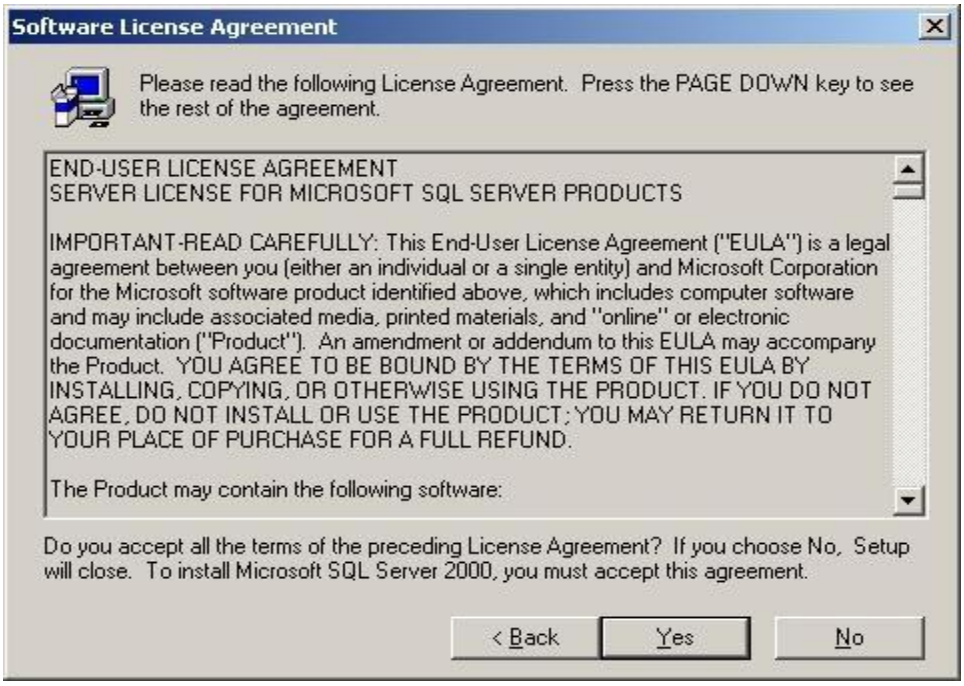
6. You will be brought to the Computer Name window. Since you are installing SQL 2000 Server on the Server computer, you will select the Local Computer and press the Next button.

7. You will be brought to the Installation Selection window. Select the “Create a new instance of SQL Server, or install Client Tools" radio button and press Next button.

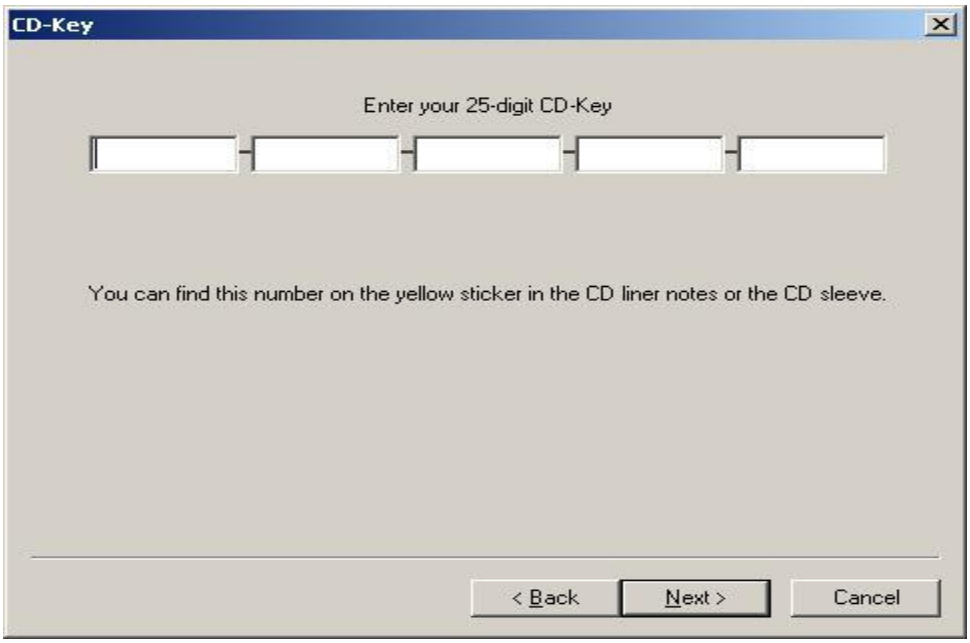


8. User will be brought to the User Information window. Enter the computer Name and Company for your system, press Next button.

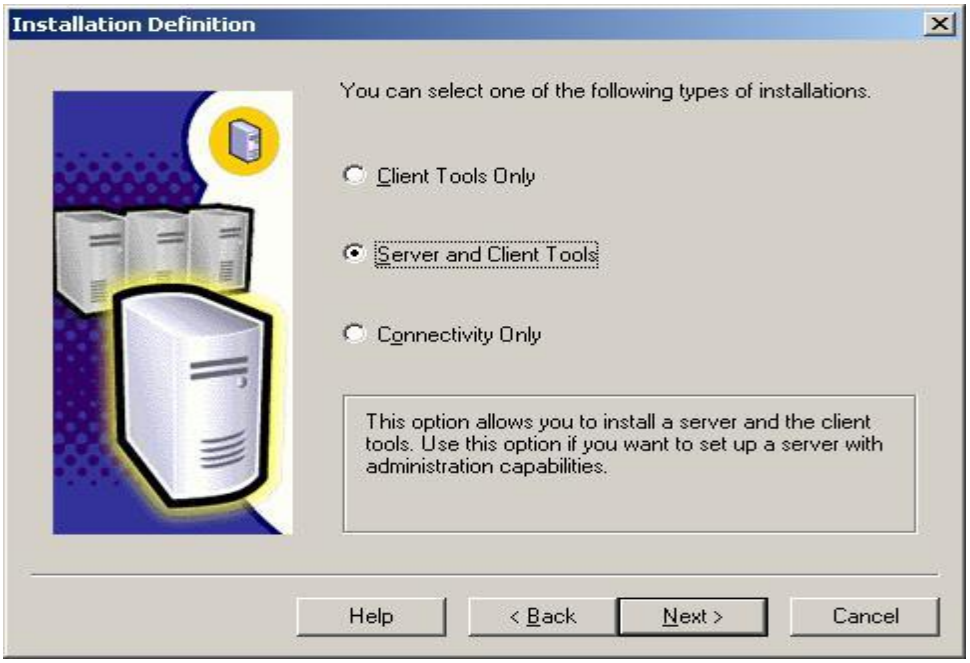
9. You will be brought to the Software License Agreement window. After reading the legal agreement, press the Yes button.



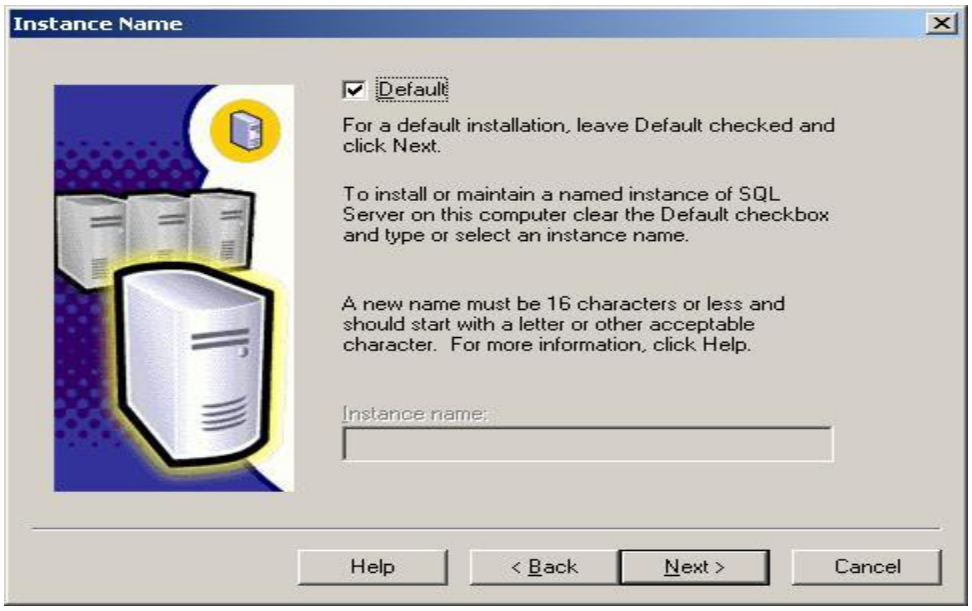
10. You will be brought to the registration key window. Enter the 25 digit registration Key and press the Next button.



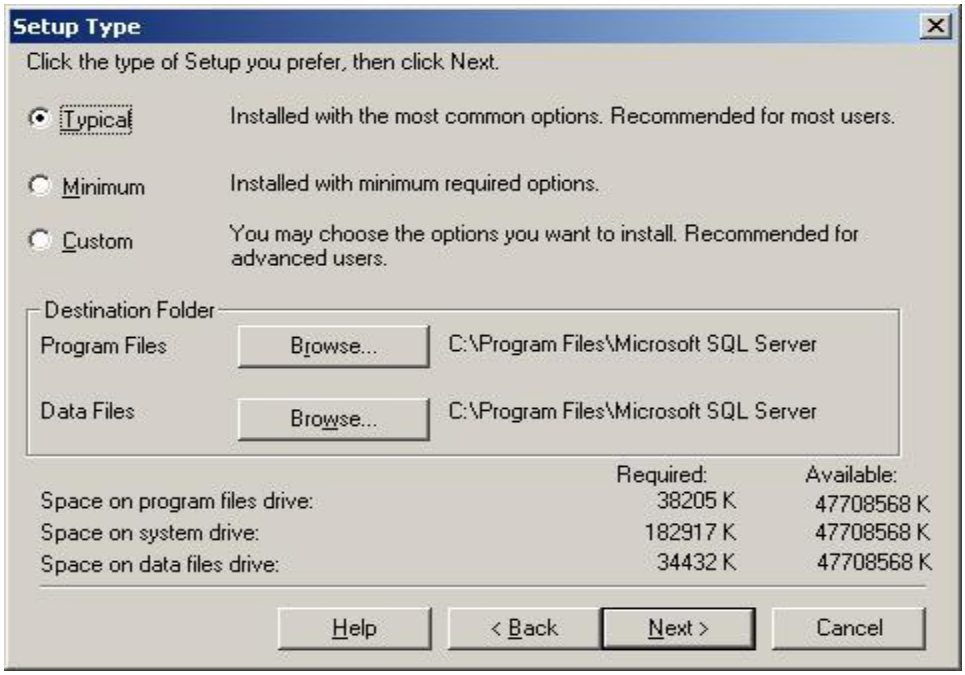
11. You will be brought to the Installation Definition window. Choose the "Serve and Client Tools" radio button and press the Next button.



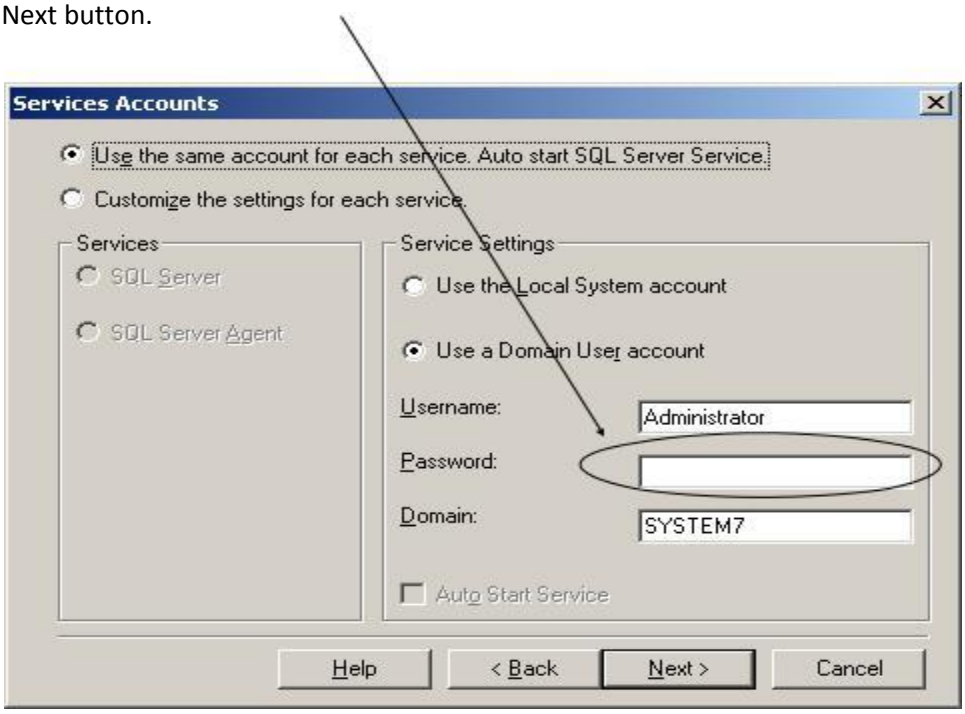
12. You will be brought to the Instance Name window. Her you have to press the Next button.



13. You will be brought to the Setup Type window. Of the type of Setup preferred, select Typical. And press the Next button.

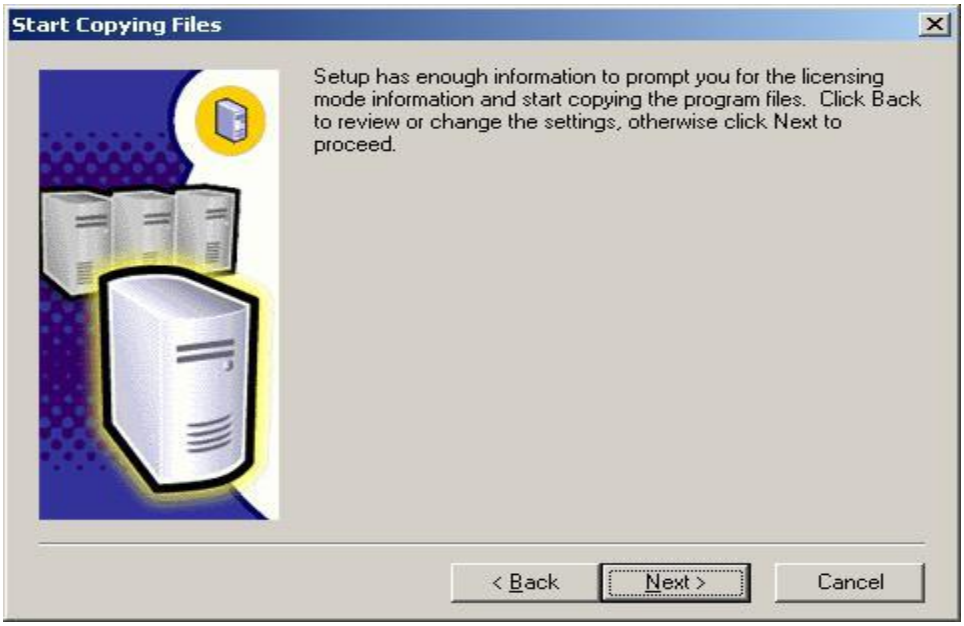


14. You will be brought to the Service Accounts window. Choose the ‘Use the same account for each service Auto start SQL Server Service.’ radio button. Enter Administrator’s Password and press the Next button.

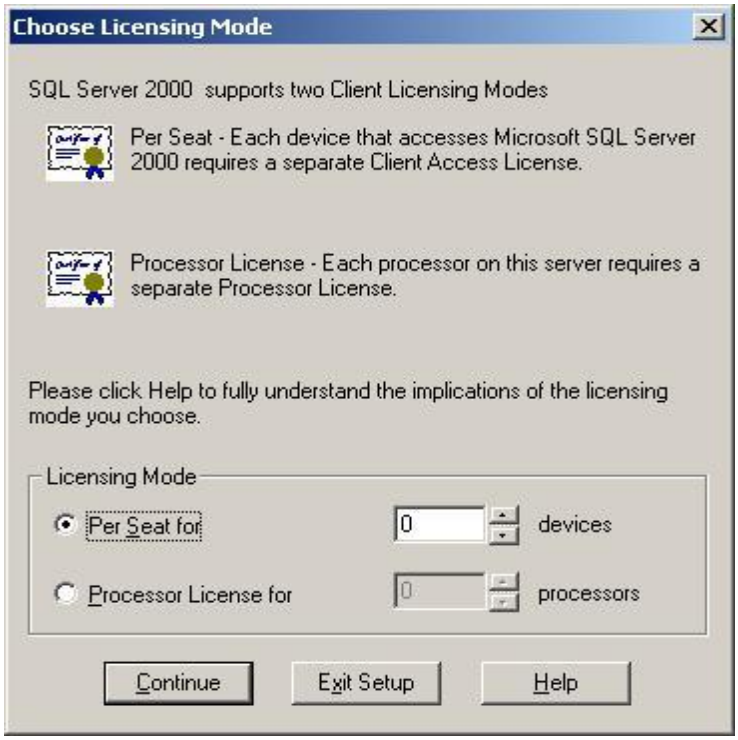


15. You will be brought to the Authentication Mode window. Select the ‘Mixed Mode (Windows Authentication and SQL Server Authentication) radio button. Enter your password in the Enter Password and Confirm Password fields and press the Next button.

16. You will be brought to the Start Copying Files window and press the Next button.



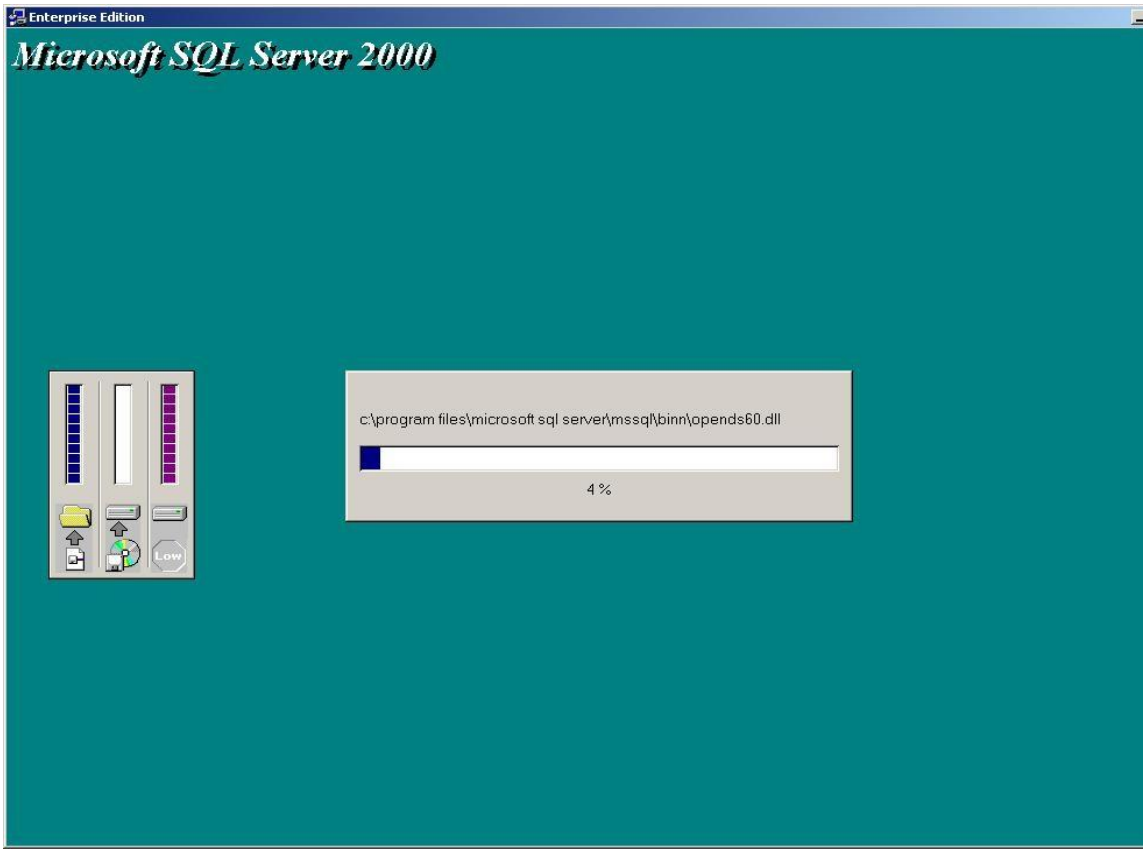
17. Once the file installation is complete, you will be brought to the Choose Licensing Mode. Select either Per Seat or Processor License. Press the Continue button.



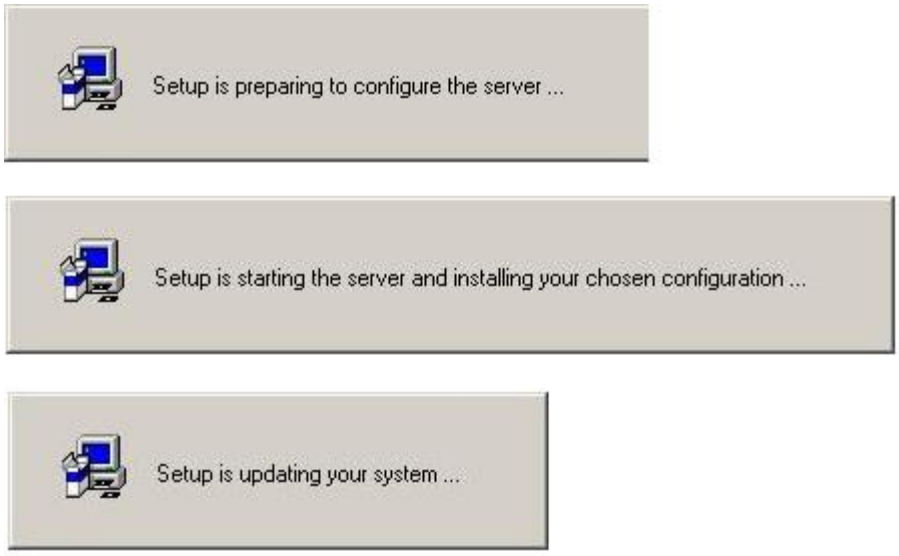
18. The System will display this window when loading the system.

19. The system will display this window when installing the Microsoft Full-Text Search Engine.

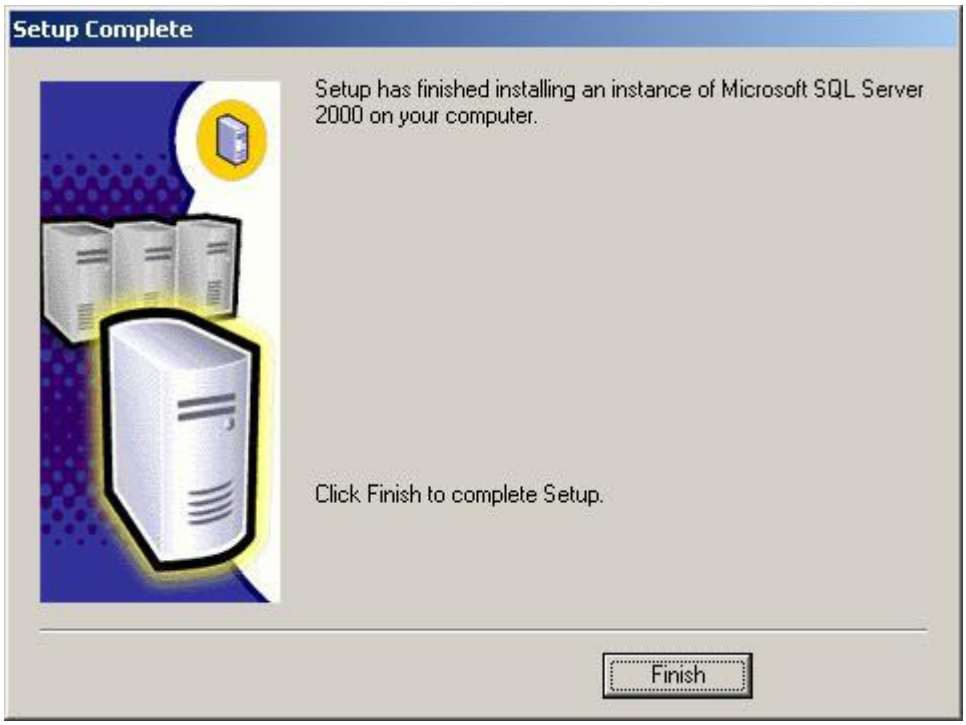
20. SQL Server 2000 progress screen will display.



21. Setup is preparing to configure the server and setup started the server and installation of selected configuration. After that setup updating window will display.

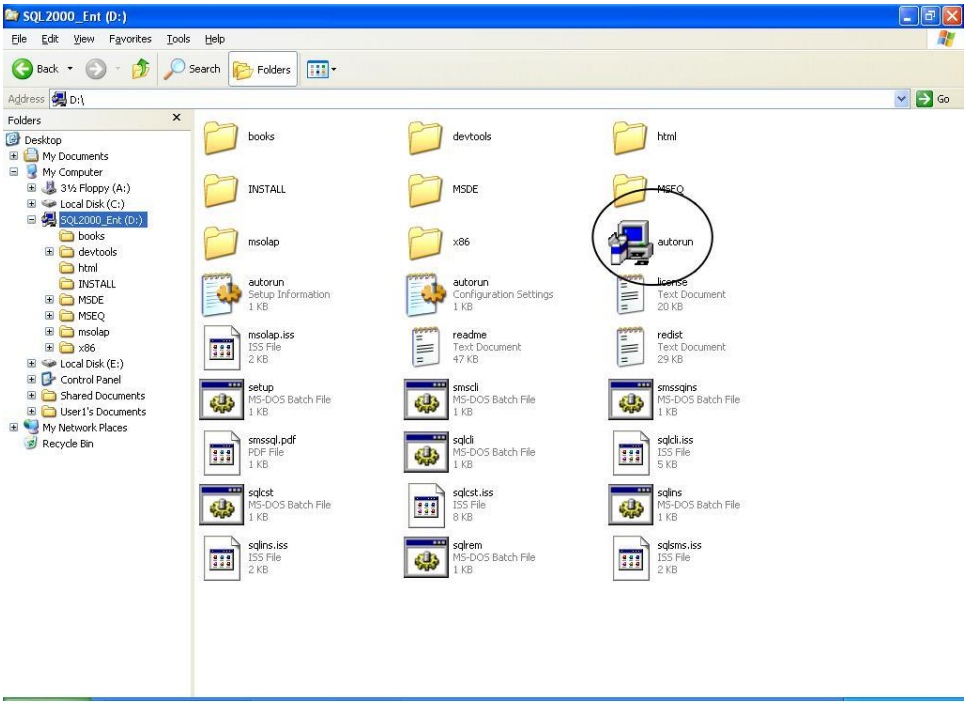


22. Once the setup is complete, you will be brought to the following window. Press the **Finish** button to complete the installation.

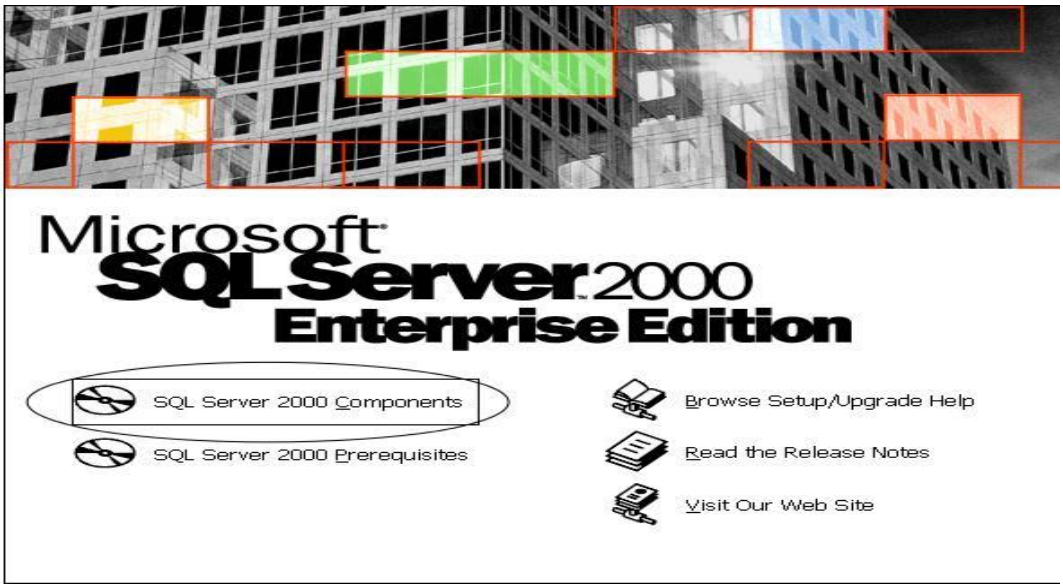


Installation of SQL Server 2000 Client:

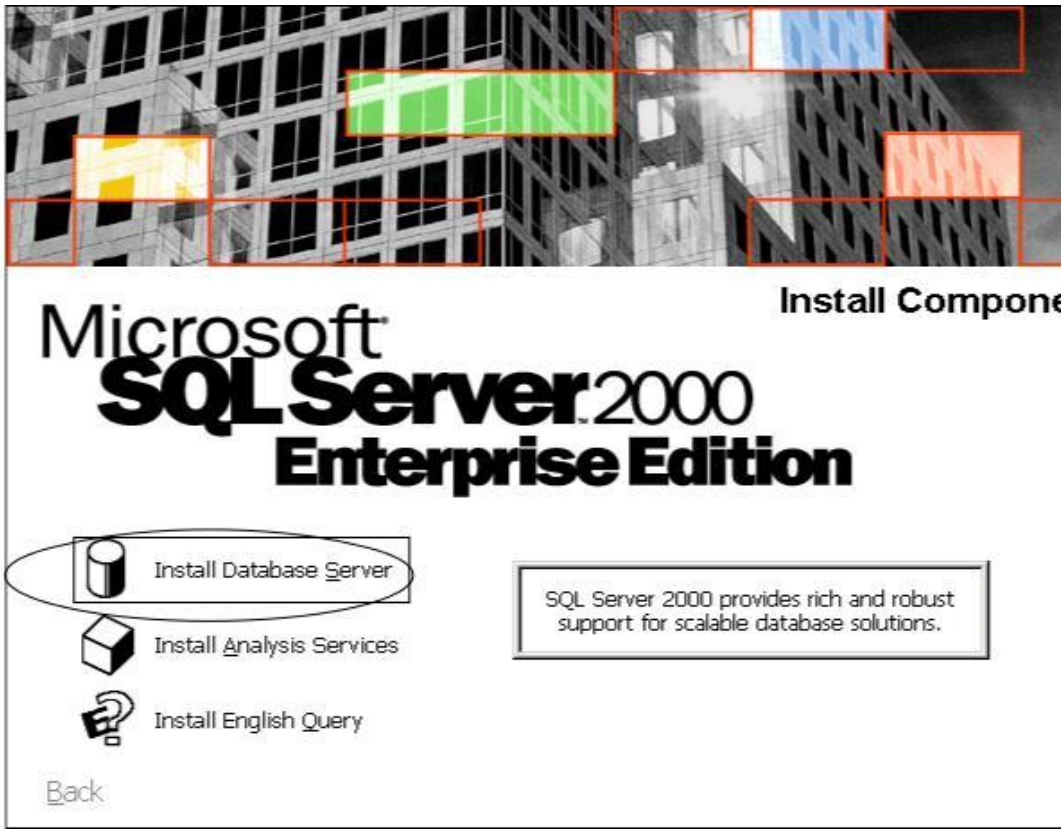
1. Open the Drive/Folder and click on “autorun”.



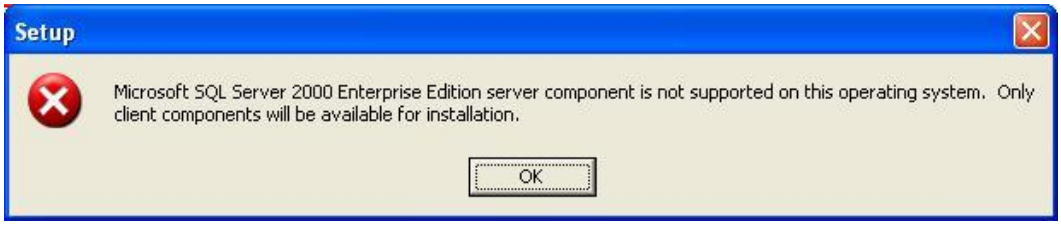
2. At the Microsoft SQL Server 2000 Enterprise Edition screen, press the SQL Server 2000 Components button.



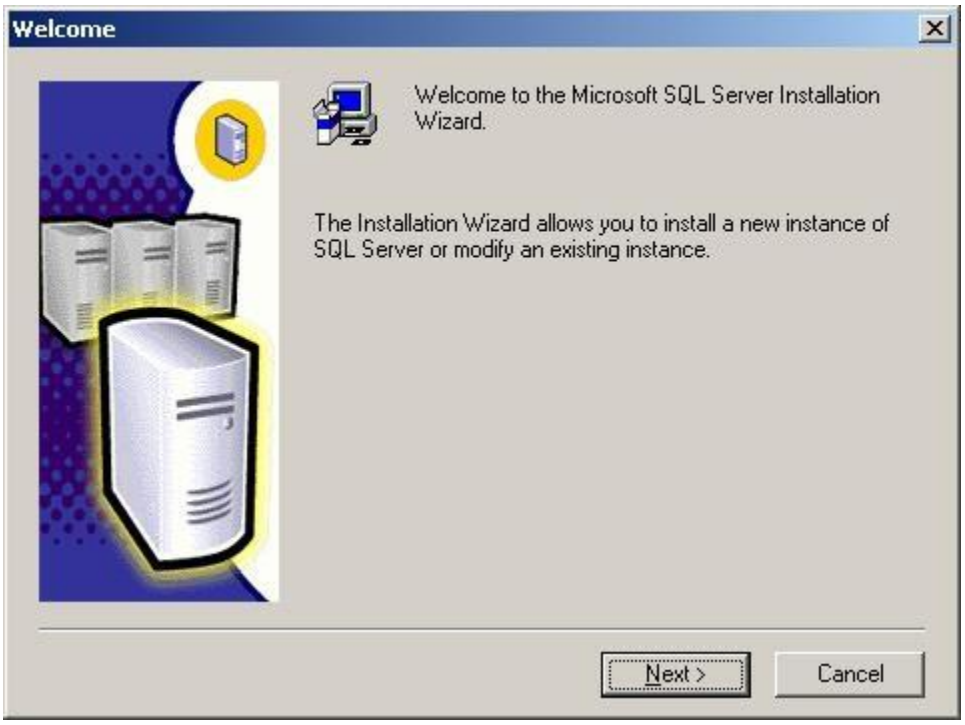
3. At the Microsoft SQL Server 2000 – Enterprise Edition – Install Components screen, press the Install Database Server button.



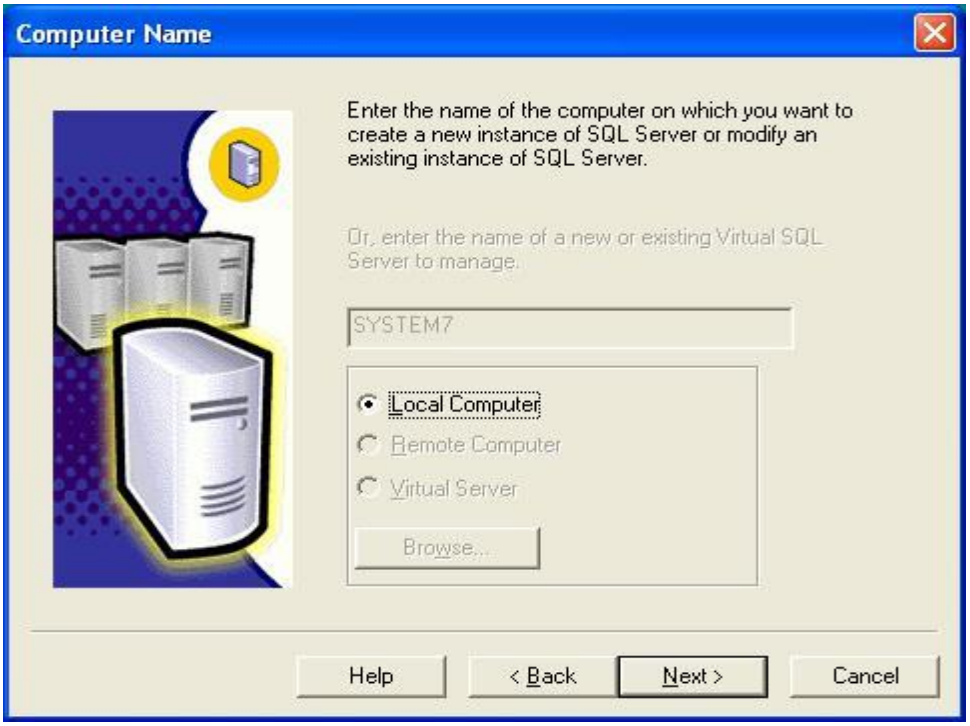
4. If Windows Server 2003 is installed, you will see the Setup window. Press the OK button.



5. You will be brought to the Welcome window to the Microsoft SQL Server Installation Wizard. Press the Next button.

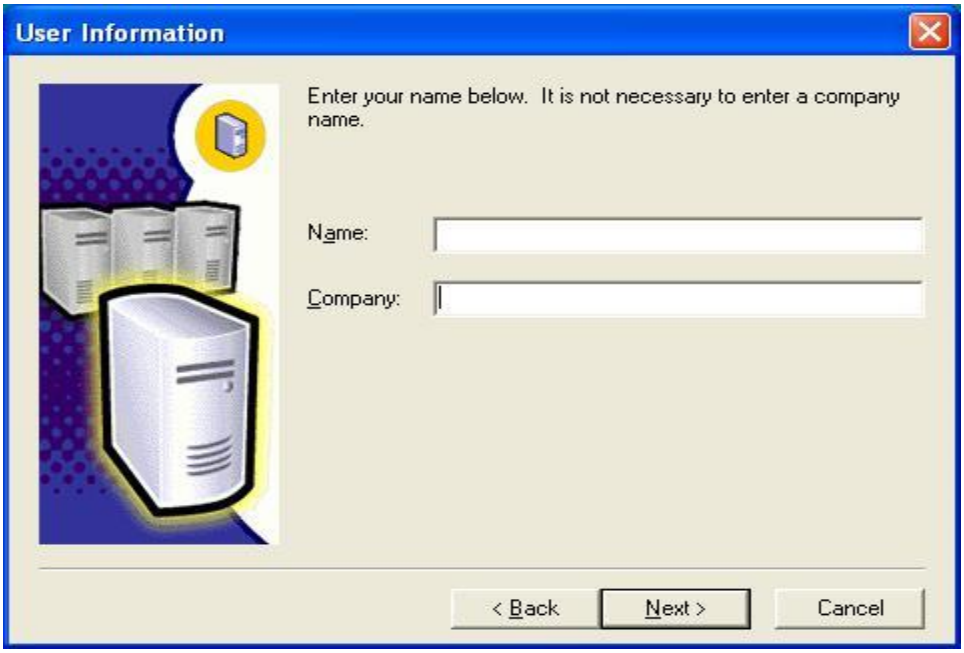


6. You will be brought to the Computer Name window. Since you are installing SQL 2000 Client, the Local Computer option will be the only selection. Press the Next button.

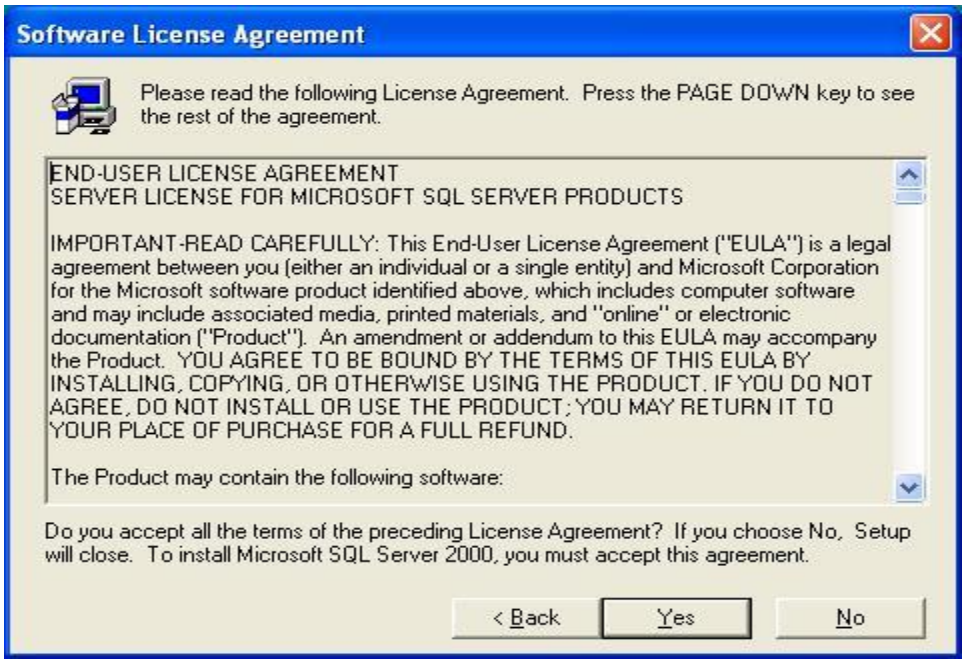


7. You will be brought to the Installation Selection window. The “Create a new instance of SQL Server, or install Client Tools" will be the only option available. Press the Next button.

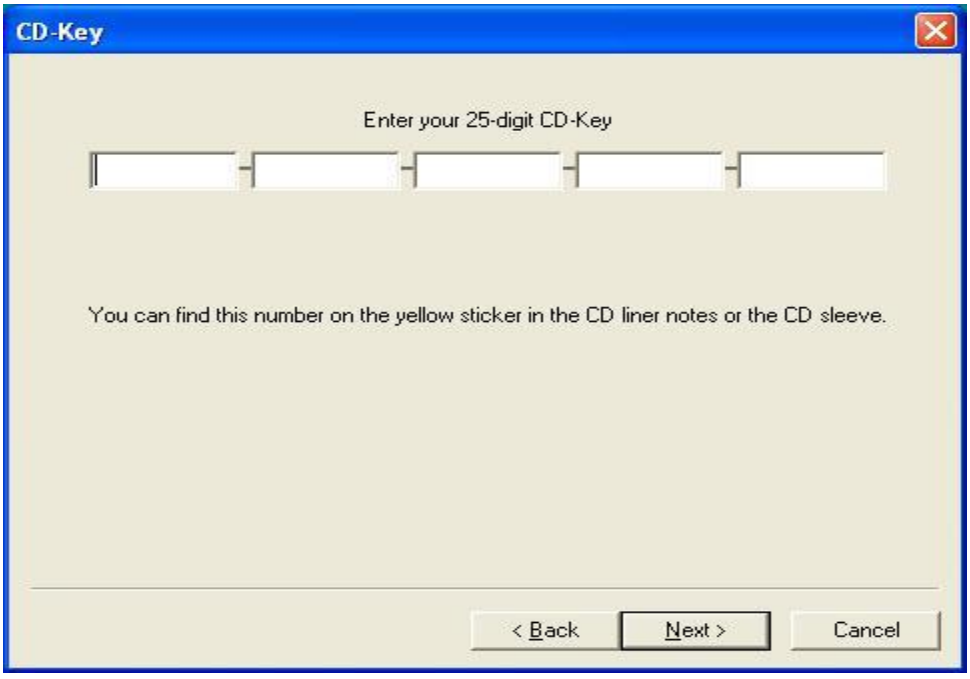
8. You will brought to the User Information window. Enter the computer Name and Company for your system. Press the Next button.



9. You will be brought to the Software License Agreement. After reading the legal agreement, press the Yes button.

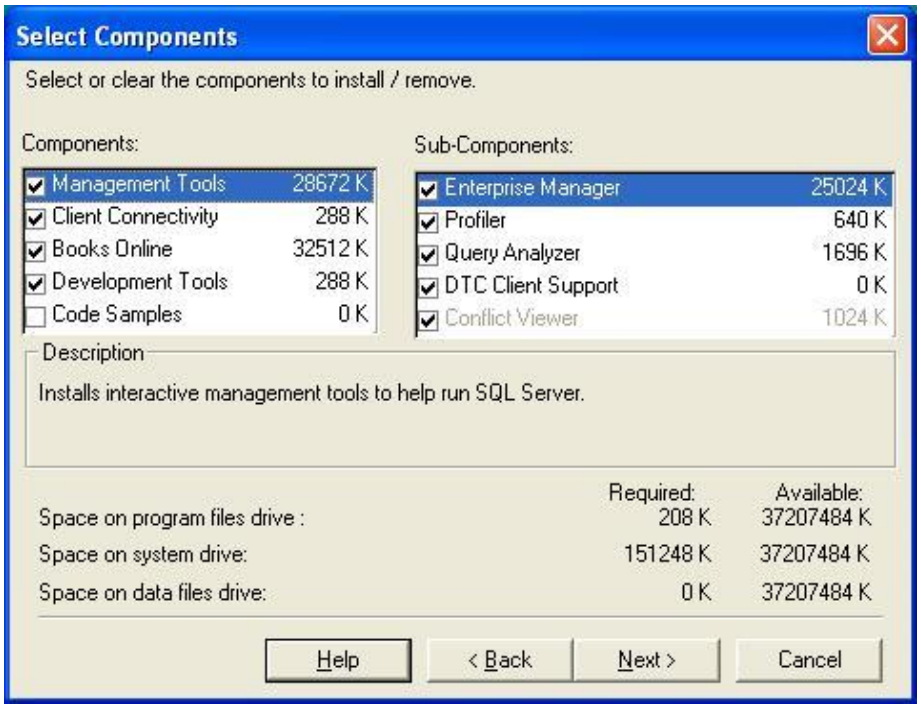


10. You will be brought to the Registration Key window. Enter the 25 digit CD-Key. Click on the Next button.

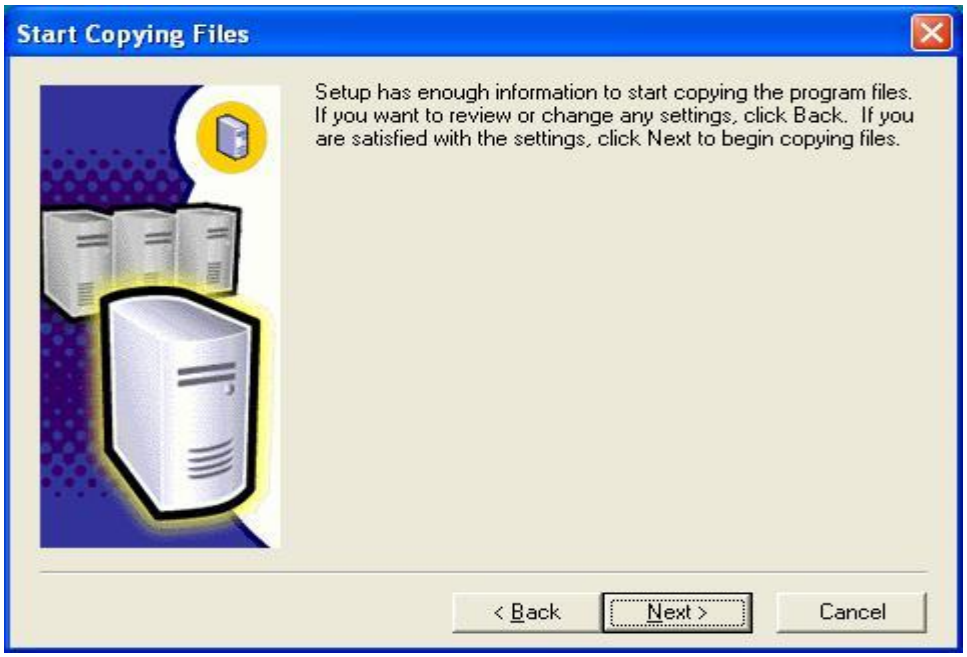


11. You will be brought to the Installation Definition window. Choose the "Client Tools Only" and then press the Next button.

12. You will be brought to the Select Components window– press on the Next button for default values.

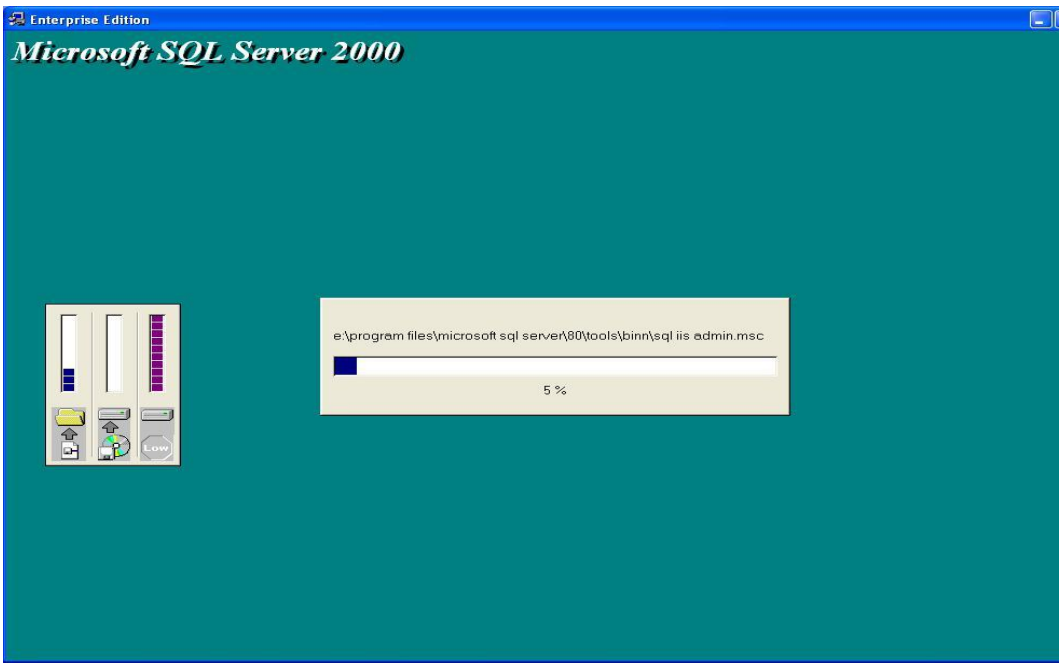


13. At the "Start Copying Files" window, press the Next button.



14. The system will display this window when loading the system.

15. SQL Server 2000 progress screen will display.



16. Once the setup is done, you will see the following window. Press the Finish button to complete the installation.

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 2

NORMALIZATION

Consider the following table

Project ID	Project Name	Project Budget	EmpID	Emp Name	Hourly Rate
1001	Pakistan International Airlines Database	1 billion	101 102 103	Sana Ali Hasan	60000 80000 45000
1002	NADRA database	20 million	111 112	Amir Umer	90000 80000

Following functional dependences exist.

1. ProjectID->ProjectName, Project Budget
2. EmpID->EmpoyeeName, HourlyRate
3. ProjectID, EmpID->ProjectName, Project Budget , EmpName, HourlyRate

Normalize above given table into first and second normal form.

Solution:

First Normal Form: Remove repeating groups

Project ID	Project Name	Project Budget	EmpID	Emp Name	Hourly Rate
1001	Pakistan International Airlines Database	1 billion	101	Sana	60000
1001	Pakistan International Airlines Database	1 billion	102	Ali	80000
1001	Pakistan International Airlines Database	1 billion	103	Hasan	45000
1002	NADRA database	20 million	111	Amir	90000
1002	NADRA database	20 million	112	Umer	80000

Second Normal Form: Remove Partial Dependencies

The above table is not in the second normal form since there exists the partial dependency through the FDs 1, 2 and 3. To bring it into second normal form, we will decompose the table into the following tables:

Employee Table

EmpID	Emp Name	Hourly Rate
101	Sana	60000
102	Ali	80000
103	Hasan	45000
111	Amir	90000
112	Umer	80000

Project Table

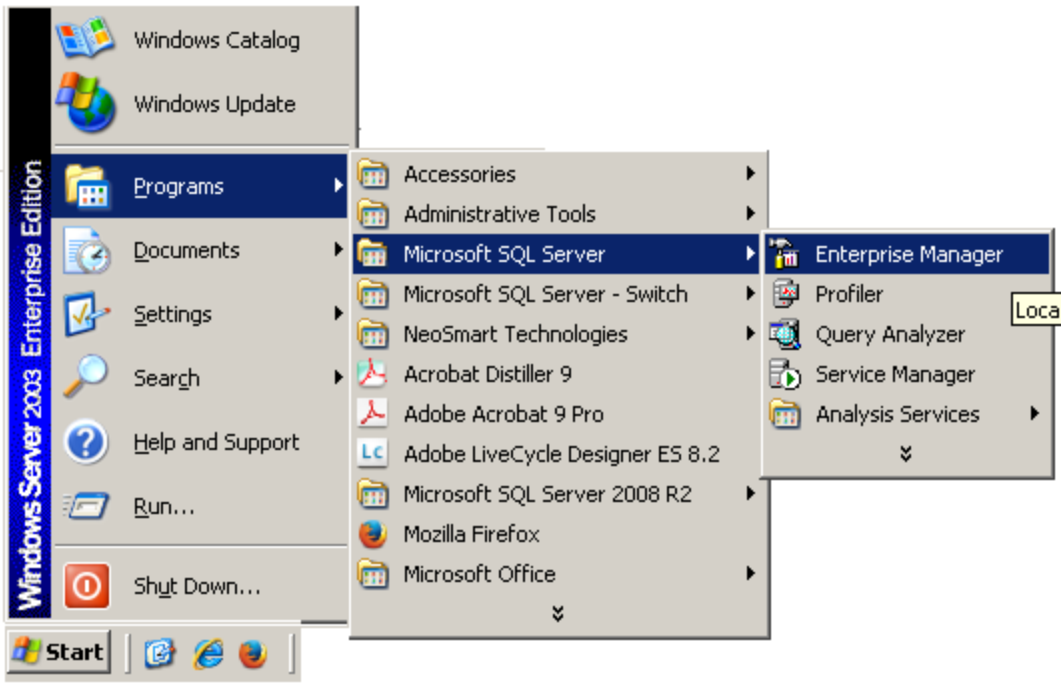
Project ID	Project Name	Project Budget
1001	Pakistan International Airlines Database	1 billion
1001	Pakistan International Airlines Database	1 billion
1001	Pakistan International Airlines Database	1 billion
1002	NADRA database	20 million
1002	NADRA database	20 million

Employee-Project Table

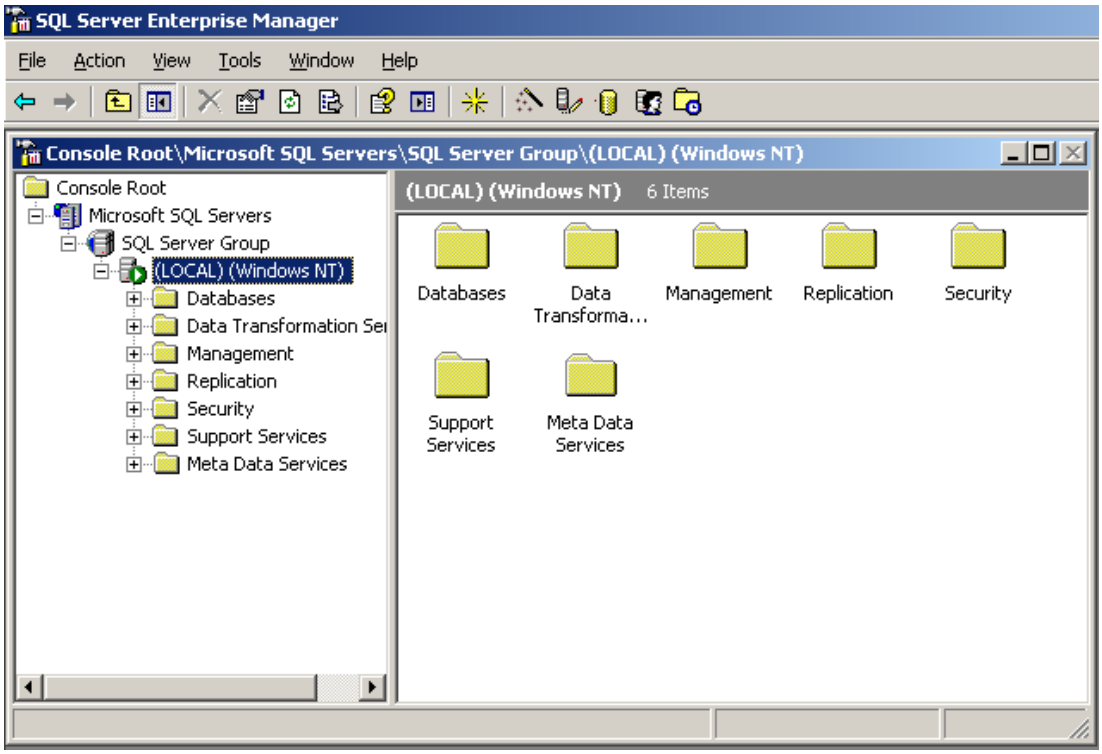
Project ID	EmpID
1001	101
1001	102
1001	103
1002	111
1002	112

Now, above tables are in second normal form. Create above all tables in SQL server enterprise manager in normalized form. The procedure of table creation is given below.

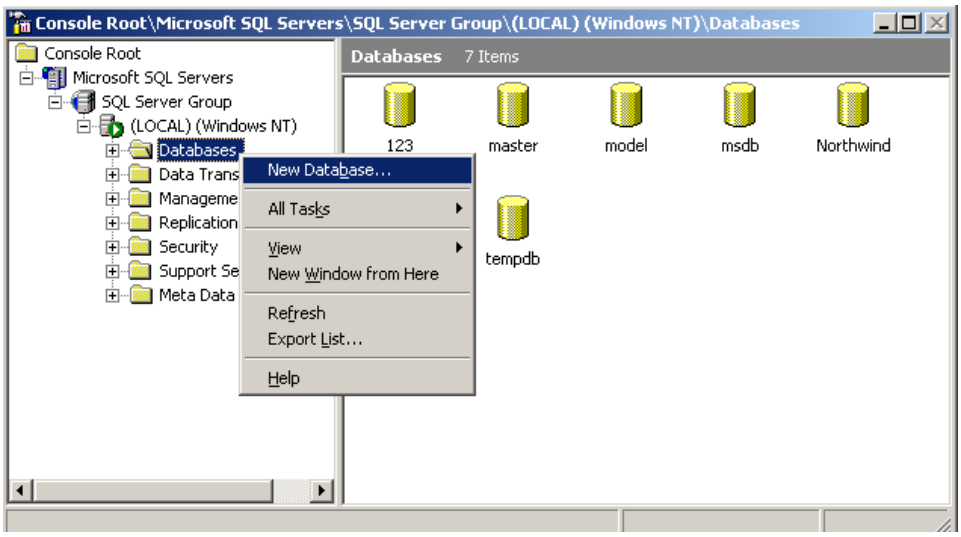
1. Open SQL Server Enterprise Manager by clicking on **Start menu->Programs->Microsoft SQL Server->Enterprise Manager**.



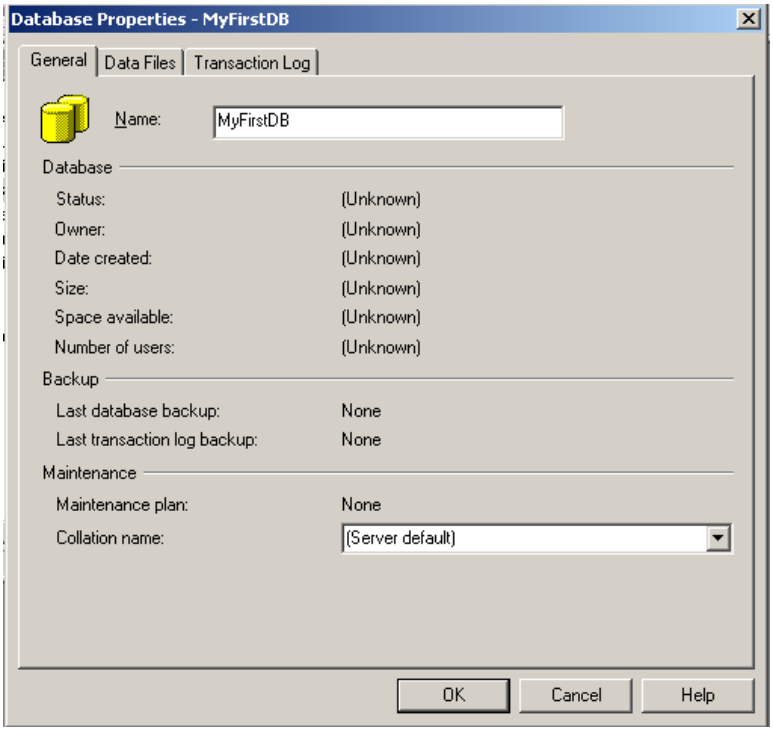
2. Click server node



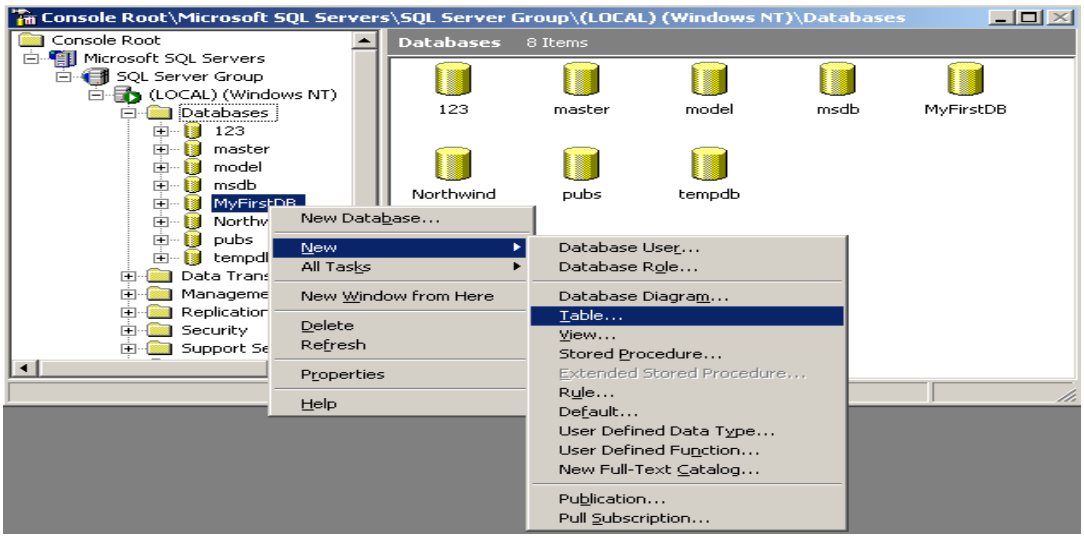
3. Right Click on **Databases** node and click on **New Database…**



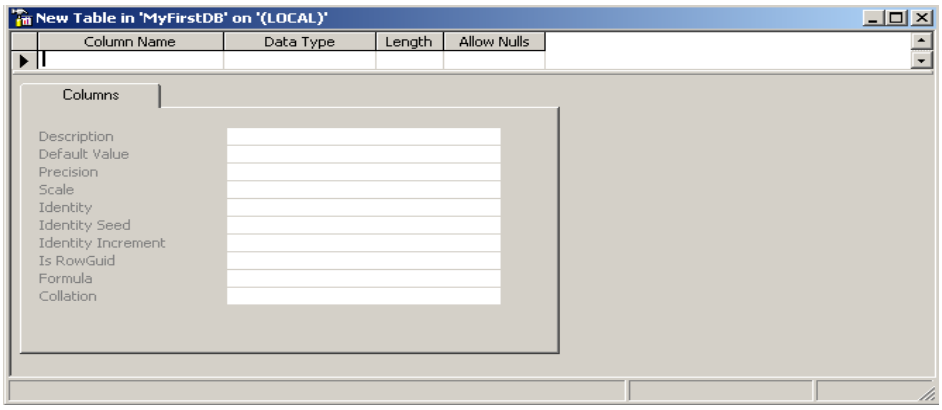
4. Type MyFirstDB in **Name** as Database Name and click oK.



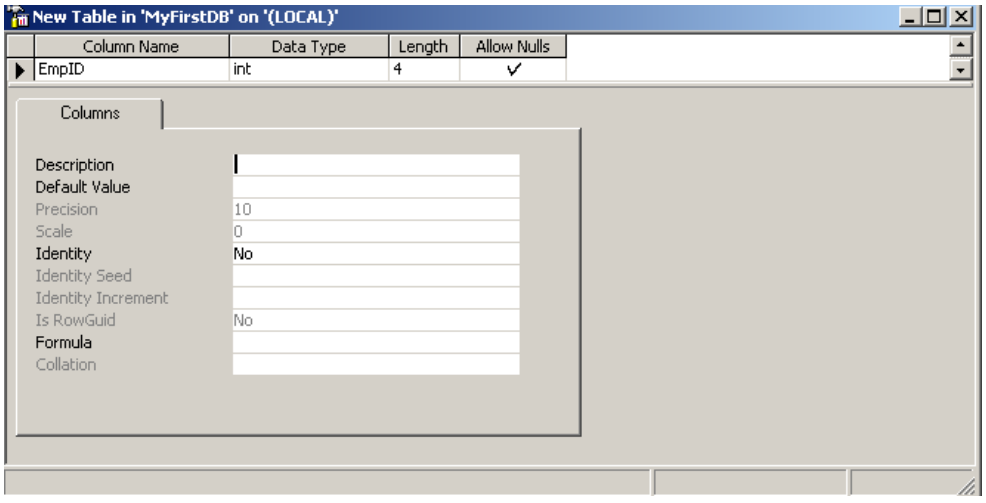
5. Expand **Databases** node, right click on your newly created databases **MyFirstDB,** click on **New** and then click on **Table**.



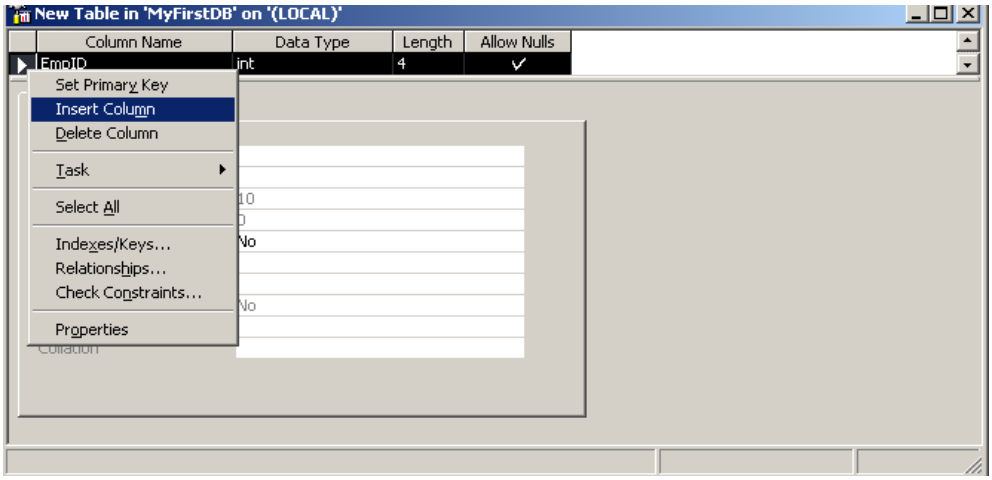
6. The following window will open.



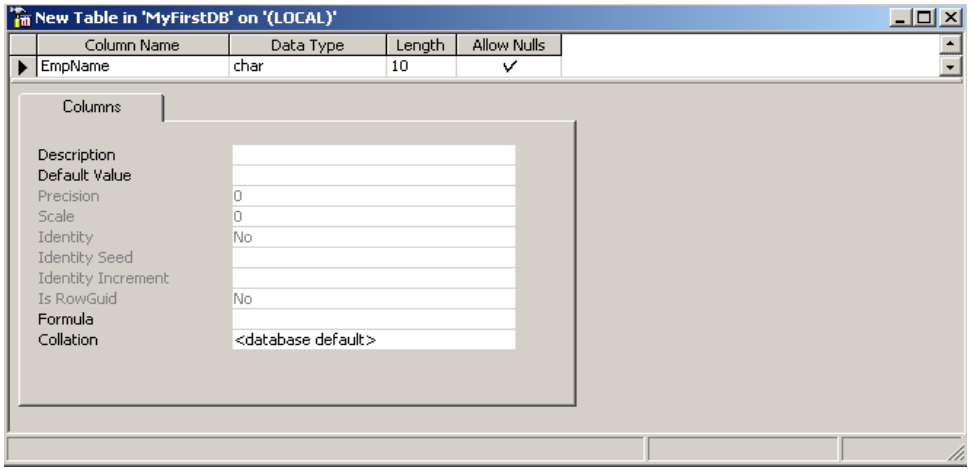
7. Type Column Name and select data type. See following figure in which we have created variable EmpID.

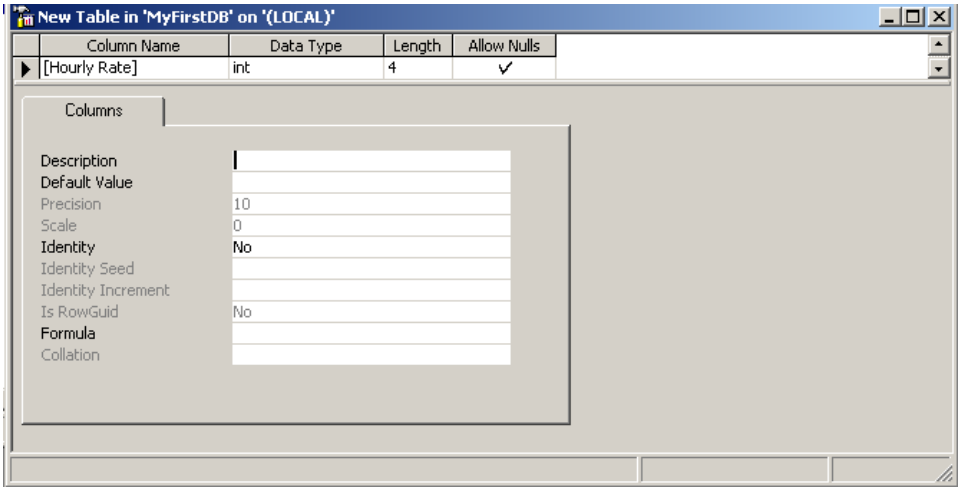


8. Right Click on column row and select option **Insert Column**.

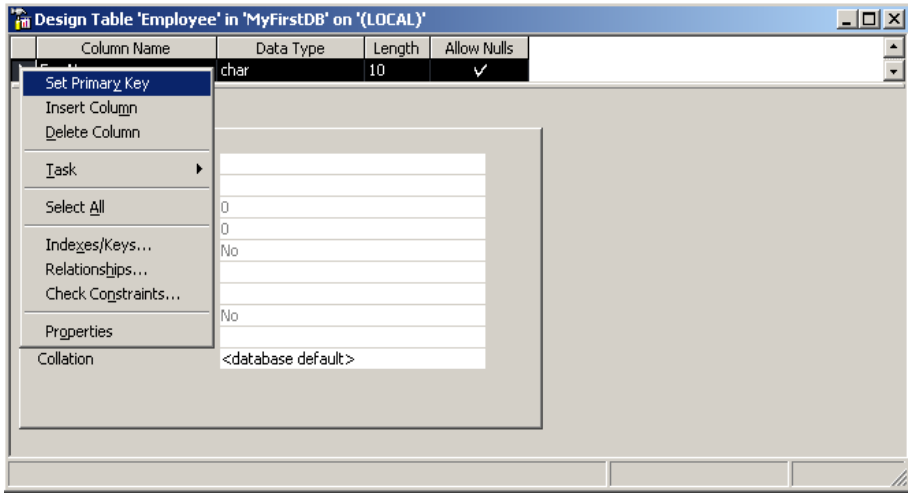


9. Similarly create other variables, EmpName and Hourly Rate

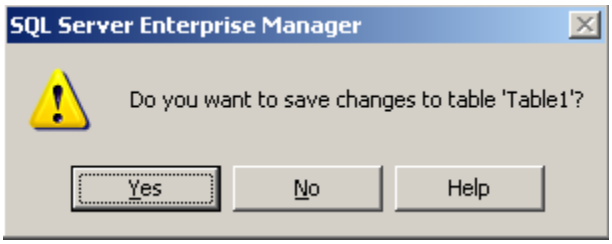




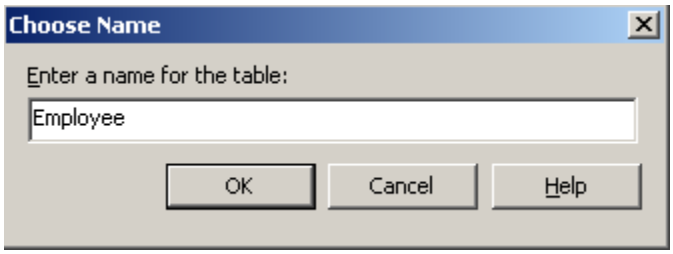
10. To set a column as primary key, right click on column name and click **Set Primary Key**.



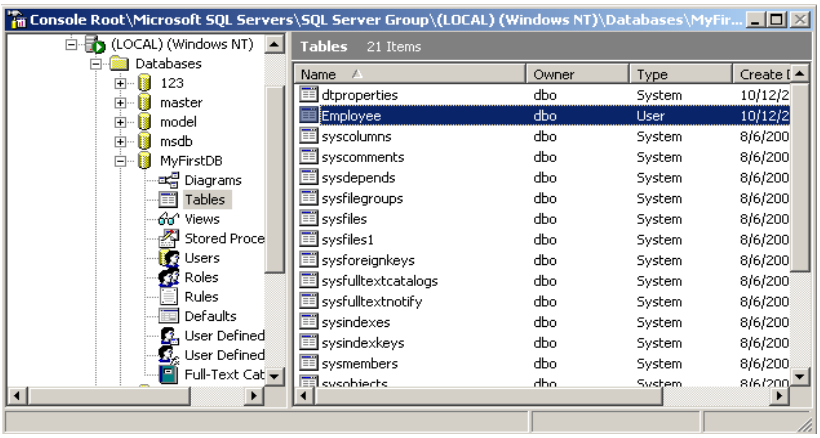
11. Click close icon, the **SQL Server Enterprise Manager** will open. Click **Yes**.



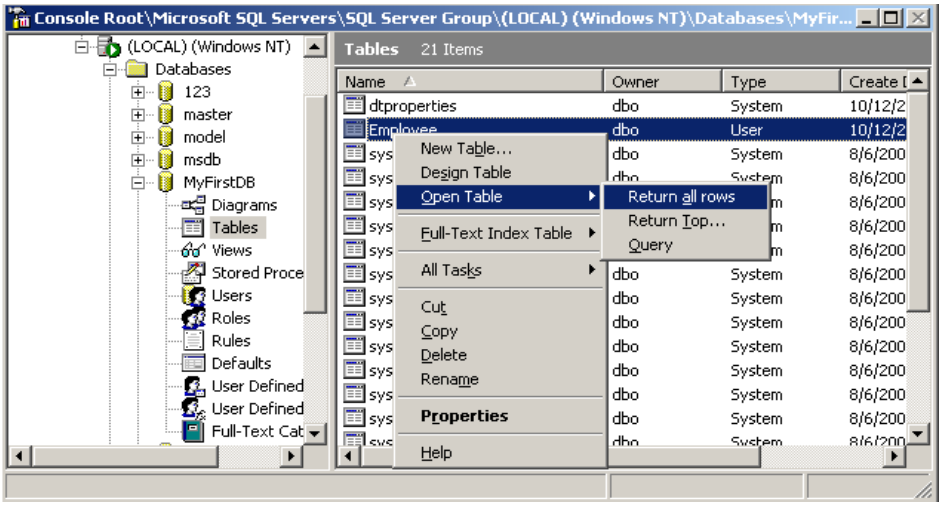
12. Type Employee in variable name. Click OK.



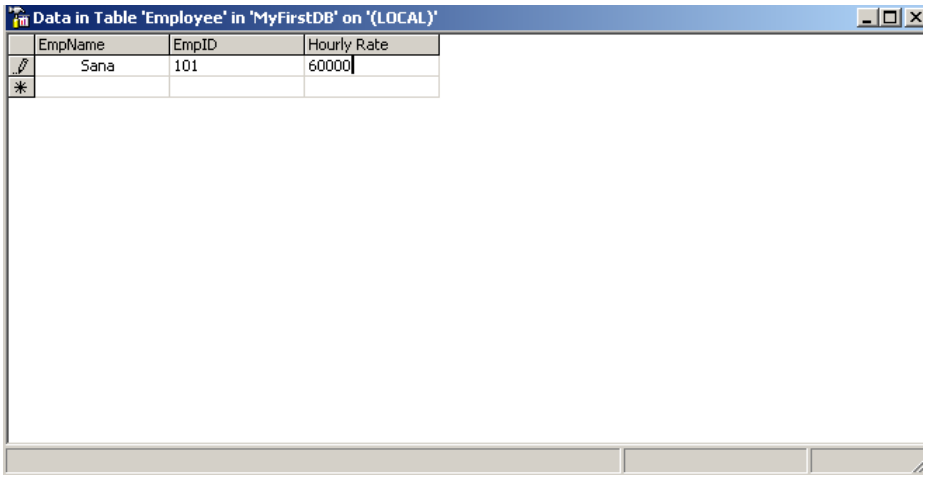
13. See in the following figure the new created table Employee. It is added in Tables list.



14. Right click on table name, select **Open Table** options and then **Return all rows** option.



15. Enter data in table manually or using some application.



Follow above method for all tables.

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

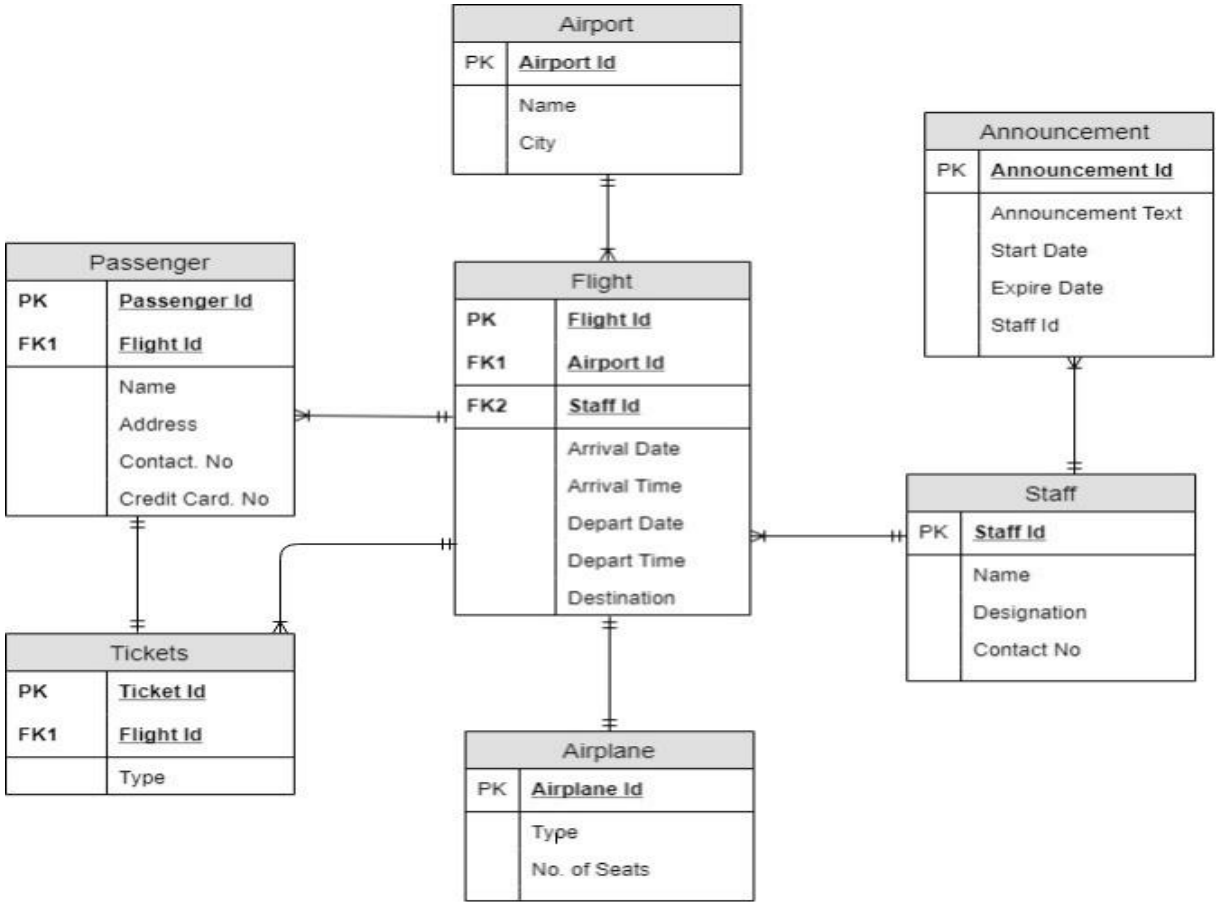
Lab 3

DENORMALIZATION

Scenario

"Pak Airline" is an airliner reservation company, which is operating in more than 10 countries. They have developed the airline reservation system to avoid the errors faced in manual system. The staff of the airline use airline reservation system form the tasks such as flight scheduling, ticket reservation, announcements in automated way. Similarly, users/passengers can search for flight schedule according to date and time and fare details. The staff of the airline can manage the reservation systems by flight rout, runway details, flight scheduling and reservation.

Ticket reservation system of the Pak Airline provides the information about schedule of flights, availability of seats, flight number and destination. For reservation of ticket user have to provide its personal information such as name, age, address etc. For payment purpose user will provide credit card number and bank details. Moreover, information about flight number, date of departure, no. of tickets to be booked is also required for confirmation of ticket. Following is the ERD of above airline reservation system.



Question Statement:

You are required to de-normalize above tables using table pre-joining technique according the relationship between entities. Carefully identify all tables with such relationships on which Pre-joining technique can be applied.

Solution:

There are One-to-Many relationships in following entities:

1. Staff-Announcement
2. Airport-Flight
3. Flight-Passenger
4. Staff-Flight
5. Flight-Ticket

As Pre-joining De-normalization technique is based on 1-many relationship so the De-normalization will be performed on following tables.

Airport Table

Airport Id	Name	City

Flight Table

Flight Id	Arrival date	Arrival time	Depart Date	Depart Time	Destination	Airport Id	Staff Id

Staff Table

Staff Id	Name	Designation	Contact No

Announcement Table

Announcement Id	Announcement Text	Start Date	Expire Date	Staff Id

Passenger Table

Passenger Id	Name	Contact No	Address	Email	Credit Cr.No	Flight Id

Ticket Table

Ticket Id	Type	Flight Id

Merge Staff and announcement table as there is one to many relationship which is the requirement of Pre-joining.

Staff-Announcement

Staff Id	Name	Designation	Contact No	Announcement Id	Text	Start Date	Expire Date

Airport-Flight Table

Airport Id	Name	City	Flight Id	Arrival date	Arrival time	Depart Date	Depart Time	Destination

Flight –Passenger Table

Flight Id	Arrival date	Arrival time	Depart Date	Depart Time	Destination	Passenger Id	Name	Contact No	Address	Email	Credit Cr.No	Airport Id	Staff Id

Staff-Flight Table

Staff Id	Name	Designation	Contact No	Flight Id	Arrival date	Arrival time	Depart Date	Depart Time	Destination

Flight-Ticket Table

Flight Id	Arrival date	Arrival time	Depart Date	Depart Time	Destination	Airport Id	Staff Id	Ticket Id	Type

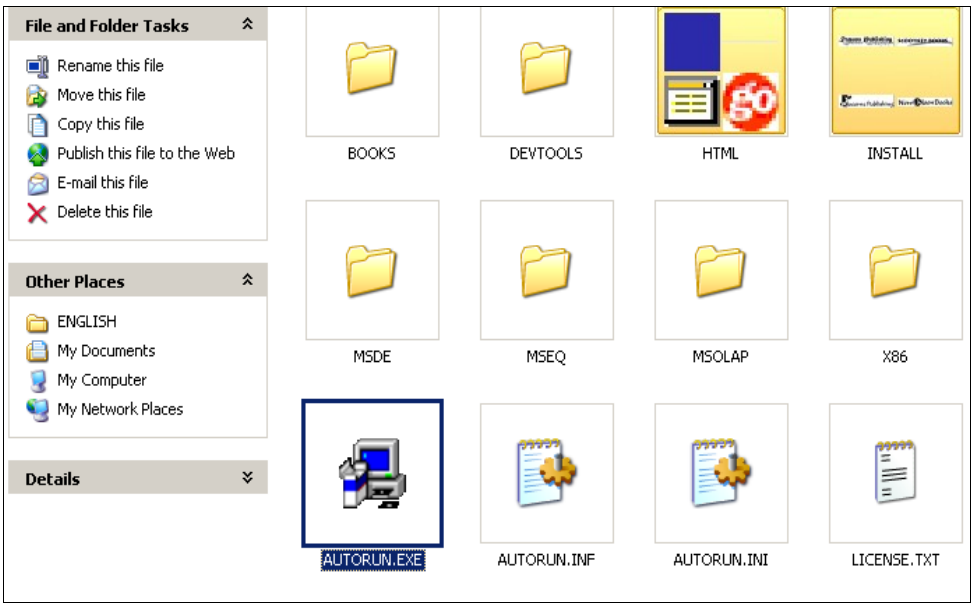
Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

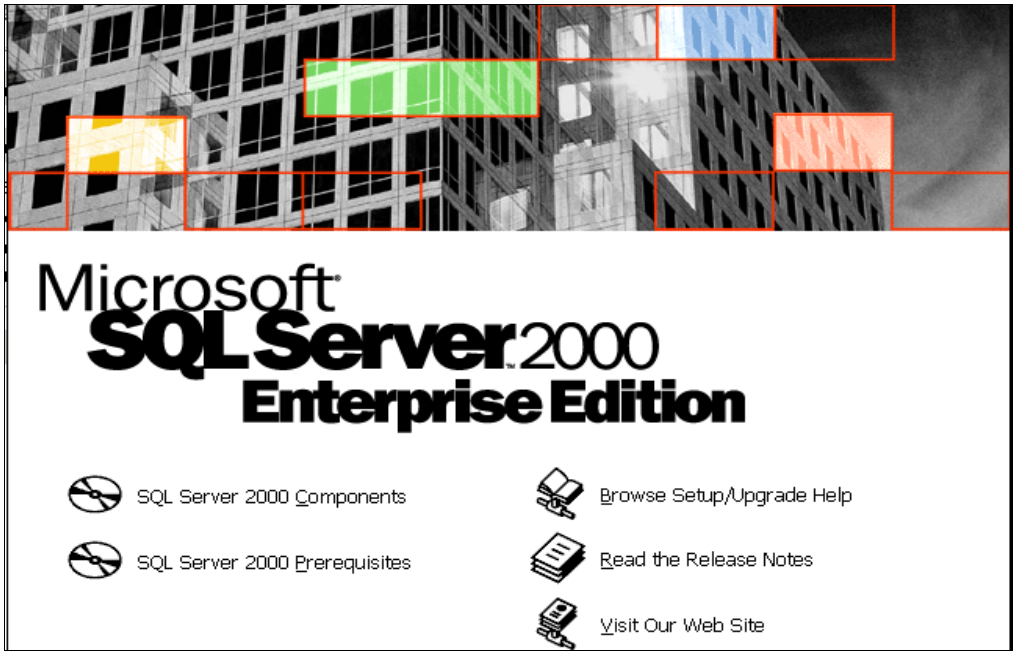
Lab 4 Part I

Installation Guide for MS SQL Server 2000 Analysis Services

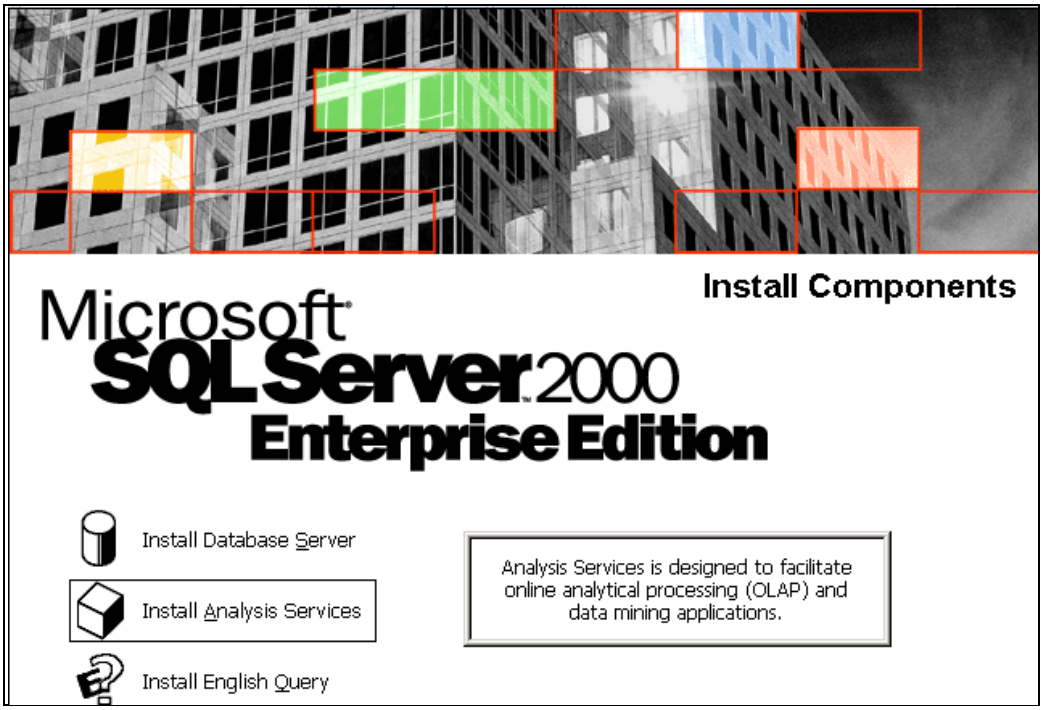
1. Follow all instructions in Lab lecture 1 section 3.2 “Installing Microsoft SQL Server 2000” and then Double Click on “AutoRun.Exe” icon.



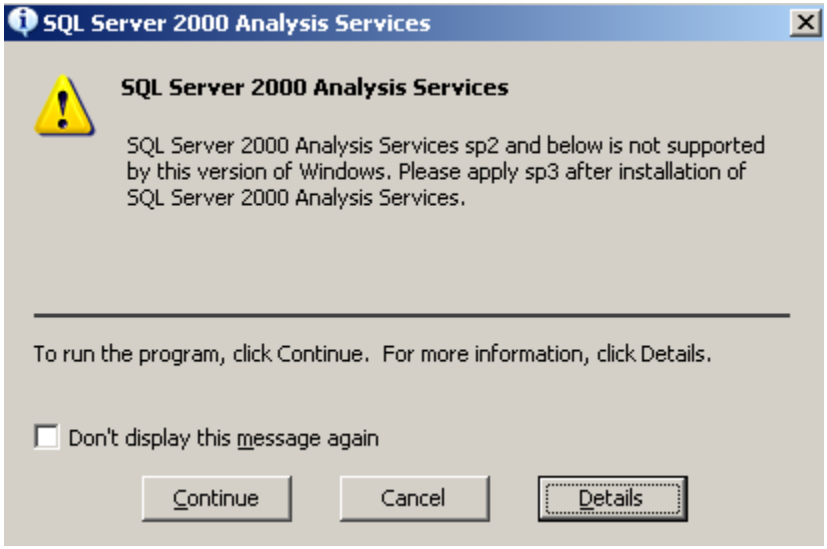
2. The Microsoft SQL Server 2000 Enterprise Edition screen will appear. Now, press the SQL Server 2000 Components button.



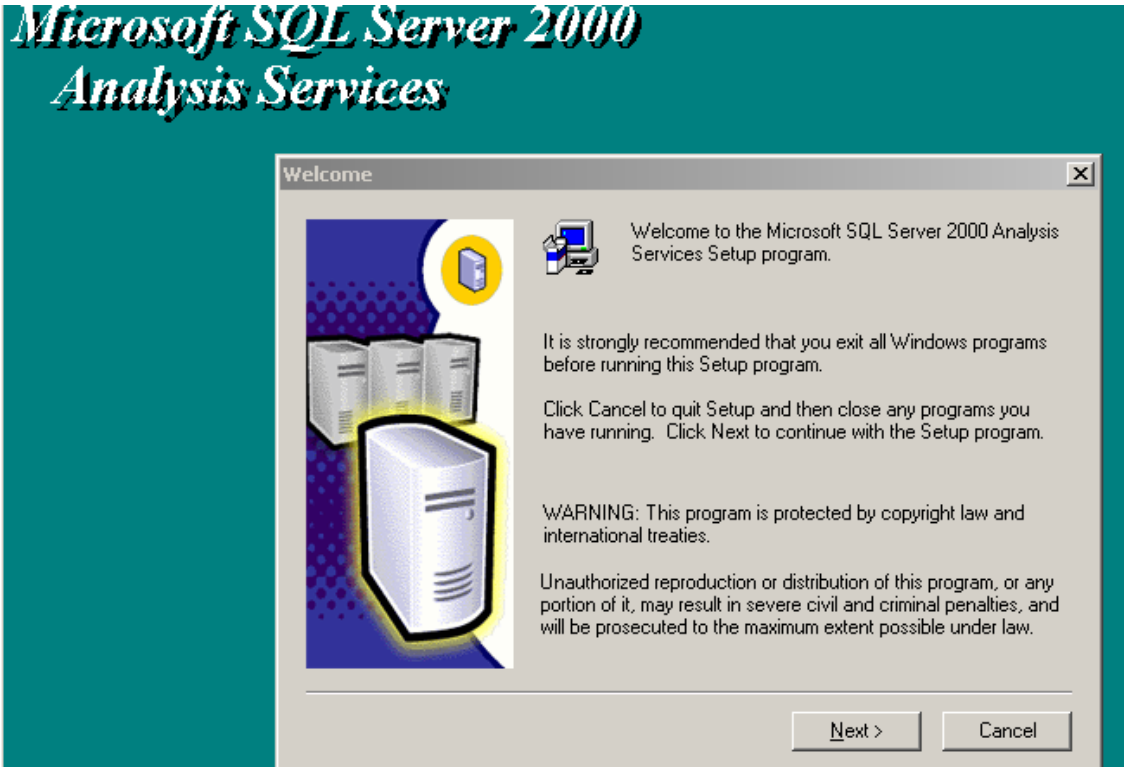
3. Now, Press “Install Analysis Services” button.



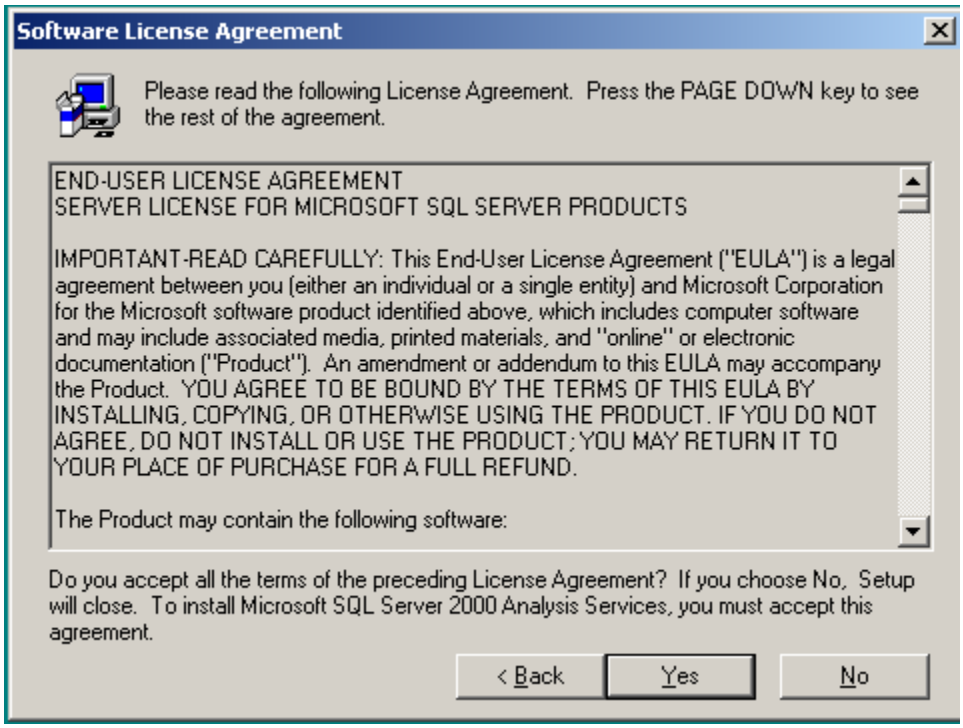
4. The following window will appear. Click “Continue” button.



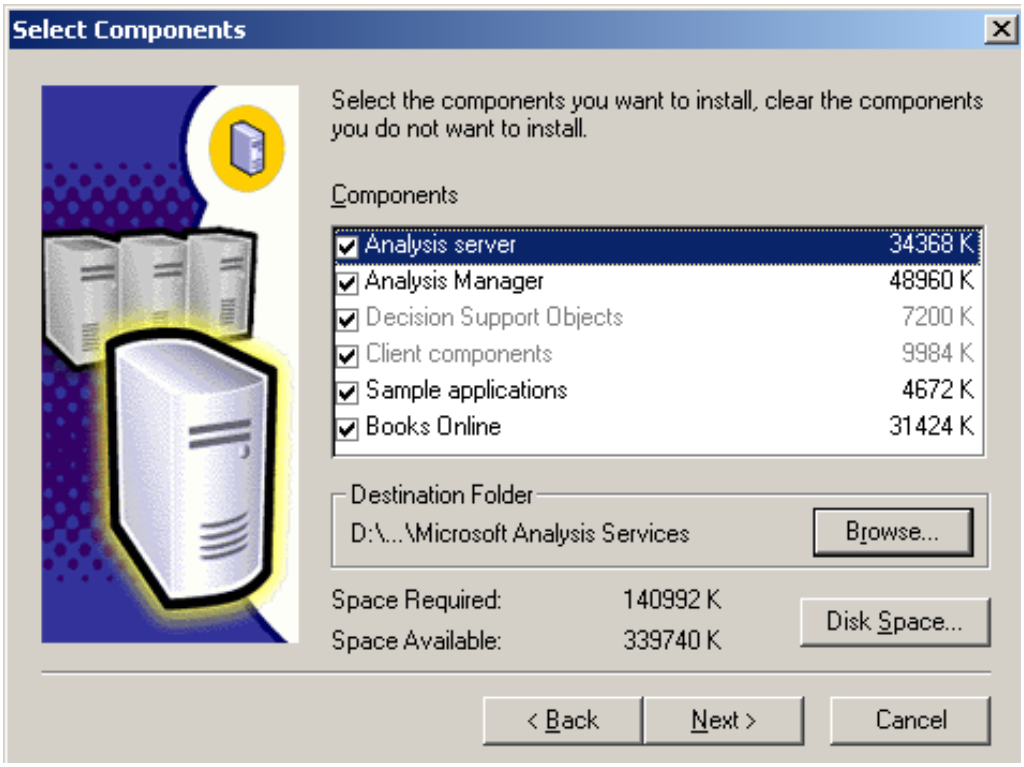
5. From Microsoft SQL Server 2000 Analysis Services screen, Click Next button.



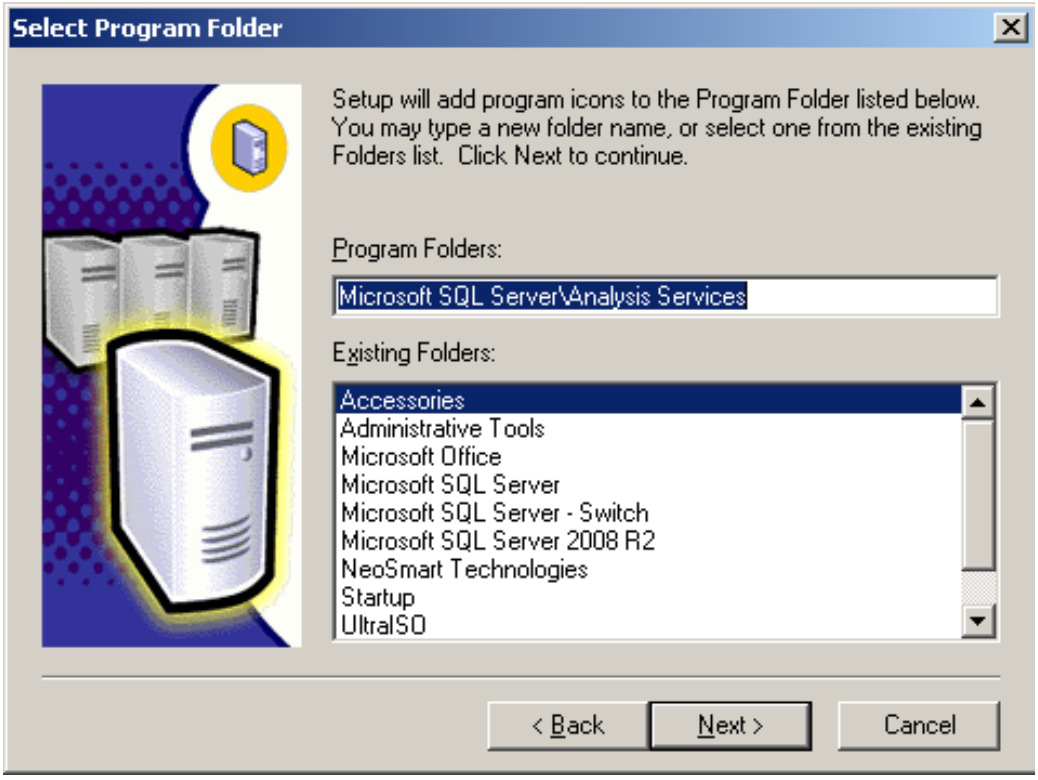
6. Click “Yes” from following “Software License Agreement” window.



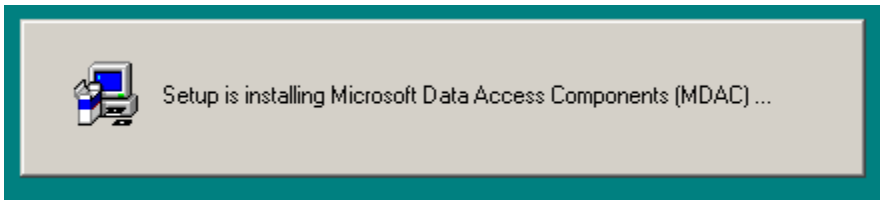
7. Select Component as selected in following “Select Components” window.



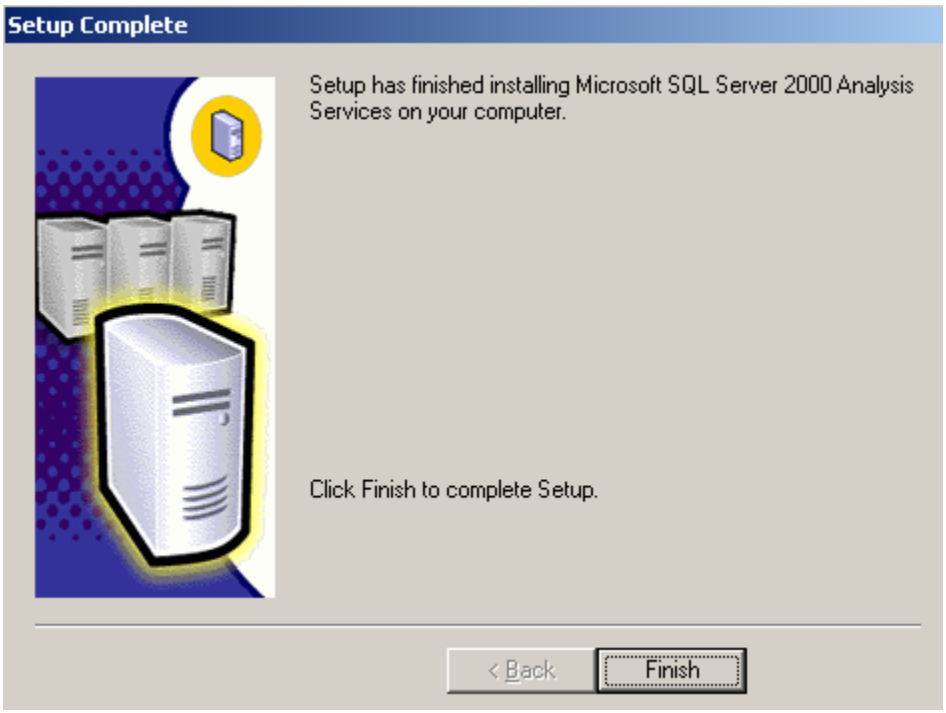
8. Select Program Folder and click “Next” button.



9. The following window will appear.

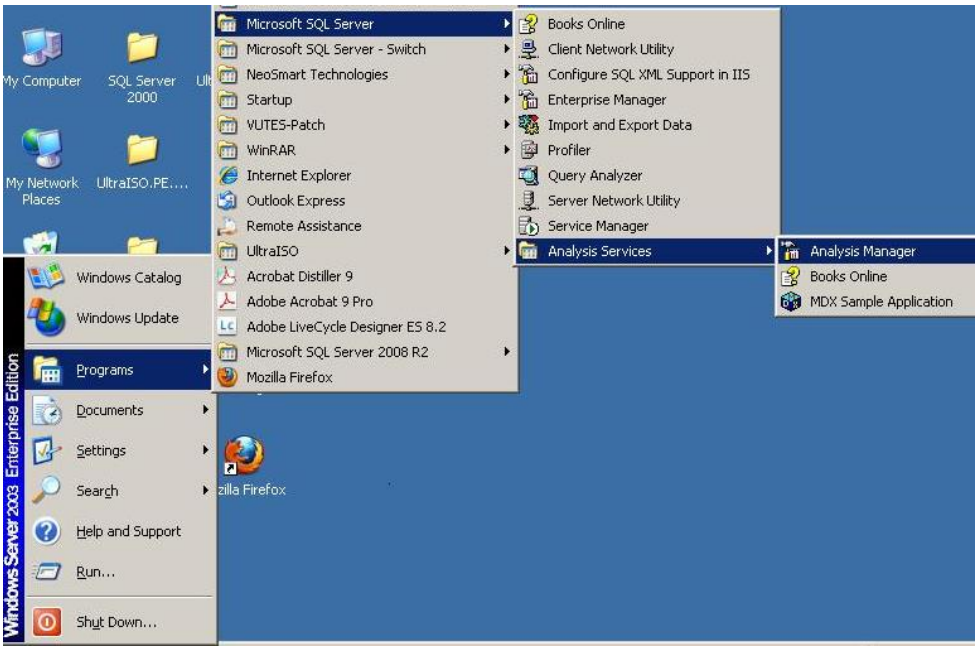


10. Click “Finish” button to complete setup.

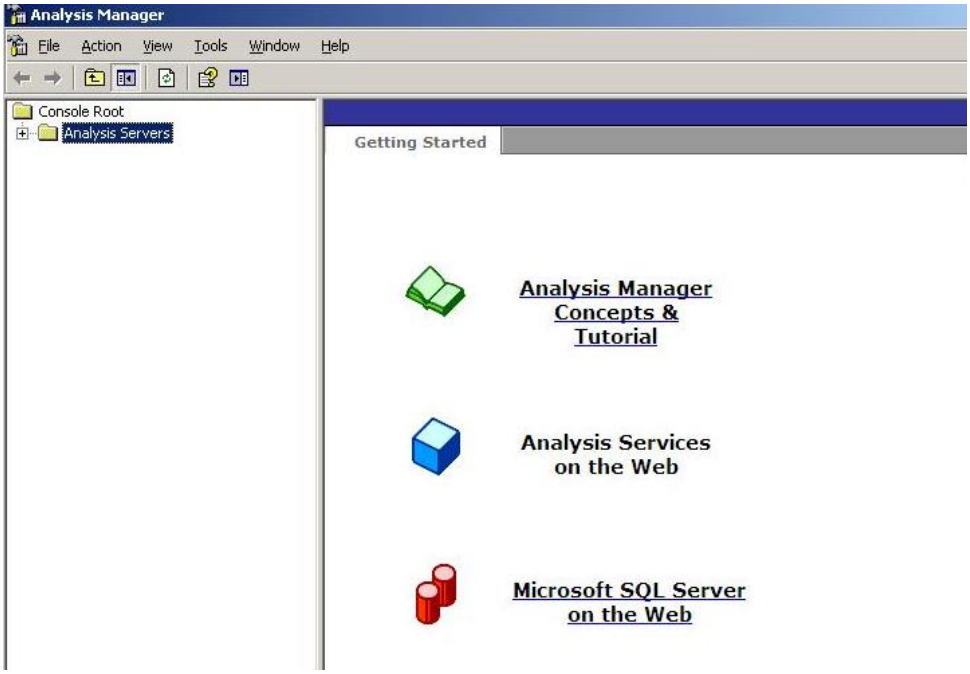


Working with Analysis Manager

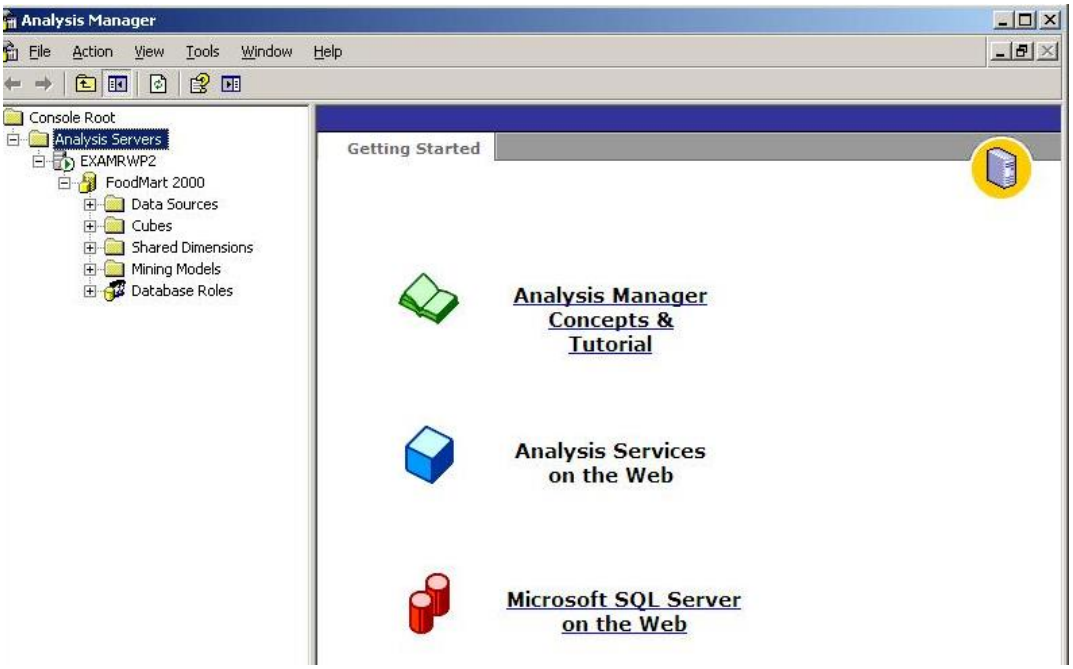
1. Open **Analysis Manager** by clicking Programs->Microsoft SQL Server->Analysis Services->Analysis Manager**.**



2. The following **Analysis Manager** Window will open. To get help on working with **Analysis Manager**, click on **Analysis Manager Concepts & Tutorial.**

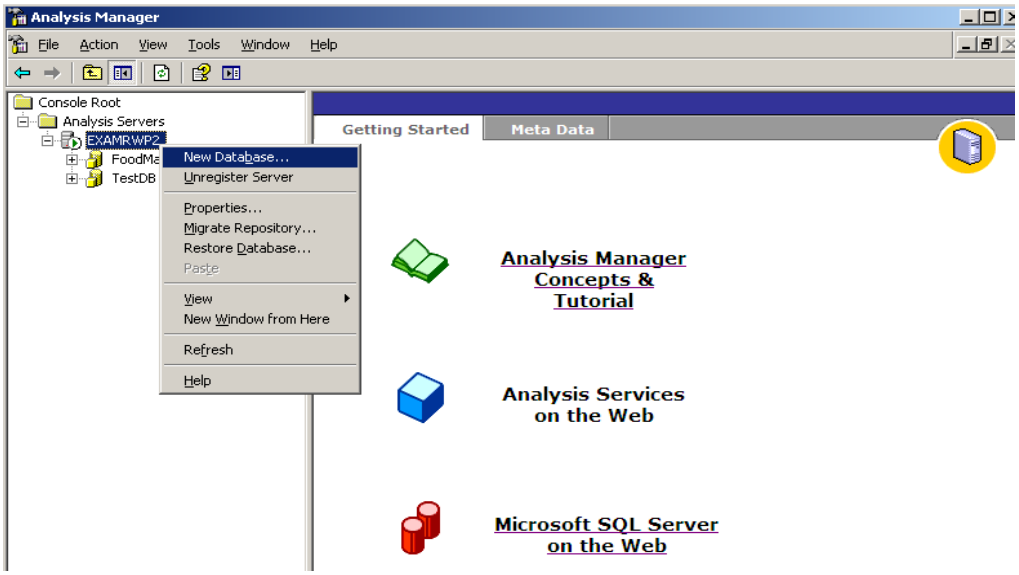


3. In the Analysis Manager tree view, expand **Analysis Servers**.

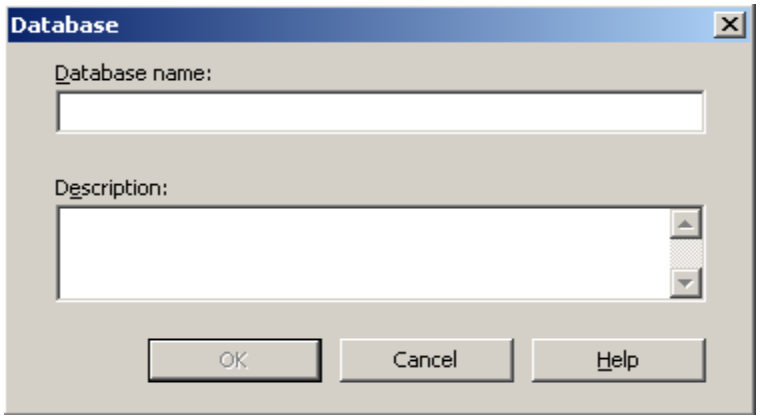


4. In above screen, **FoodMart2000** under the server node is the sample database available in Analysis Manager.
5. **To create a new database:**

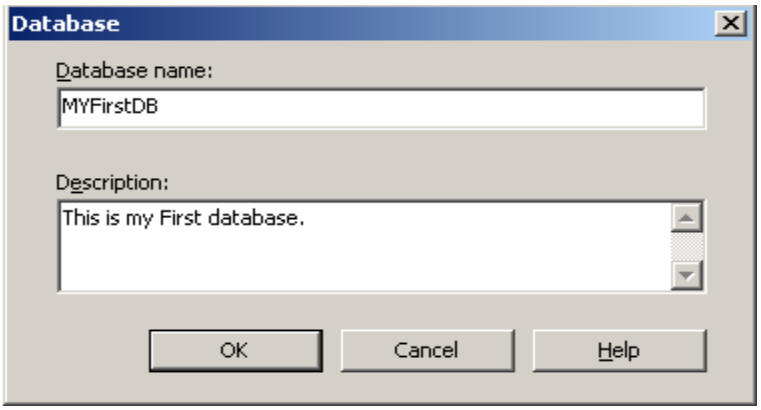
1. Click the name of your server (e.g. E**XAMRWP2**), a connection with the Analysis server will be established.
2. Right-click your server's name, and then click **New Database**.



3. The following database dialog box will open.



4. In the **Database** dialog box, in the **Database name** box, enter "MYFirstDB", and in **Description** box, enter "This is my Fisrt database.", then click **OK**.



5. In the Analysis Manager tree pane, expand the server, and then expand the **MYFirstDB** database that you just created.
6. Your new **MYFirstDB** database contains the following items:

- Data Sources
- Cubes
- Shared Dimensions
- Mining Models
- Database Roles

Lab Exercise: Explore sample database **FoodMart2000,** all of its tables, tables' schema and browse their data.

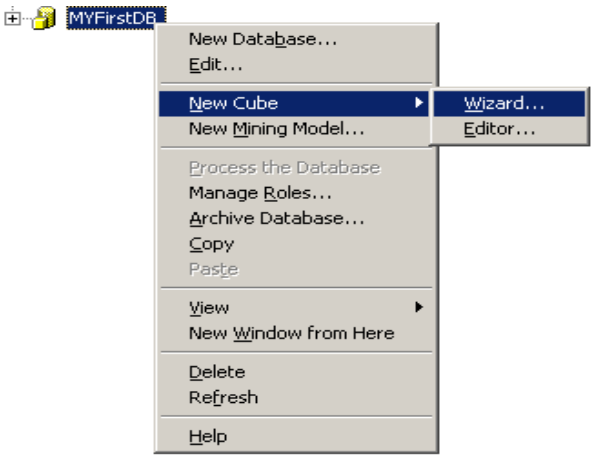
Mechanism to Conduct Lab:

Students and teacher communicate through Skype/Adobe Connect. Students perform the task using the following simulator:

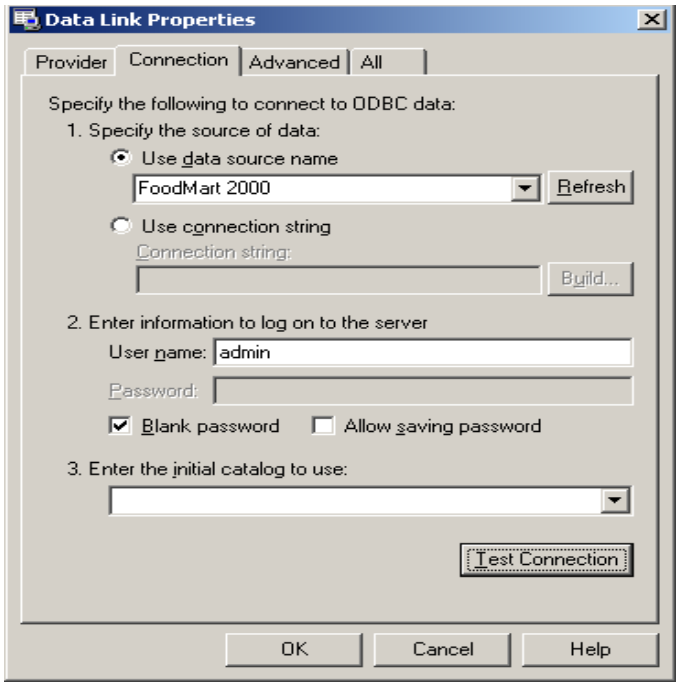
Lab 4 Part II

CREATING AND PROCESSING A CUBE

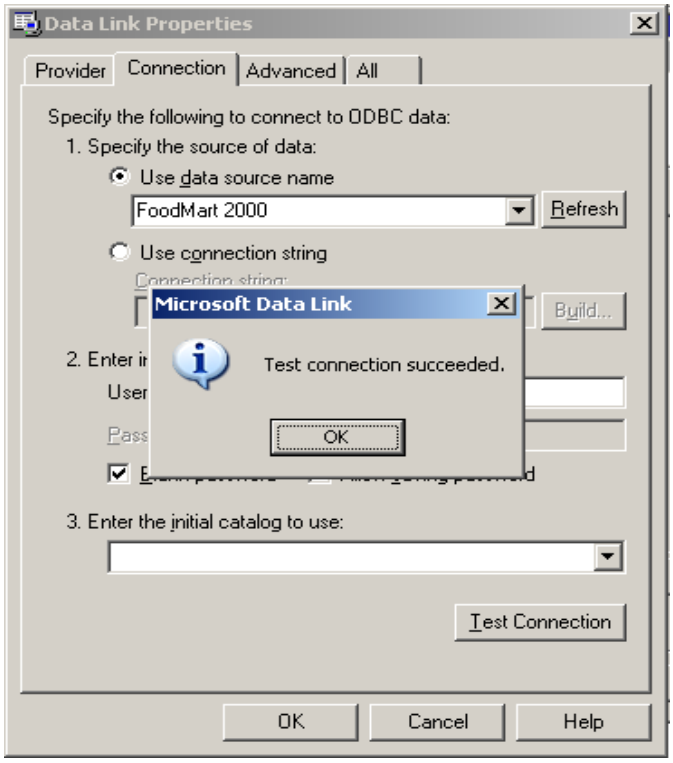
1. Click on newly created database **"MYFirstDB",** from its drop down list, select **New Cube,** then select **Wizard** option**.**



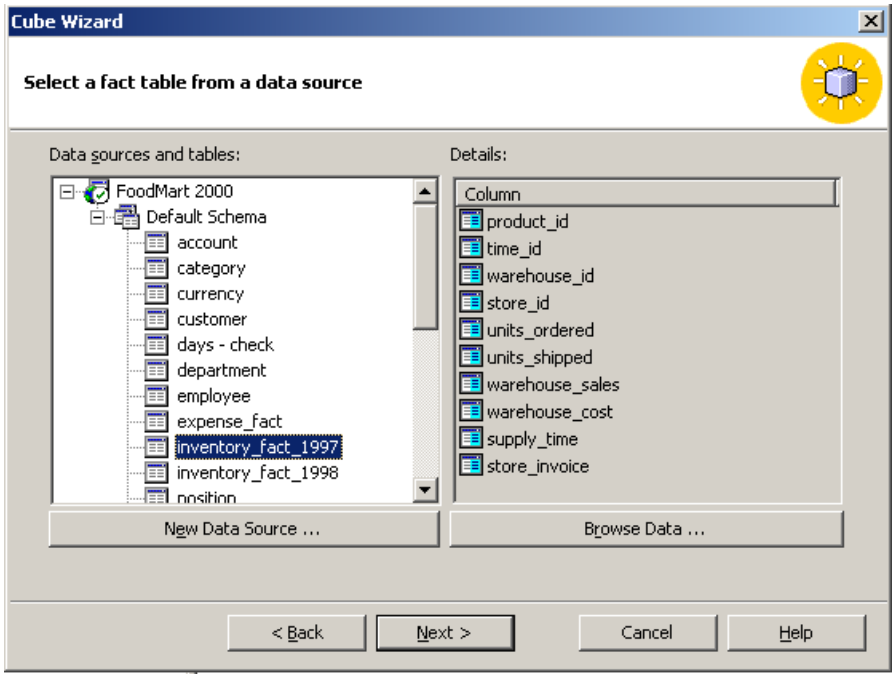
2. The following **Data Link Properties** window will open. Select option **Use data source name** and then select **FoodMart2000** from its dropdown list.
3. Write **Admin** as **User name** and check **Blank Password** option.
4. After this, Click Test Connection button.



5. You will get message “Test Connection Succeeded”. Click OK, then again click OK from Data Link Properties.

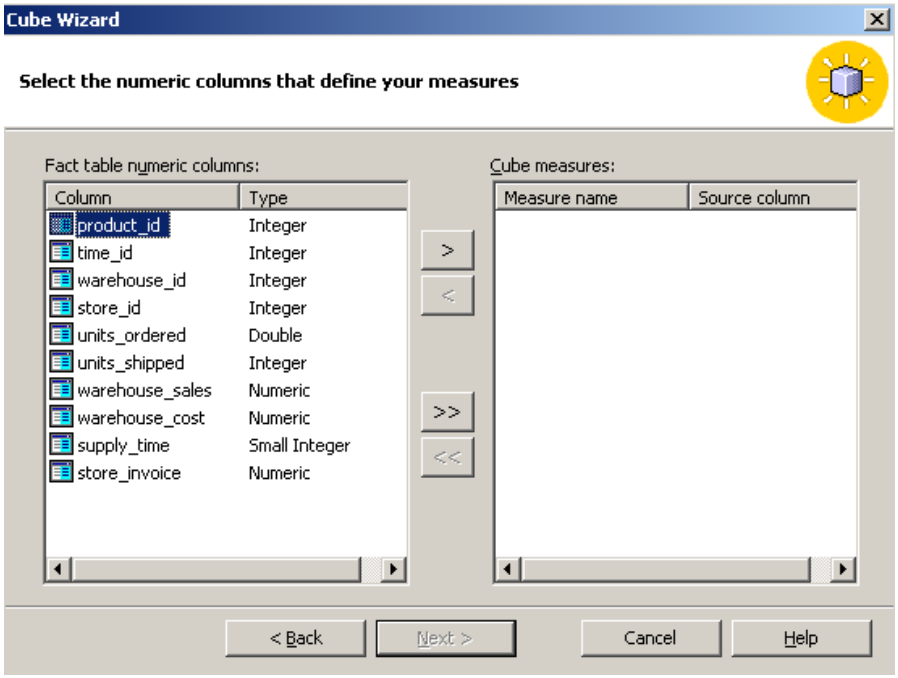


6. Cube Wizard window will show different tables which are created in FoodMart2000 database. Select **inventory_fact_1997** table.

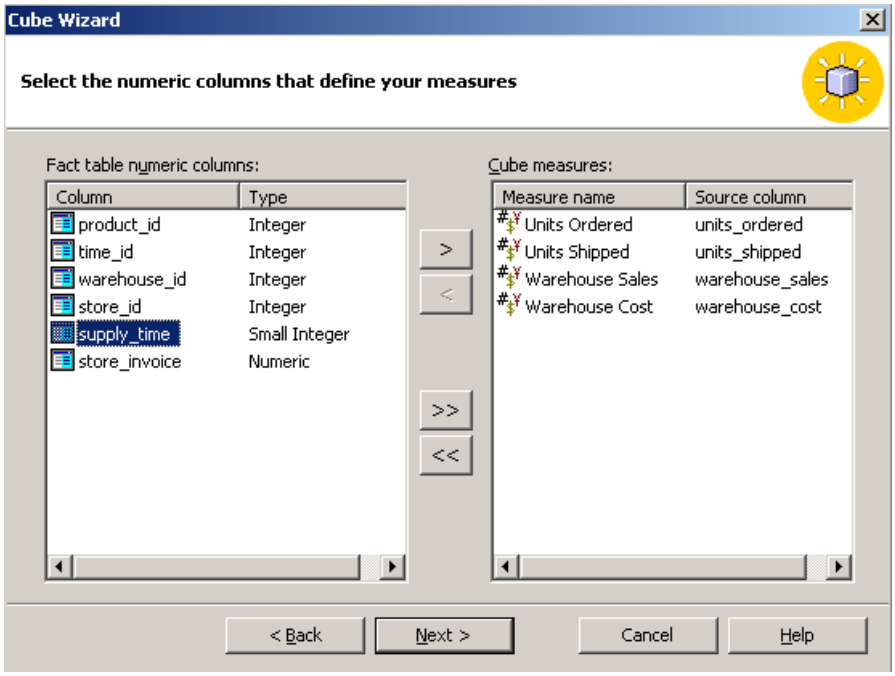


7. You can browse data of selected table by clicking on Browse Data button (See above window).

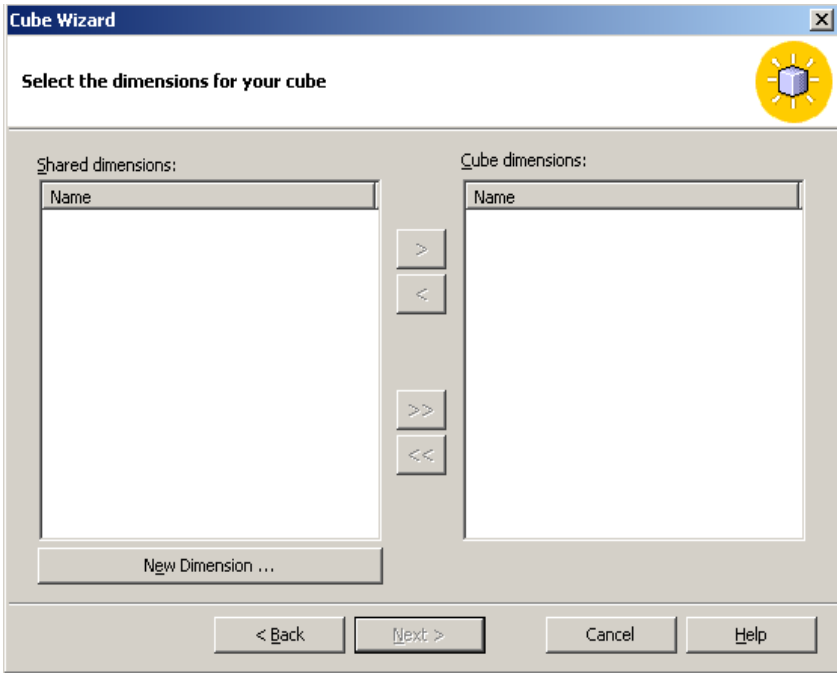
8. Now you will define your cube measures. Select table numeric columns to define cube measures.



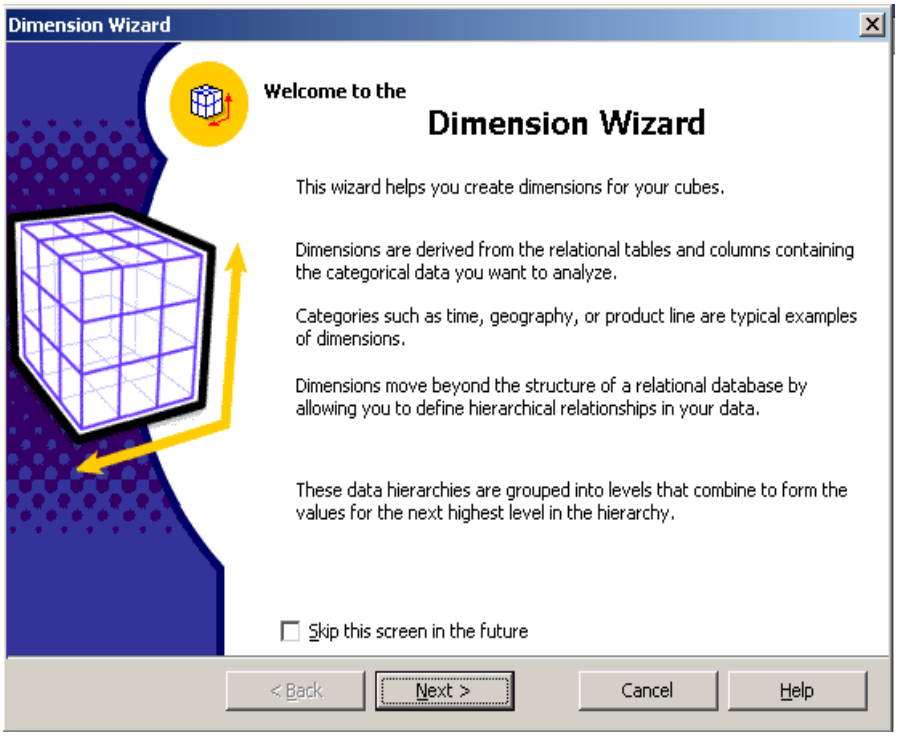
9. Select e.g. Units Ordered, Links Shipped,, Warehouse Sales, and Warehouse Cost. Then click on Next button.



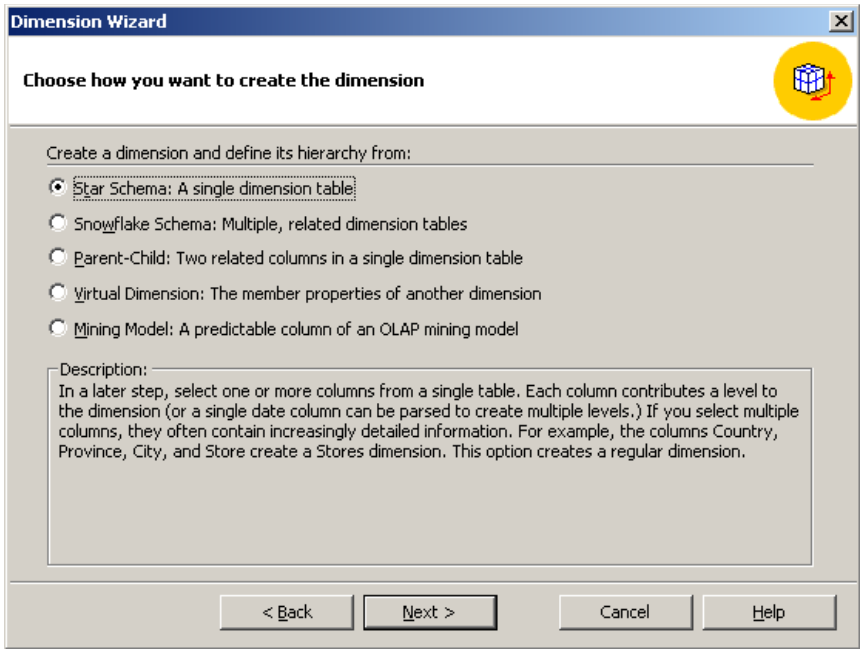
10. Now, you have to select cube dimension. From **Cube Wizard** window, click on **New Dimension.**



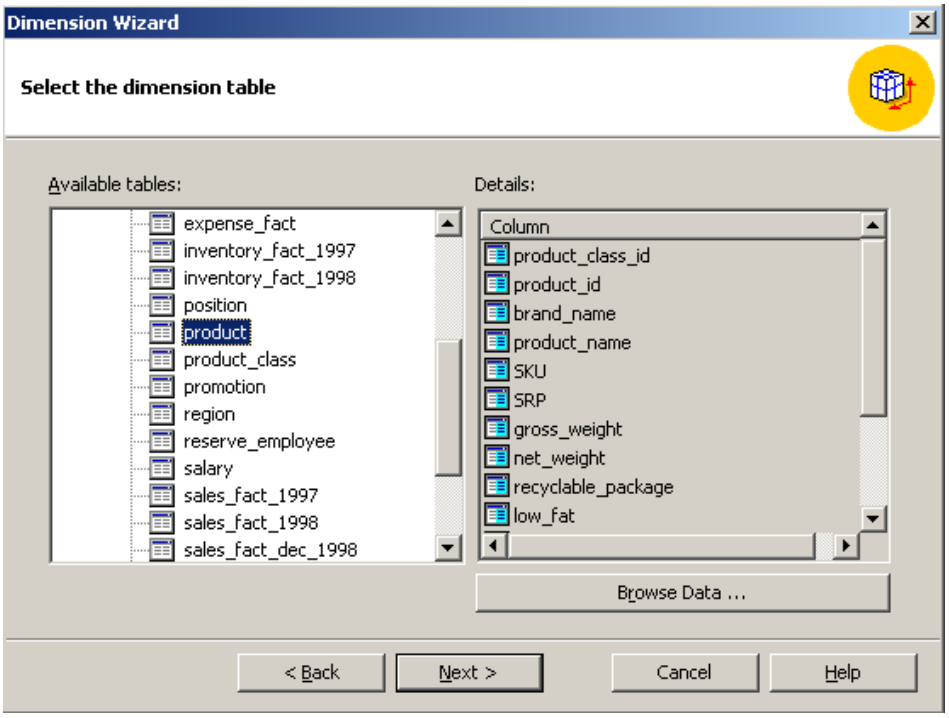
11. Click on **Next** button.



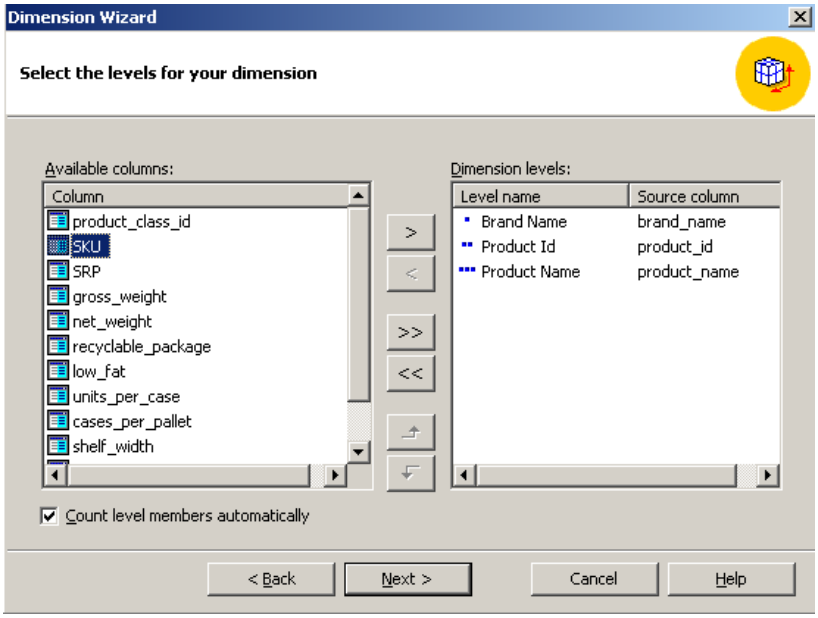
13. Select **Star Schema: A single dimension tables** option**.** You can use some other option too. Click **Next**.



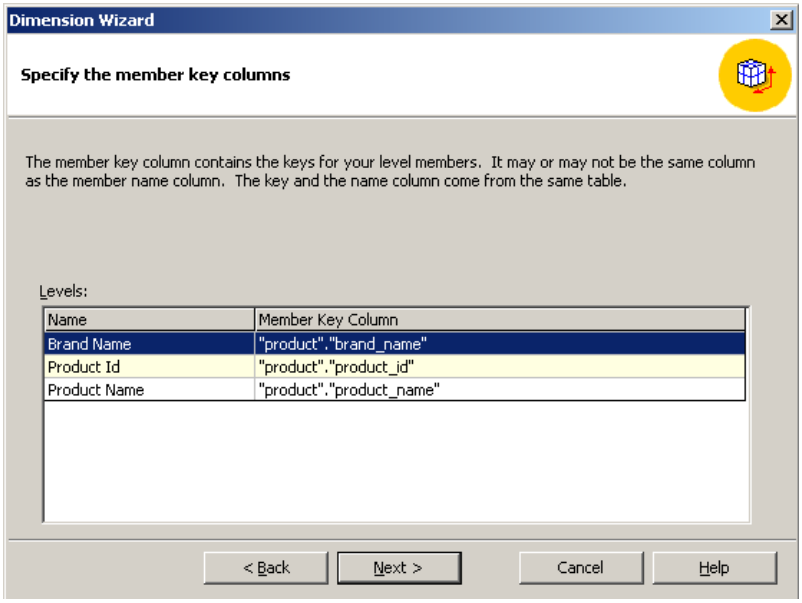
14. Now, you have to select Dimension table. In this lab, we have selected **product** table. You can choose any table of your choice. Click **Next**.



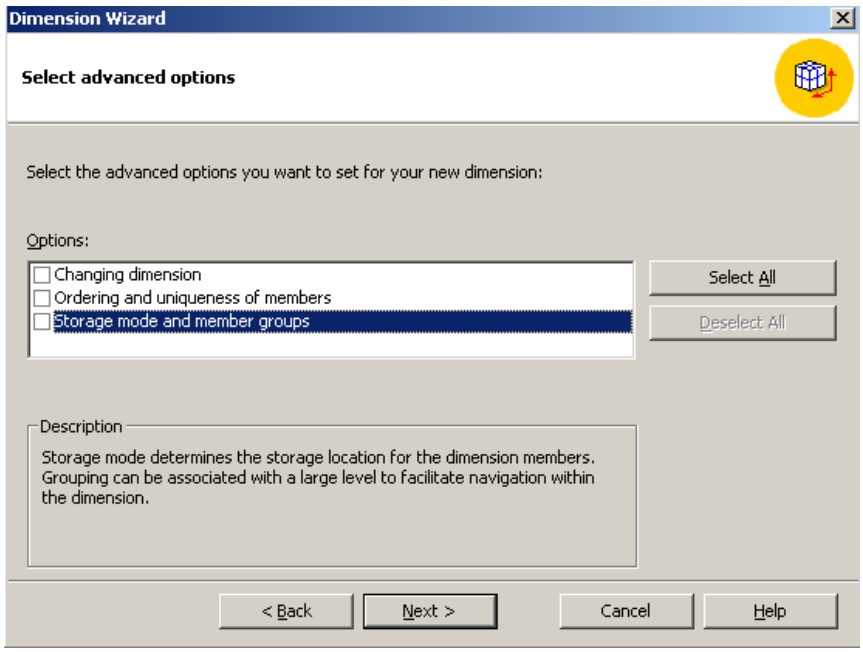
15. Now, select **level of your dimension**. Here in this example, we have selected BrandName, ProductId and ProductName as shown in following Figure. You can also select any other columns. Then, click **Next**.



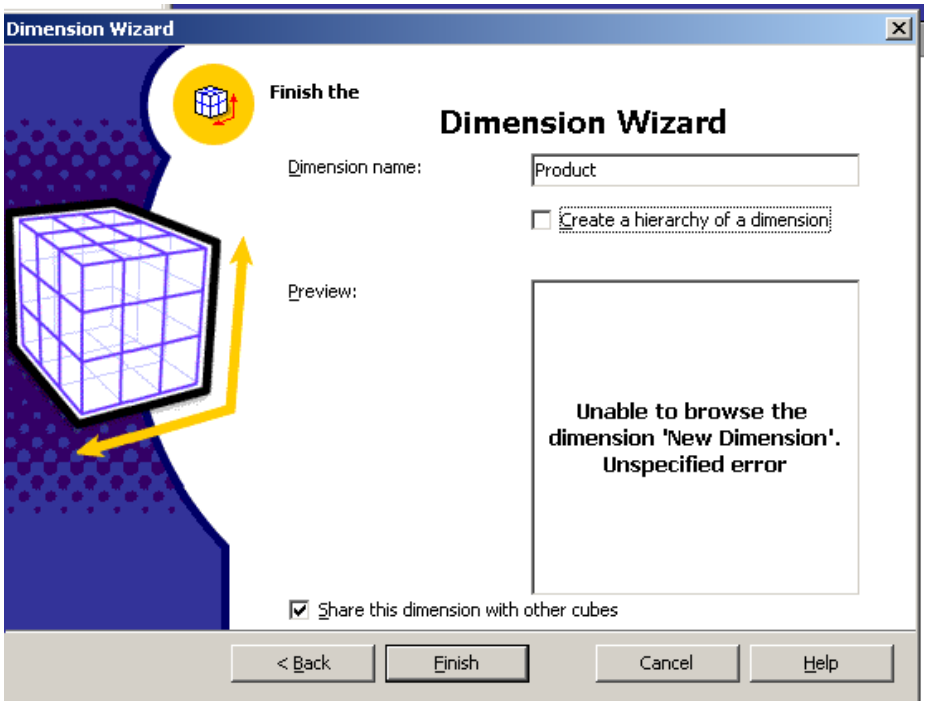
16. Now, select **member key columns** and Click **Next**.



17. Click **Next**.

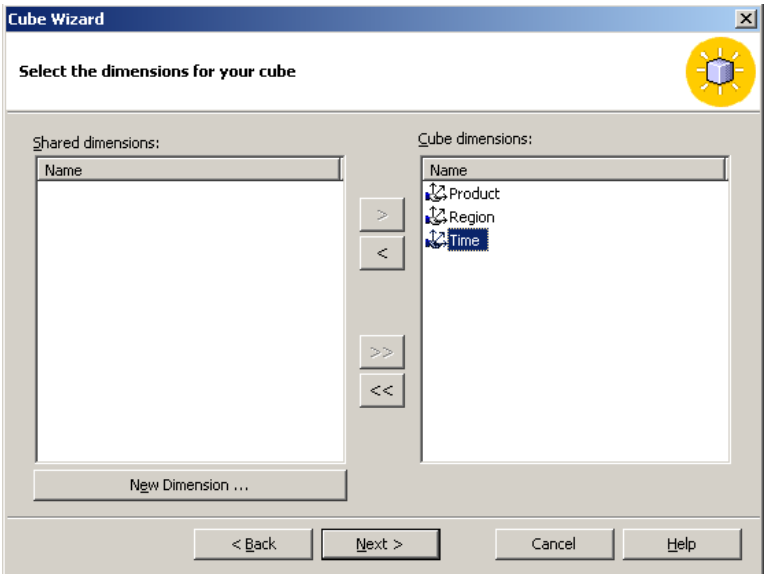


18. Click **Next**.

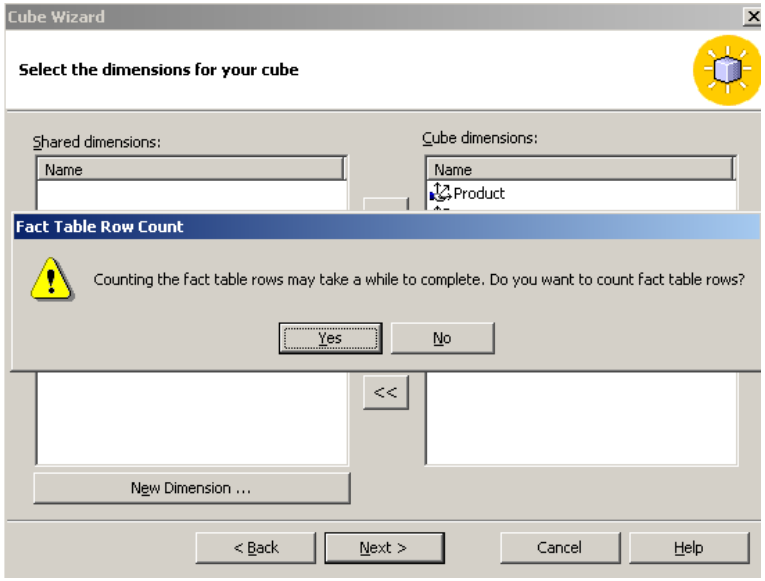


19. Type Product in **Dimension Name** and click **Finish**.

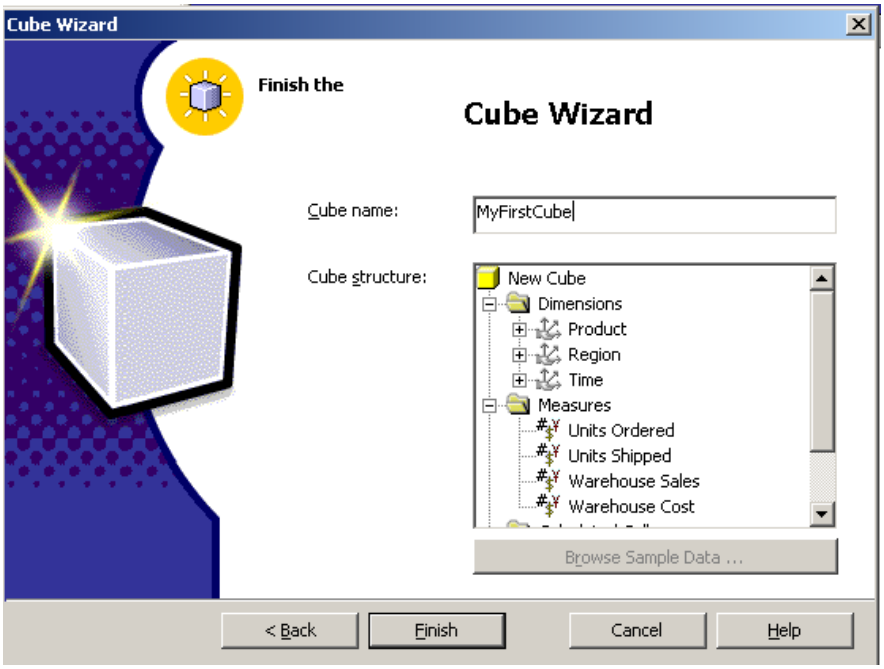
20. By following same steps, add two more dimensions Product, Region and time.



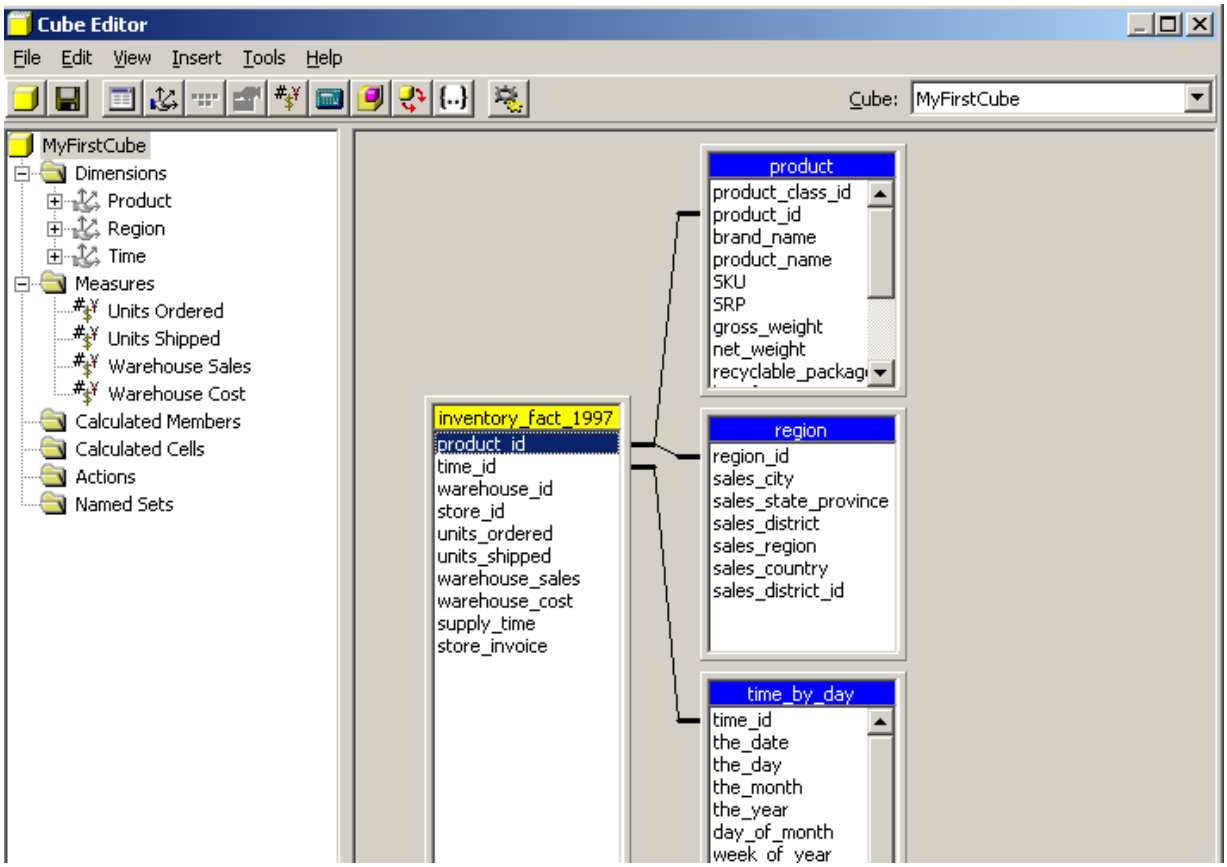
21. Click **Next**. Then, **Cube Wizard** window will open.



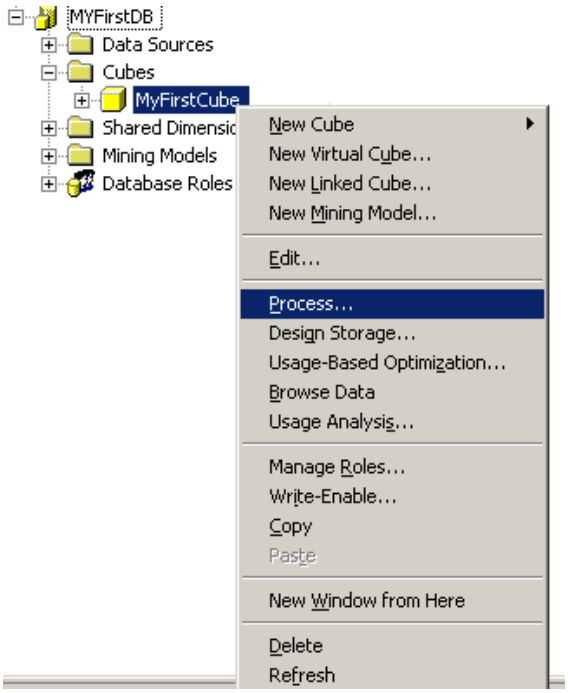
22. Click **Yes** in **Fact Table Row Count** window. Cube Wizard window will open. Type MyFirstCube in **Cube Name** and then Click **Finish**.



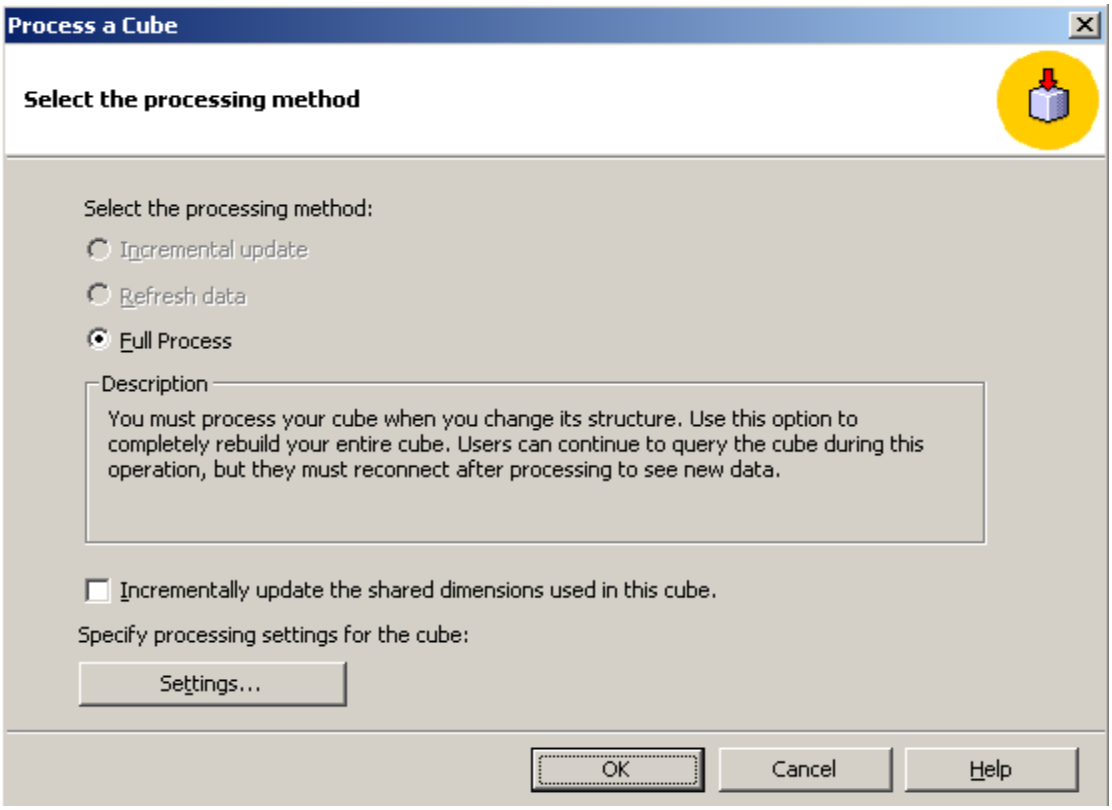
23. **Cube Edi**tor Window will open which shows Star Schema of your cube.



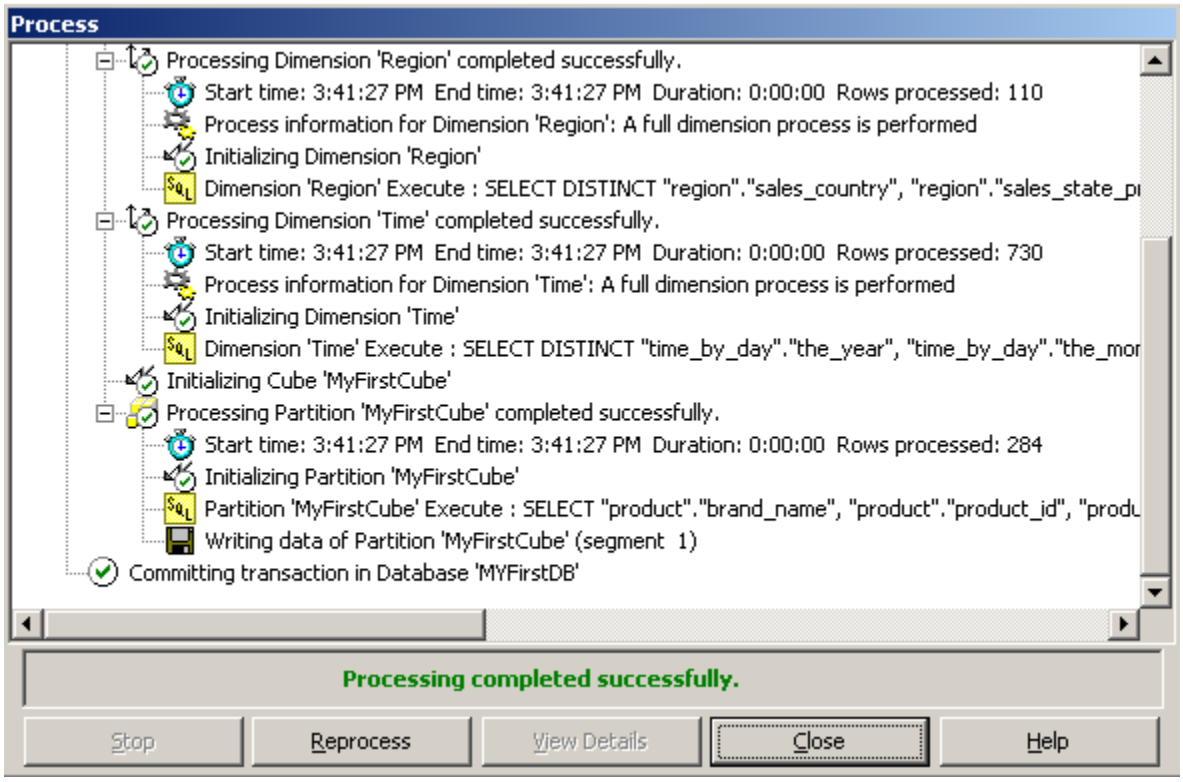
24. To process cube, click on cube name **MyFirstCube->Process.**



25. Select the processing method and click **Next**.



26. Cube processing completed.



Lab Exercise: Right Click on cube name and select Browse Data option to view cube data. Perform different cube operation on this data e.g. drill down, roll up etc.

Mechanism to Conduct Lab:

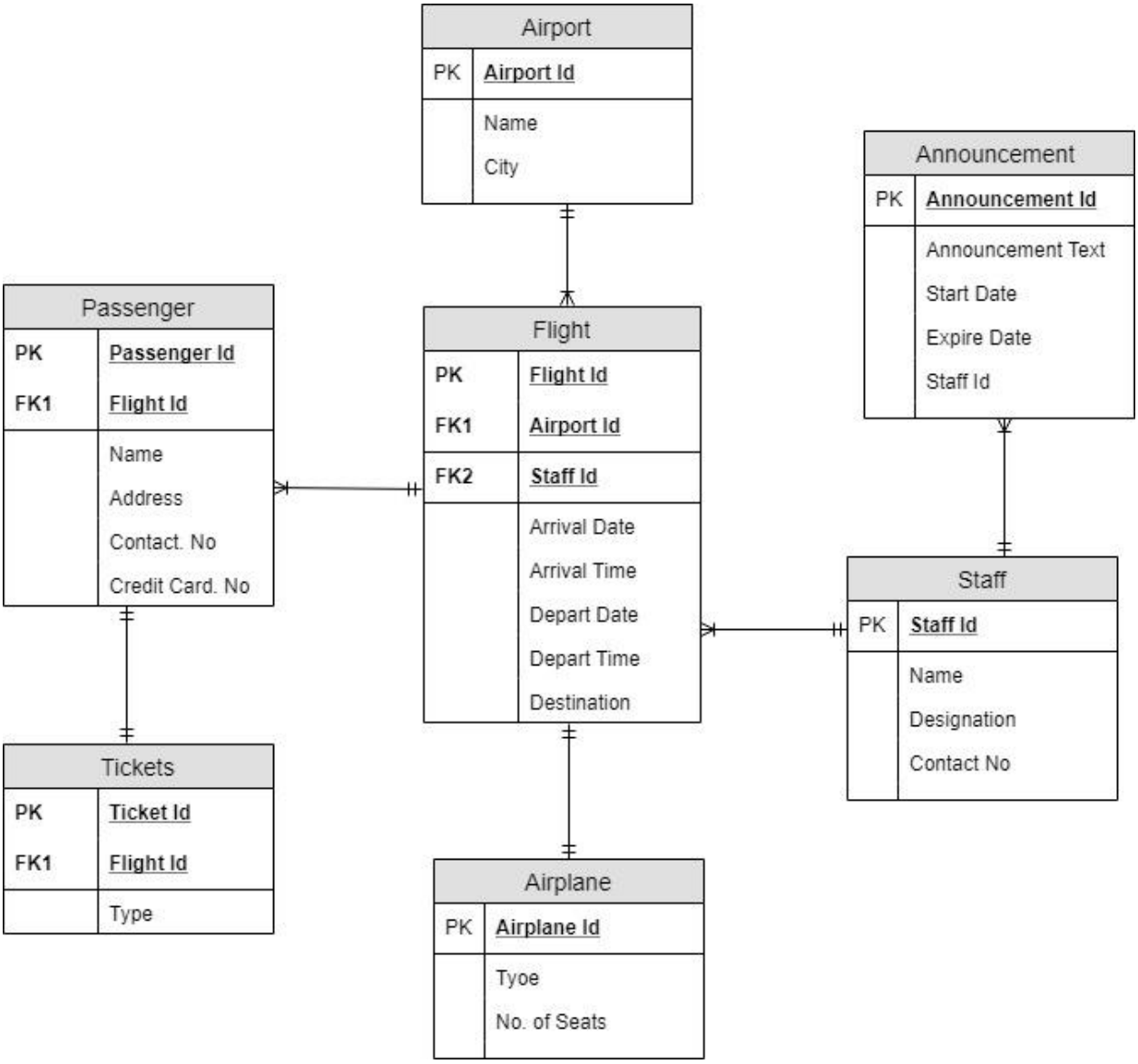
Students and teacher communicate through Adobe Connect.

Lab 5

STAR SCHEMA

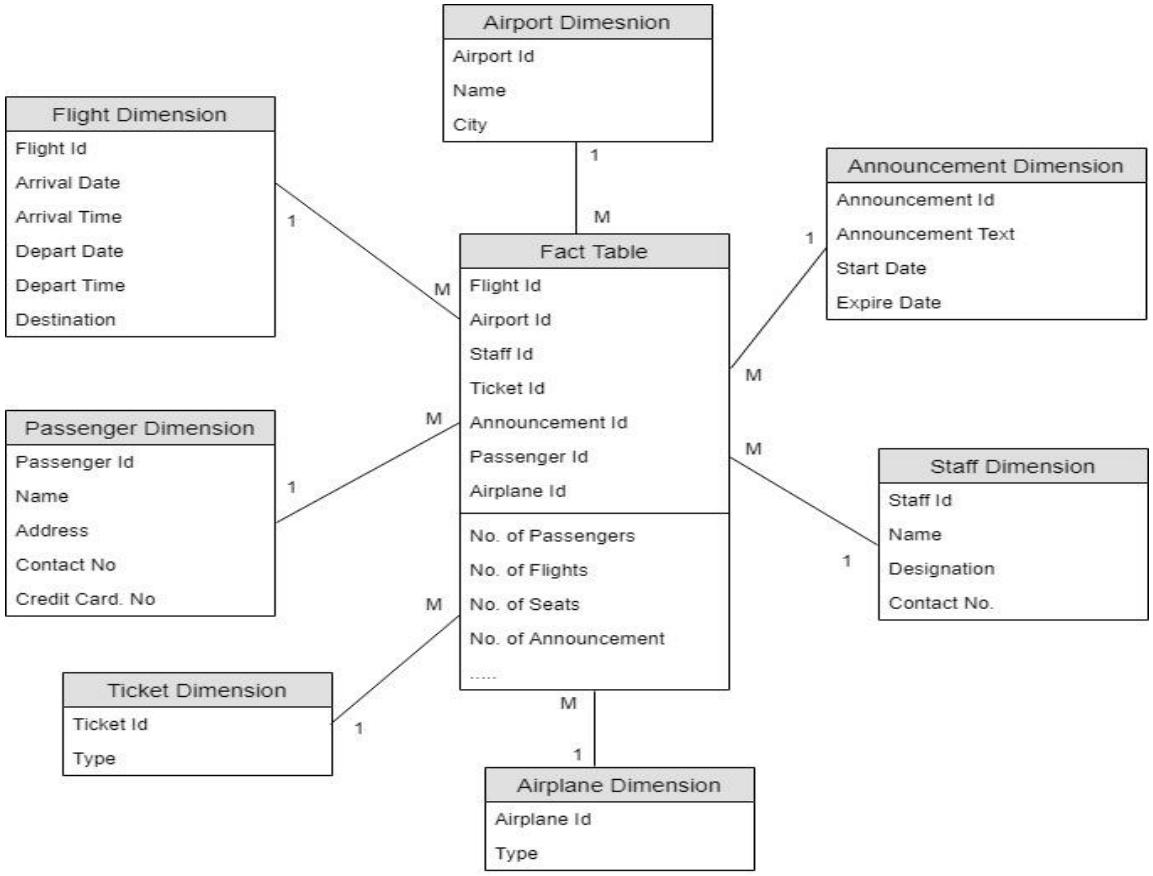
Question Statement:

Following is an **Entity Relationship Diagram (ERD)** of Airline Reservation System, you have to design **Star Schema** using all steps of dimension modelling in any drawing tool e.g. MS Visio etc.



Solution:

Following are steps of dimensional modeling.
Step-1: Choose the Business Process
The business process is “Air Ticket Reservation System”.
Step-2: Choose the Grain
Grain represents the atomic level of information required from business process and it is termed as unit of analyses. The grain statement is **“Total number of passengers arrives/depart in each flight by the airline.”**
Step-3: Choose the Facts
Facts are numeric, continuously valued and additive. The fact in our case is “**Total number of passengers in an airplane**”.
Step-4: Choose the Dimensions
The dimensions will be “Airport”, “Flight”, “Staff”, “Announcement”, “Passenger”, “Airplane”, and “Tickets”.
Star Schema:



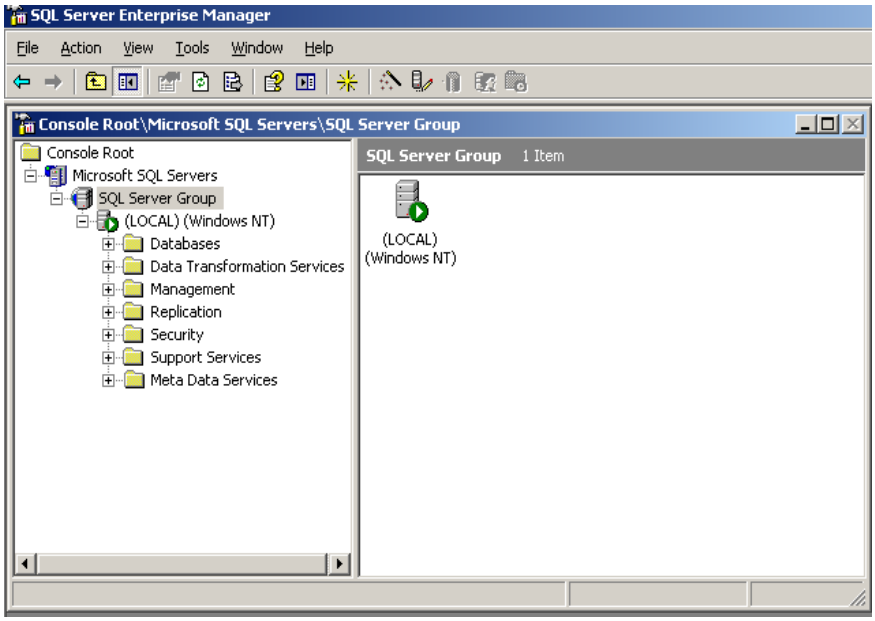
Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect Session.

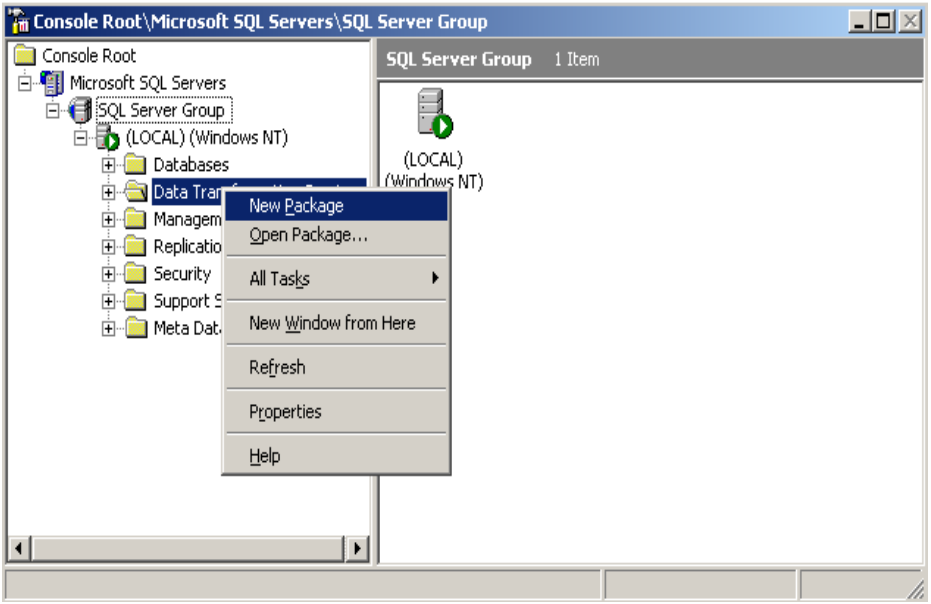
Lab 6

DATA EXTRACTION

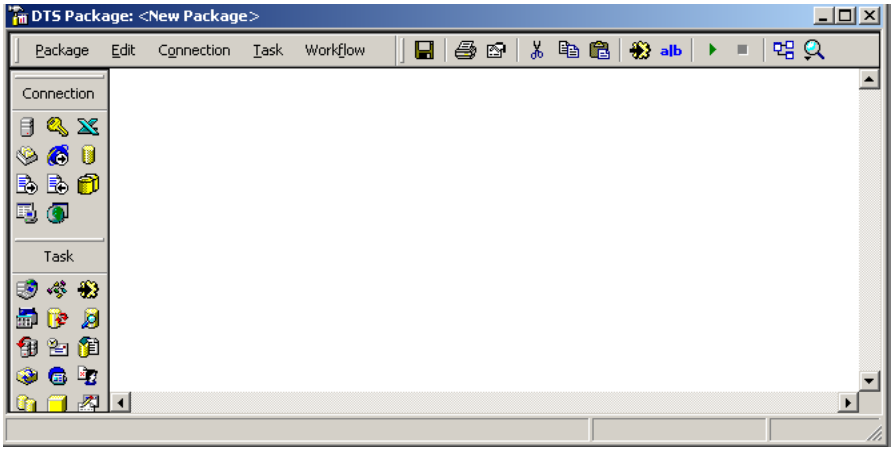
1. Open **SQL Server Enterprise Manager** and click on **SQL Server Group**.



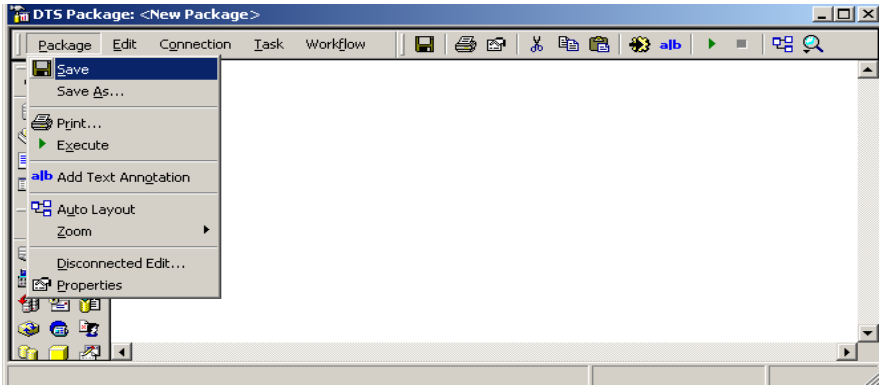
2. Right Click on **Data Transformation Services** node and click **New Package.**



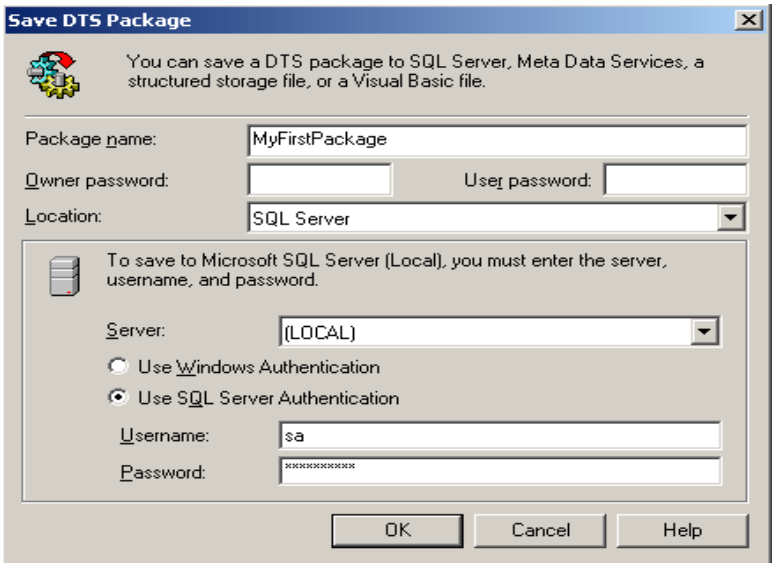
1. The following window will open.



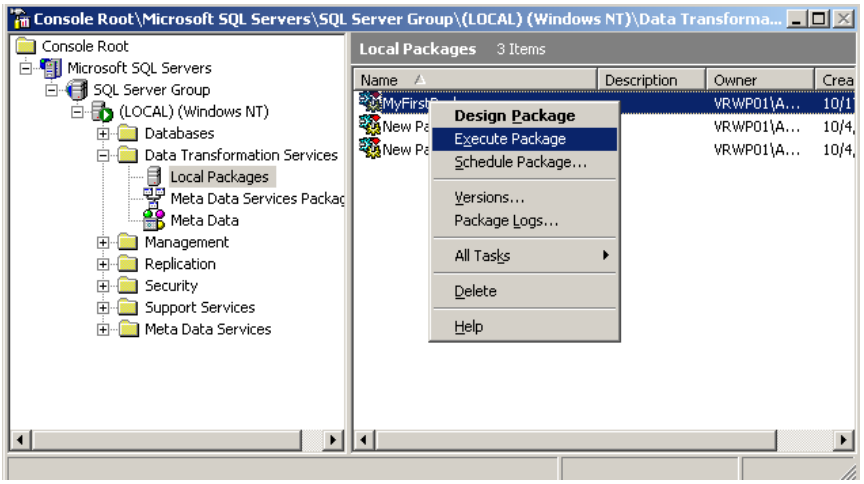
2. Click on **Package** menu and click **Save** to save the package.



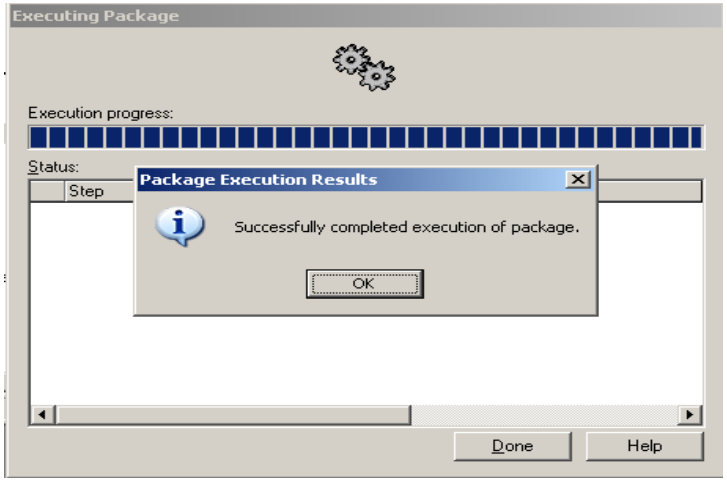
3. Enter **MyFirstPackage in** Package name and Click OK.



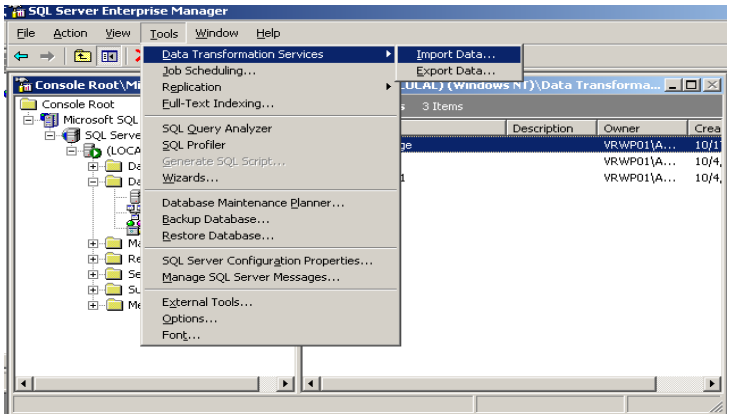
4. Right click on **MyFirstPackage** and select execute package option.



5. Click OK. Then Click Done.

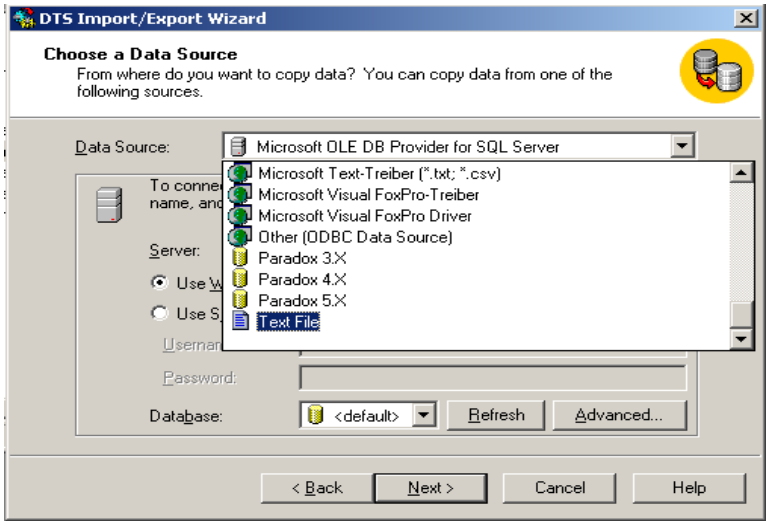


6. Click on **Tools Menu->Data Transformation Services->Import Data**.

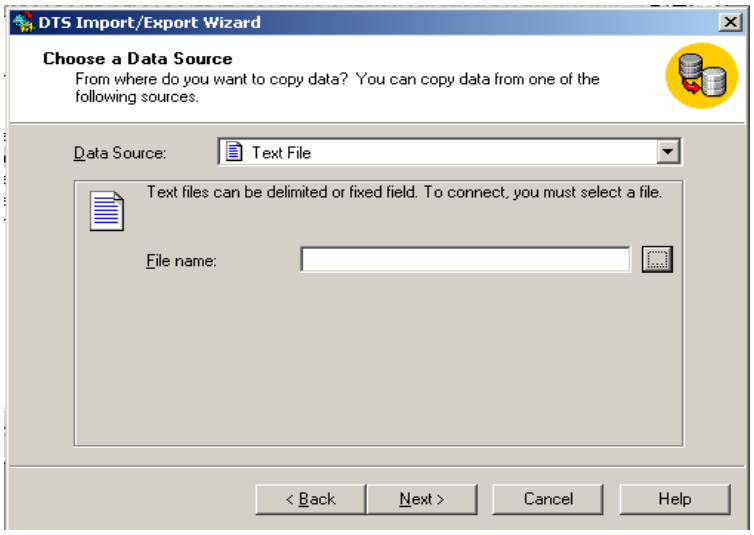


7. Click Next.

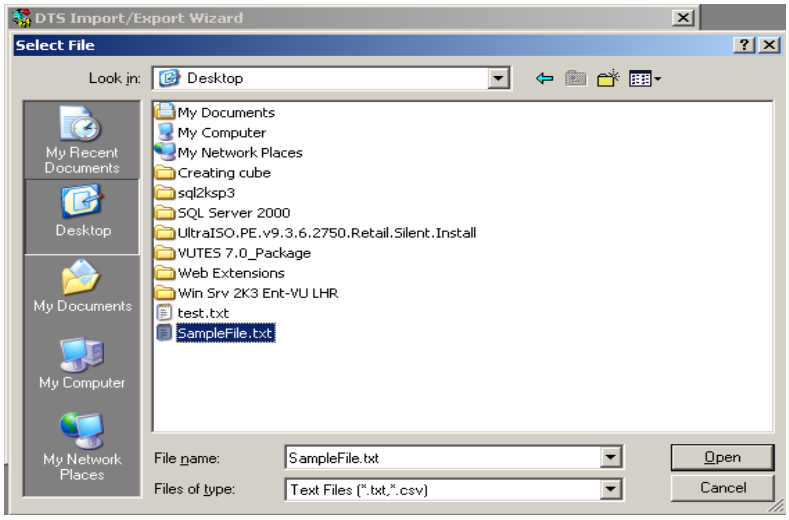
8. Select **Text File** as data source and click Next.



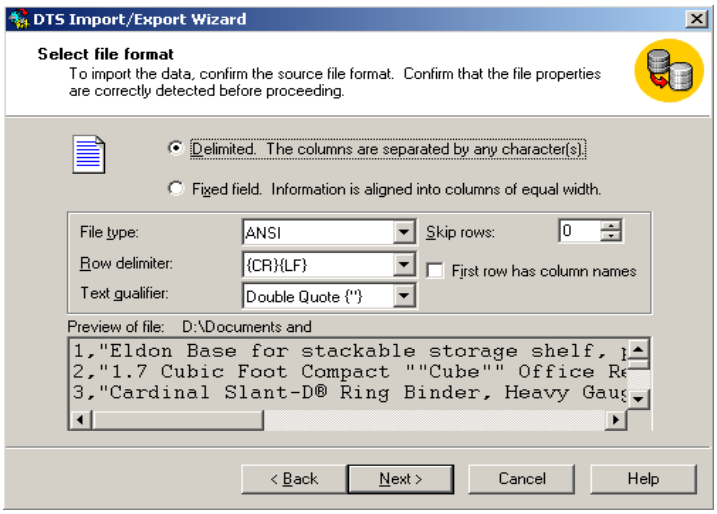
9. Select text file which you want to import and then click Open.



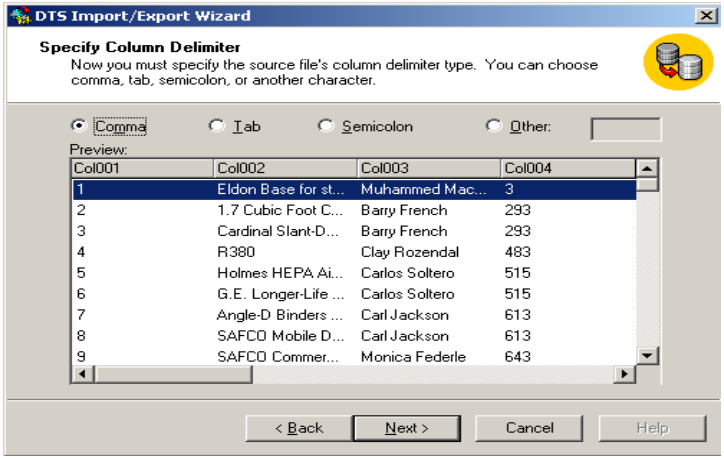
10. You can create a text on your system containing sample data. You can enter data about customers or students etc. Select the file and click Open. Then click **Next.**



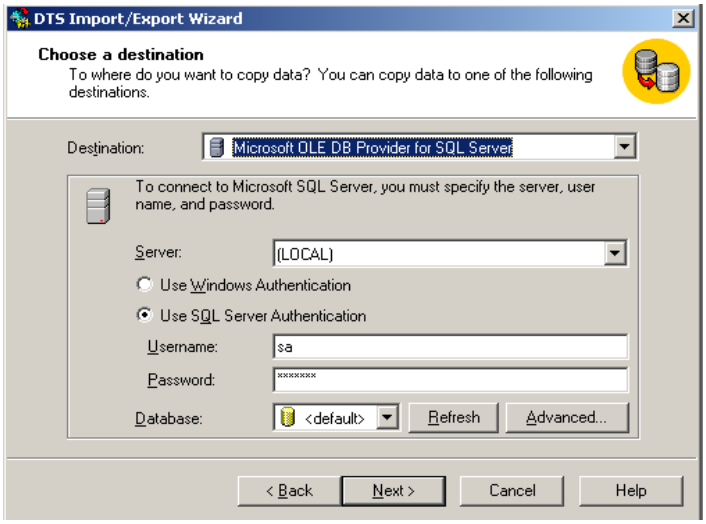
11. Click Next.



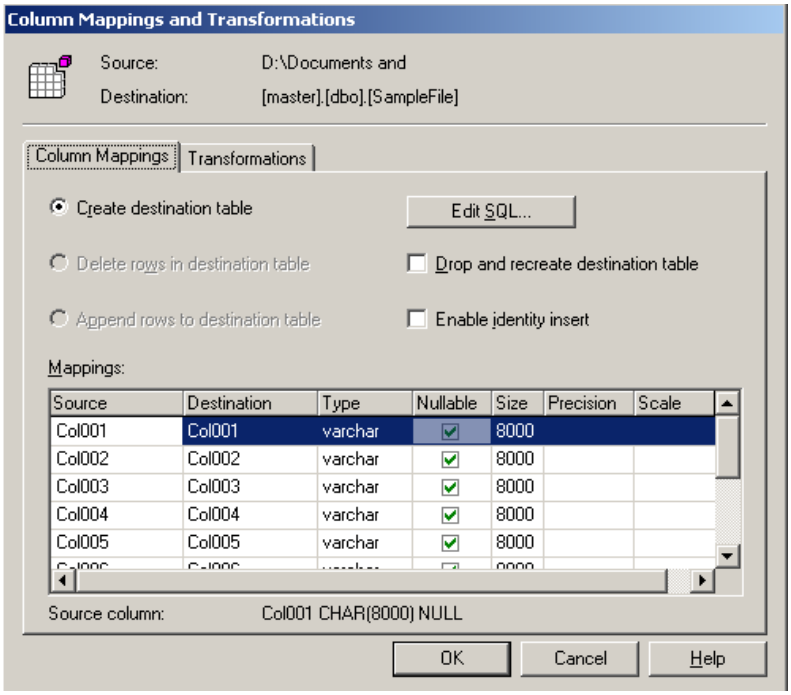
12. Click Next. Select Column Delimiter which in this case is comma. Click Next.

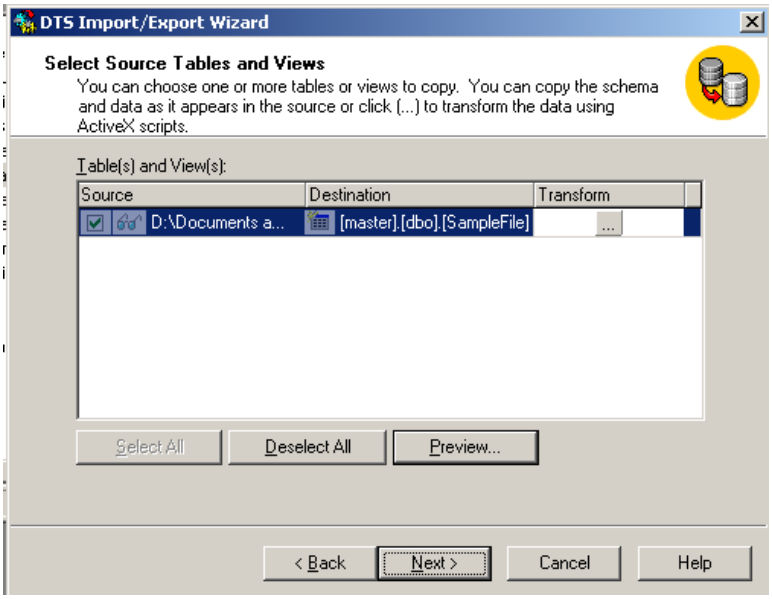


13. Choose destination and click Next.

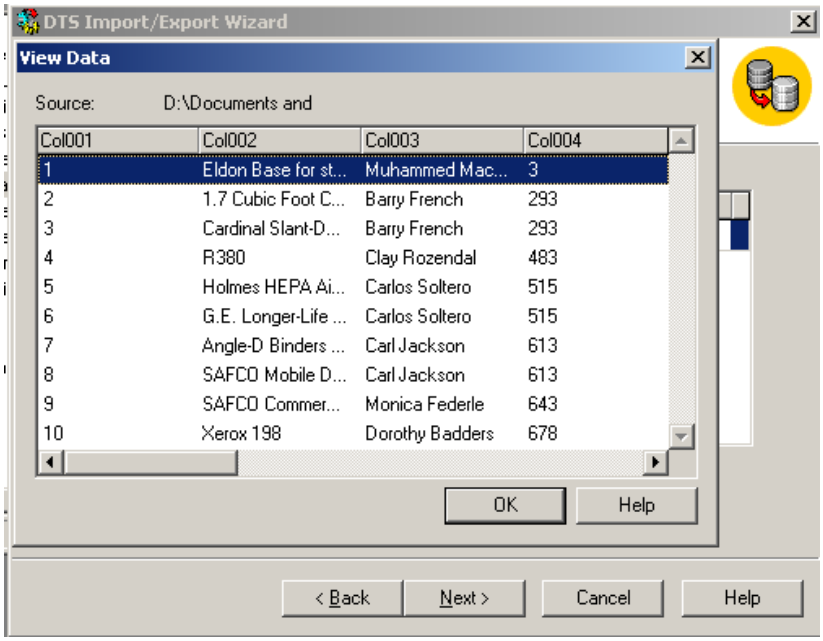


14. Select **Create destination table** option and create OK.

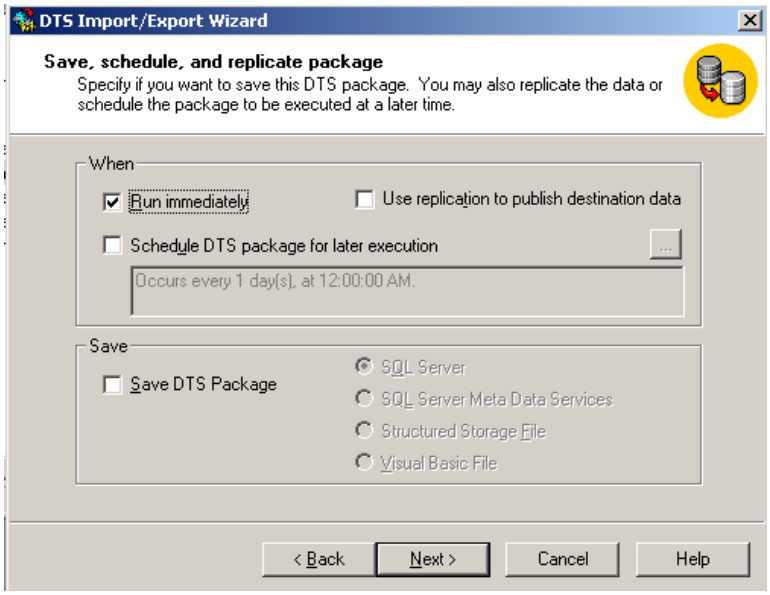




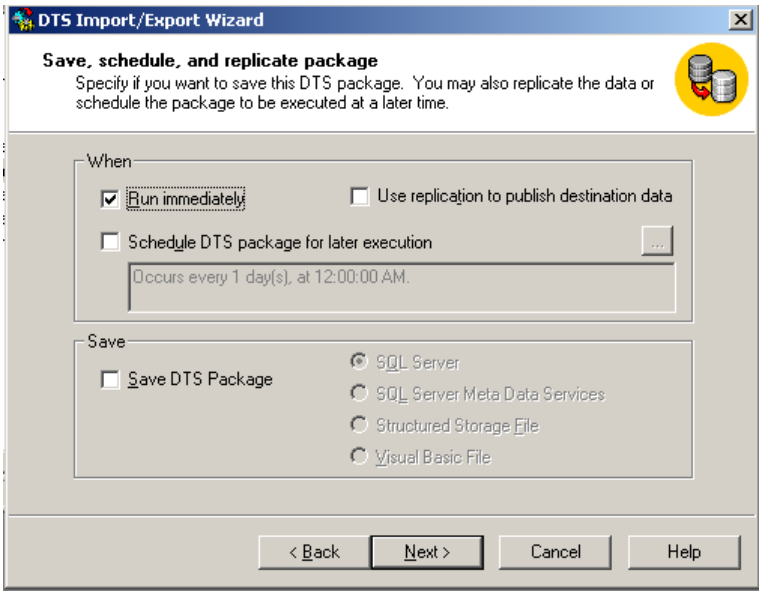
15. Click **Preview….** to view data.



16. Click Next.

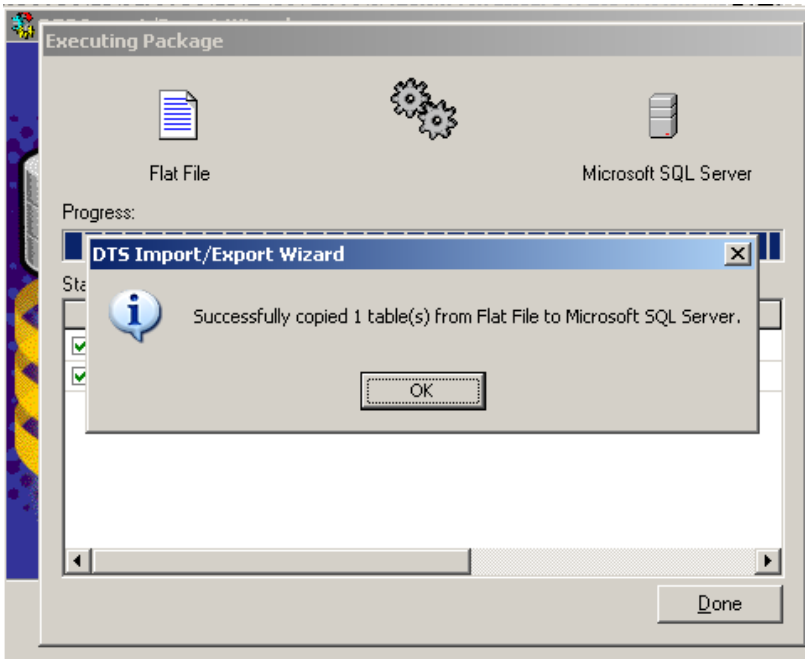


17. Select **Run immediately** and click Next.

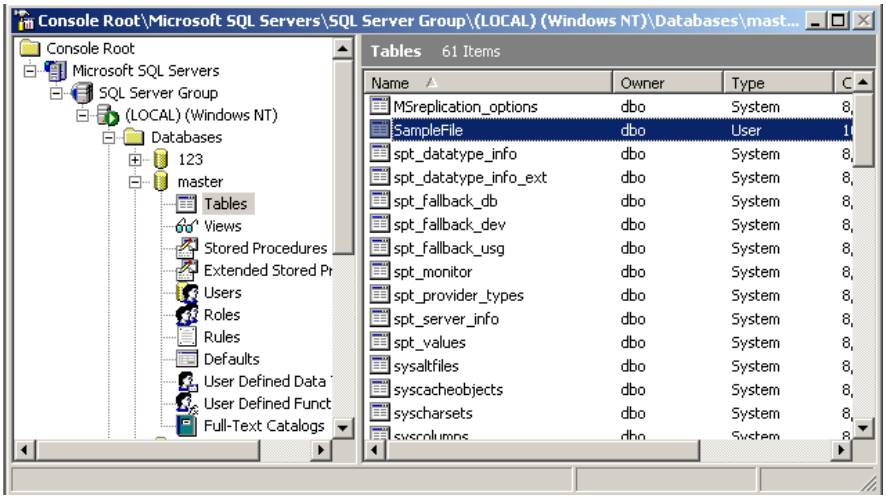


18. Click Finish.

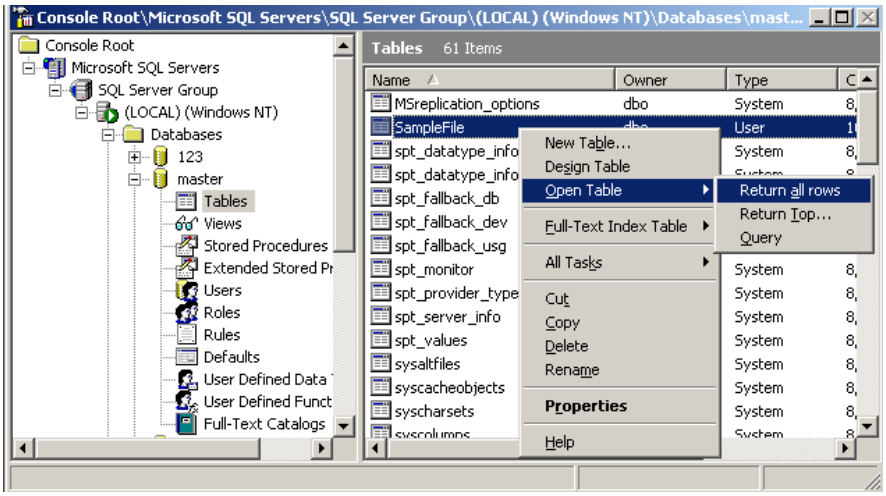
19. Click OK. Then Click Done.



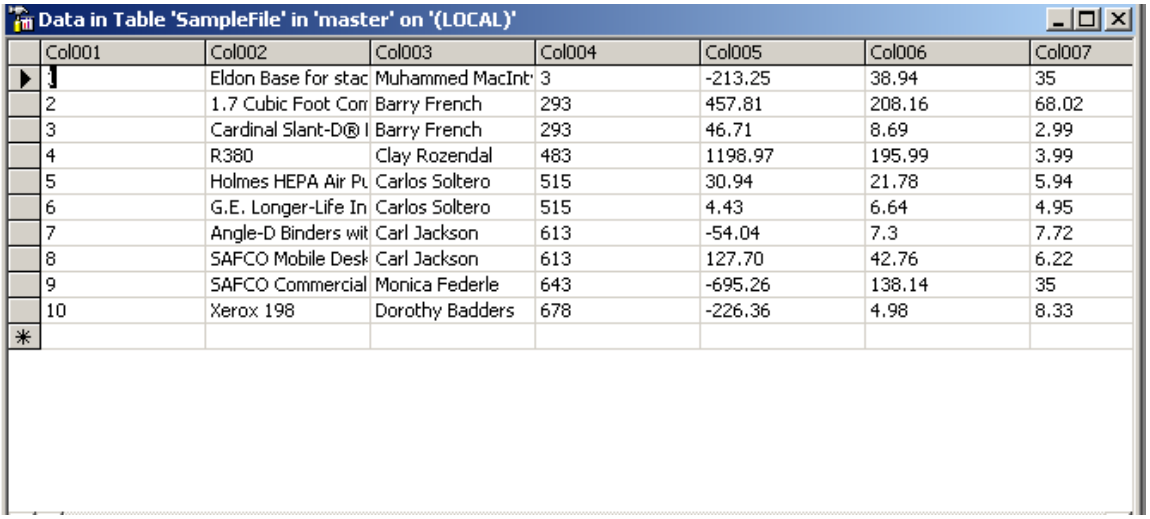
20. View your table in master table node.



21. To view data, right click on table SampleFile, click on Open Table->Return all rows.



22. This is data of table **SampleFile**.



	Col001	Col002	Col003	Col004	Col005	Col006	Col007
▶	1	Eldon Base for stac	Muhammed MacInt	3	-213.25	38.94	35
	2	1.7 Cubic Foot Com	Barry French	293	457.81	208.16	68.02
	3	Cardinal Slant-D® I	Barry French	293	46.71	8.69	2.99
	4	R380	Clay Rozendal	483	1198.97	195.99	3.99
	5	Holmes HEPA Air Pu	Carlos Soltero	515	30.94	21.78	5.94
	6	G.E. Longer-Life In	Carlos Soltero	515	4.43	6.64	4.95
	7	Angle-D Binders wit	Carl Jackson	613	-54.04	7.3	7.72
	8	SAFCO Mobile Desk	Carl Jackson	613	127.70	42.76	6.22
	9	SAFCO Commercial	Monica Federle	643	-695.26	138.14	35
	10	Xerox 198	Dorothy Badders	678	-226.36	4.98	8.33
*							

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 7

BASIC SORTED NEIGHBORHOOD (BSN) METHOD

PatientID	Patient Name	Patient Contact No	Address
F101	Sana	051-123456	Islamabad
M102	Ali	051-123456	Lahore
F103	Aliya		Rawalpindi
F104	Hina		Faisalabad
F103	Aliaa		Rawalpindi
F101	Sanah		Islamabad
M105	Hassan		Karachi

Question:

Consider the above table and apply all three steps (**Create key, Sort the data, Merge)** of Basic Sorted Neighborhood (BSN) method to find out the duplicate records in the table. Records will be considered duplicate if the value of column "PatientID" is same in this table data.

The steps which you have to follow are:

Key:

Key will consist of one character from "**PatientID**", then first three characters from **"Patient Name"** and then first two characters from **"Address"** column.

Step 1: Create key

In step-1, you will create the key according to the rules as mentioned above against each record. Add an extra column at the end of the table to show the new key created against each record.

Step 2: Sort the data

In step-2, you will sort the record on the basis of key which you created in step-1.

Step 3: Merge

In step-3, consider the window size (w) equal to two (2) and identify the similar records on the basis of sorted key.

Solution:

Step 1: Create key

Key is created in first step as per rules given in question statement.

PatientID	Patient Name	Patient Contact No	Address	Key
F01	Sana	051-123456	Islamabad	FSanIs
M01	Ali	051-123456	Lahore	MAliLa
F02	Aliya		Rawalpindi	FAliRa
F03	Hina		Faisalabad	FHinFa
F02	Aliaa		Rawalpindi	FAliRa
F01	Sanah		Islamabad	FSanIs
M02	Hassan		Karachi	MHasKa

Step 2: Sort the data

Now sort the records based on the above created keys

PatientID	Patient Name	Patient Contact No	Address	Key
F02	Aaliya		Rawalpindi	FAalRa
F02	Aaliaa		Rawalpindi	FAalRa
F03	Hina		Faisalabad	FHinFa
F01	Sana	051-123456	Islamabad	FSanIs
F01	Sanah		Islamabad	FSanIs
M01	Ali	051-123456	Lahore	MAliLa
M02	Hassan		Karachi	MHasKa

Step 3: Merge

Duplicate/Identical keys

PatientID	Patient Name	Patient Contact No	Address	Key
F02	Aaliya		Rawalpindi	FAalRa
F02	Aaliaa		Rawalpindi	FAalRa

F01	Sana	051-123456	Islamabad	FSanIs
F01	Sanah		Islamabad	FSanIs

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 8

DATA QUALITY RULES

"Pak Airline" is running in more than 15 countries and maintaining its data in a database. They have also started using data warehouse. They have different policies for running their business. One of their policies is that, if the airline itself cancels the flight due to weather or any other technical reason, then they have to payback 100% amount to customers and if flight is canceled by passenger then 40% amount will be returned. The Payback codes 'FP' or 'PP' are put in the payment claims to identify whether the claim is for full or partial payment respectively. Another policy is that allowed luggage weight per passenger is 25 kg.

There exists some specific data problems which are linked to business rules, and then generic and specific rule sets are developed for measuring how good the data is within an information system.
These rule sets shows the data quality metrics in order to judge conformance of data according to these business rules.

Note: Here, Assume, amount of ticket is Rs. 15,000 per passenger.

Question:

Considering above scenario, apply data quality rules on the historical data problems faced by "Pak Airline". For this, fill the following table (Rule Type, Generic Rule Set and Specific Rule Set) against each historical data problems as given below.

Historical Data Problem	Rule Type	Generic Rule Set	Specific Rule Set
Payback amount is less than 100% when flight is canceled due to weather or other technical reasons.			
The field "reason_cancel" were often left blank.			
The luggage per passenger exceeds 25 kg.			
The payback code in payback claim is not **'FP'** or **'PP'** sometimes.			

Solution:

Historical Data Problem	Rule Type	Generic Rule Set	Specific Rule Set
Payback amount is less than 100% when flight is canceled due to weather or other technical reasons.	Business Rule	If total payback amount is less than Rs. 15,000 and reason of flight cancelation is weather or technical, then, error.	Select total_payback_amount from Claim where total_payback_amount <15000 and reason_cancel='weather' or reason_cancel='technical';
The field "reason_cancel" were often left blank.	Null Constraints	If reason_cancel is blank or null, then, error	Select reason_cancel from Claim where reason_cancel= ' ' or reason_cancel =NULL;
The luggage per passenger exceeds 25 kg.	Operational Rule	If luggage_weight is greater than 25kg, then, error.	Select luggage_weight from Passenger where luggage_weight>25; Note: Assume that weight in field luggage_weight is input as kg.
The payback code in payback claim is not **'FP'** or **'PP'** sometimes.	Domain Validation	If payback_code is not 'FP' or 'PP', then, error.	Select payback_code from Claim where NOT (payback_code= 'FP' or 'PP');

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 9

KEY RANGE PARTITIONING

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Region	Product ID
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	South	FUR-BO-10001798
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	South	FUR-CH-10000454
CA-2016-138688	6/12/2016	6/16/2016	DV-13045	Darrin Van Huff	California	West	OFF-LA-10000240
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	South	FUR-TA-10000577
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	South	OFF-ST-10000760
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	FUR-FU-10001487
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	OFF-AR-10002833
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	TEC-PH-10002275
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	OFF-BI-10003910
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	OFF-AP-10002892
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	FUR-TA-10001539
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	TEC-PH-10002033
CA-2017-114412	4/15/2017	4/20/2017	AA-10480	Andrew Allen	North Carolina	South	OFF-PA-10002365
CA-2016-161389	12/5/2016	12/10/2016	IM-15070	Irene Maddox	Washington	West	OFF-BI-10003656
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	Central	OFF-AP-10002311
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	Central	OFF-BI-10000756
CA-2014-105893	11/11/2014	11/18/2014	PK-19075	Pete Kriz	Wisconsin	Central	OFF-ST-10004186
CA-2014-167164	5/13/2014	5/15/2014	AG-10270	Alejandro Grove	Utah	West	OFF-ST-10000107
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	West	OFF-AR-10003056
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	West	TEC-PH-10001949
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	West	OFF-BI-10002215
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	Central	OFF-AR-10000246
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	Central	OFF-AP-10001492
US-2017-156909	7/16/2017	7/18/2017	SF-20065	Sandra Flanagan	Pennsylvania	East	FUR-CH-10002774

PART I

Consider the subset of dataset data taken from https://community.tableau.com/docs/DOC-1236.

You are required to perform range partitioning on this data into yearly partitions of ship date.

Solution:

Partition 1 of year 2014:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Region	Product ID
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	FUR-FU-10001487
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	OFF-AR-10002833
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	TEC-PH-10002275
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	OFF-BI-10003910
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	OFF-AP-10002892
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	FUR-TA-10001539
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	West	TEC-PH-10002033
CA-2014-105893	11/11/2014	11/18/2014	PK-19075	Pete Kriz	Wisconsin	Central	OFF-ST-10004186
CA-2014-167164	5/13/2014	5/15/2014	AG-10270	Alejandro Grove	Utah	West	OFF-ST-10000107
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	West	OFF-AR-10003056
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	West	TEC-PH-10001949
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	West	OFF-BI-10002215

Partition 2 of year 2015:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Region	Product ID
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	South	FUR-TA-10000577
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	South	OFF-ST-10000760
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	Central	OFF-AP-10002311
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	Central	OFF-BI-10000756

Partition 3 of year 2016:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Region	Product ID
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	South	FUR-BO-10001798
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	South	FUR-CH-10000454
CA-2016-138688	6/12/2016	6/16/2016	DV-13045	Darrin Van Huff	California	West	OFF-LA-10000240
CA-2016-161389	12/5/2016	12/10/2016	IM-15070	Irene Maddox	Washington	West	OFF-BI-10003656
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	Central	OFF-AR-10000246
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	Central	OFF-AP-10001492

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Region	Product ID
CA-2017-114412	4/15/2017	4/20/2017	AA-10480	Andrew Allen	North Carolina	South	OFF-PA-10002365
US-2017-156909	7/16/2017	7/18/2017	SF-20065	Sandra Flanagan	Pennsylvania	East	FUR-CH-10002774

PART II

Perform list portioning by Partitioning by following sales table by region. You can use values of region from table given in **Question 1**.

Solution:

First you will define values for region partitions.

Region partitions will be **East**, **West**, **South** and **Central Partition**.

List of East partition values: Pennsylvania

List of West Partition Values: Utah, California, Washington

List of South Partition Values: Florida, Kentucky, North Carolina

List of Central Partition Values: Wisconsin, Texas, Nebraska

South Partition:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Product ID
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	FUR-BO-10001798
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	FUR-CH-10000454
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	FUR-TA-10000577
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	OFF-ST-10000760
CA-2017-114412	4/15/2017	4/20/2017	AA-10480	Andrew Allen	North Carolina	OFF-PA-10002365

Central Partition:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Product ID
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	FUR-BO-10001798
CA-2016-152156	11/8/2016	11/11/2016	CG-12520	Claire Gute	Kentucky	FUR-CH-10000454
CA-2016-138688	6/12/2016	6/16/2016	DV-13045	Darrin Van Huff	California	OFF-LA-10000240
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	FUR-TA-10000577
US-2015-108966	10/11/2015	10/18/2015	SO-20335	Sean O'Donnell	Florida	OFF-ST-10000760
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	FUR-FU-10001487
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	OFF-AR-10002833
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	TEC-PH-10002275
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	OFF-BI-10003910
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	OFF-AP-10002892
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	FUR-TA-10001539
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	TEC-PH-10002033
CA-2017-114412	4/15/2017	4/20/2017	AA-10480	Andrew Allen	North Carolina	OFF-PA-10002365
CA-2016-161389	12/5/2016	12/10/2016	IM-15070	Irene Maddox	Washington	OFF-BI-10003656
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	OFF-AP-10002311

US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	OFF-BI-10000756
CA-2014-105893	11/11/2014	11/18/2014	PK-19075	Pete Kriz	Wisconsin	OFF-ST-10004186
CA-2014-167164	5/13/2014	5/15/2014	AG-10270	Alejandro Grove	Utah	OFF-ST-10000107
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	OFF-AR-10003056
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	TEC-PH-10001949
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	OFF-BI-10002215
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	OFF-AR-10000246
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	OFF-AP-10001492
US-2017-156909	7/16/2017	7/18/2017	SF-20065	Sandra Flanagan	Pennsylvania	FUR-CH-10002774

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Region	Product ID
US-2017-156909	7/16/2017	7/18/2017	SF-20065	Sandra Flanagan	Pennsylvania	East	FUR-CH-10002774

West Partition:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Product ID
CA-2016-138688	6/12/2016	6/16/2016	DV-13045	Darrin Van Huff	California	OFF-LA-10000240

CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	FUR-FU-10001487
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	OFF-AR-10002833
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	TEC-PH-10002275
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	OFF-BI-10003910
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	OFF-AP-10002892
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	FUR-TA-10001539
CA-2014-115812	6/9/2014	6/14/2014	BH-11710	Brosina Hoffman	California	TEC-PH-10002033
CA-2016-161389	12/5/2016	12/10/2016	IM-15070	Irene Maddox	Washington	OFF-BI-10003656
CA-2014-167164	5/13/2014	5/15/2014	AG-10270	Alejandro Grove	Utah	OFF-ST-10000107
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	OFF-AR-10003056
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	TEC-PH-10001949
CA-2014-143336	8/27/2014	9/1/2014	ZD-21925	Zuschuss Donatelli	California	OFF-BI-10002215

East Partition:

Order ID	Order Date	Ship Date	Customer ID	Customer Name	State	Product ID
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	OFF-AP-10002311
US-2015-118983	11/22/2015	11/26/2015	HP-14815	Harold Pawlan	Texas	OFF-BI-10000756
CA-2014-105893	11/11/2014	11/18/2014	PK-19075	Pete Kriz	Wisconsin	OFF-ST-10004186
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	OFF-AR-10000246
CA-2016-137330	12/9/2016	12/13/2016	KB-16585	Ken Black	Nebraska	OFF-AP-10001492

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 10

CLUSTER INDEX

"Pak Airline" is an airliner reservation company, which is operating in more than 10 countries. They have developed the airline reservation system to avoid the errors faced in manual system. The staff of the airline use airline reservation system form the tasks such as flight scheduling, ticket reservation, announcements in automated way. Similarly, users/passengers can search for flight schedule according to date and time and fare details. The staff of the airline can manage the reservation systems by flight rout, runway details, flight scheduling and reservation.

Ticket reservation system of the Pak Airline provides the information about schedule of flights, availability of seats, flight number and destination. For reservation of ticket user have to provide its personal information such as name, age, address etc. For payment purpose user will provide credit card number and bank details. Moreover, information about flight number, date of departure, no. of tickets to be booked is also required for confirmation of ticket.

Question Statement:

You are required to create Cluster Index based on destination city, Designation of staff, and address of passenger. For this consider the following tables and apply Cluster indexing technique on required columns.

Aiport-Flight Table

Airport Id	**Flight Id**	**Arrival date**	**Arrival time**	**Depart Date**	**Depart Time**	**Destination**	**Aiport Name**	**City**
AP1	Pk01	15-10-2018	12:30 PM	16-10-2018	10:00 AM	United Kingdom	Allama Iqbal Airport	Lahore
AP2	Pk02	19-11-2018	10:00 AM	19-11-2018	04:00 PM	Dubai	Baynazir Airport	Rawalpindi
AP3	Bg01	16-10-2018	12:30 PM	16-10-2018	06:30 PM	KSA	New Islamabad Airport	Islamabad
Ap4	Pk03	17-10-2018	09:00 PM	17-10-2018	04:00 PM	United Kingdom	Jinnah Airport	Karachi
Ap5	Bg02	25-10-2018	08:00 AM	26-10-018	12:00 PM	USA	New Lahore Airport	Lahore
Ap6	Bg03	19-11-2018	10:00 AM	19-11-2018	04:00 PM	Dubai	International Quetta Airport	Quetta

Staff Table

Staff Id	**Name**	**Designation**	**Contact No**
St01	Micheal	Operation Agent	0092-345-7865439
St02	Jackob	Flight Attendant	0062-876-0987654
St03	Alfard	Avionic Technician	0092-321-9865321
St04	Jackson	Operation Agent	0072-098-7854321
St05	Joseph	Flight Dispatcher	0092-333-9054213
St06	Joliana	Passenger Service Agent	0062-900-6789012
St07	Thomsan	Flight Attendant	0052-321-9084563

Passenger Table

Passenger Id	**Name**	**Contact No**	**Address**	**Email**	**Credit Cr.No**
Ps01	Julia	0092-345-7865439	Islamabad	julia@gmail.com	123-987
Ps02	Alexandra	0062-876-0987654	Dubai	alex@gmail.com	324-908
Ps03	Robert	0092-321-9865321	London	robert@live.com	457-975
Ps04	Alaf	0072-098-7854321	Islamabad	alaf@yahoo.com	345-075
Ps05	Julia Sanf	0092-333-9054213	New York	jausanf@gmail.com	123-890
Ps06	Charistea	0062-900-6789012	London	charist@live.com	768-054

Solution:

Cluster index on **Destination** column

Airport Id	**Flight Id**	**Arrival date**	**Arrival time**	**Depart Date**	**Depart Time**	**Destination**	**Aiport Name**	**City**
AP2	Pk02	19-11-2018	10:00 AM	19-11-2018	4:00 PM	Dubai	Baynazir Airport	Rawalpindi
Ap6	Bg03	19-11-2018	10:00 AM	19-11-2018	4:00 PM	Dubai	International Quetta Airport	Quetta
AP3	Bg01	16-10-2018	12:30 PM	16-10-2018	16-10-2018	KSA	New Islamabad Airport	Islamabad
AP1	Pk01	15-10-2018	12:30 PM	16-10-2018	10:00 AM	United Kingdom	Allama Iqbal Airport	Lahore
Ap4	Pk03	17-10-2018	9:00 PM	17-10-2018	4:00 PM	United Kingdom	Jinnah Airport	Karachi
Ap5	Bg02	25-10-2018	8:00 AM	26-10-018	12:00 PM	USA	New Lahore Airport	Lahore

Cluster index on **Designation** column

Staff Id	**Name**	**Designation**	**Contact No**
St03	Alfard	Avionic Technician	0092-321-9865321
St02	Jackob	Flight Attendant	0062-876-0987654
St07	Thomsan	Flight Attendant	0052-321-9084563
St05	Joseph	Flight Dispatcher	0092-333-9054213
St01	Micheal	Operation Agent	0092-345-7865439
St04	Jackson	Operation Agent	0072-098-7854321
St06	Joliana	Passenger Service Agent	0062-900-6789012

Cluster index on **Address** column

Passenger Id	**Name**	**Contact No**	**Address**	**Email**	**Credit Cr.No**
Ps01	Julia	0092-345-7865439	Islamabad	julia@gmail.com	123-987
Ps04	Alaf	0072-098-7854321	Islamabad	alaf@yahoo.com	345-075
Ps02	Alexandra	0062-876-0987654	Dubai	alex@gmail.com	324-908
Ps03	Robert	0092-321-9865321	London	robert@live.com	457-975
Ps06	Charistea	0062-900-6789012	London	charist@live.com	768-054
Ps05	Julia Sanf	0092-333-9054213	New York	jausanf@gmail.com	123-890

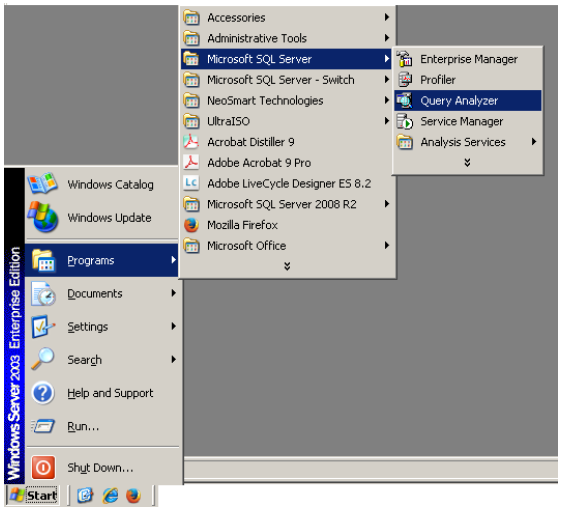
Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

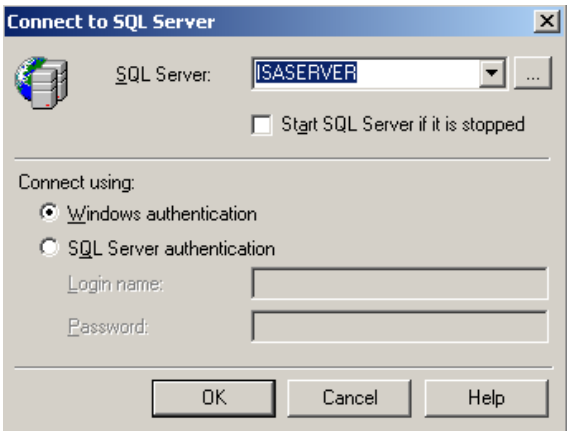
Lab 11

Nested Loop, Sort Merge, and Hash Join using SQL Server Query Analyzer

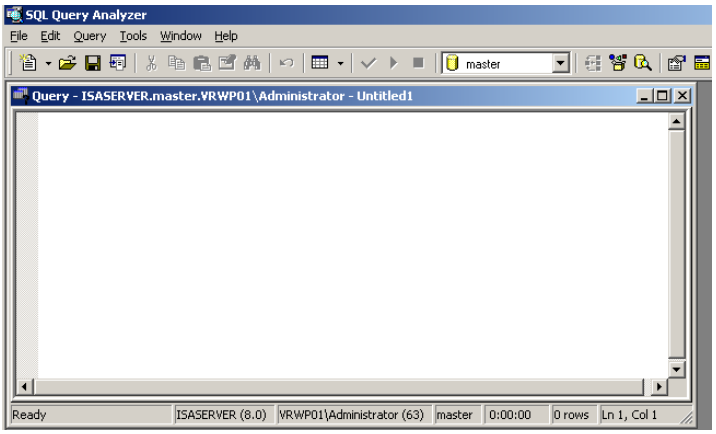
1. Open SQL Query Analyzer by clicking Programs->Microsoft SQL Server and Query Analyzer.



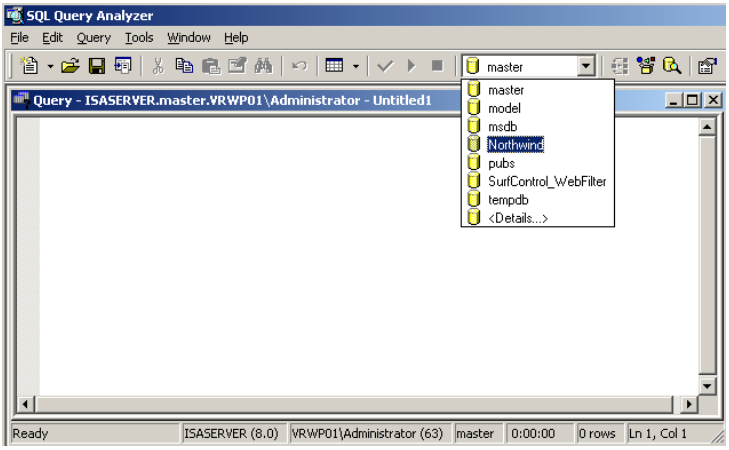
2. Select SQL Server **ISASERVER** and click oK.



3. The following SQL Query Analyzer window will open. In this window, you will write SQL queries.



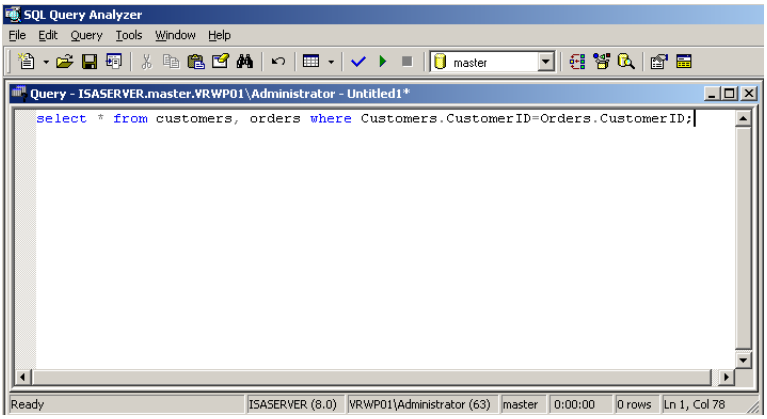
4. First select sample database **Northwind.**



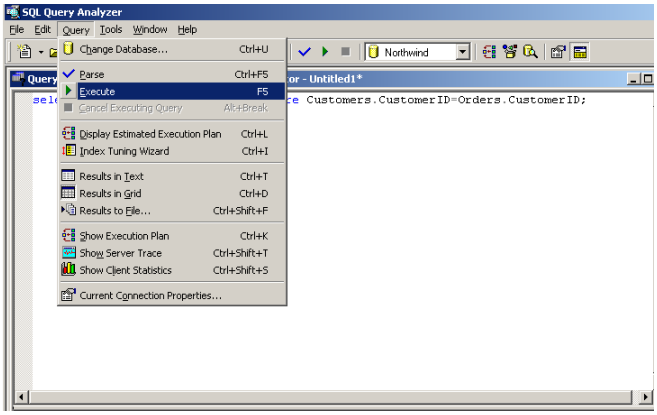
5. Write following SQL query in query analyzer window.

   select * from customers, orders where Customers.CustomerID=Orders.CustomerID;

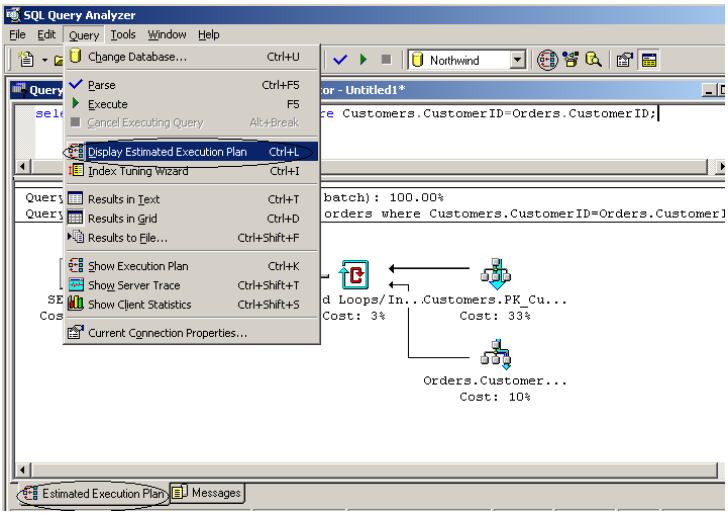
   Here customers and orders are table of **Northwind** database**.**



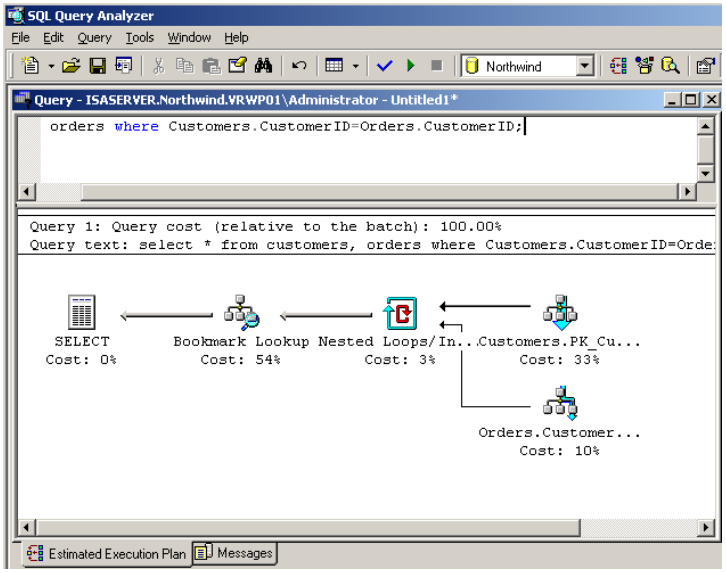
6. Execute SQL query by clicking **Query** menu and select **Execute** option. You can also click execute button available on standard toolbar.



7. Click on **Query** menu and select option **Display Estimated Execution Plan** to see execution plan of query. Estimated execution plan can also be viewed by using highlighted options in following figure.



8. By default, Nested loop join is performed, you are required to analyze execution plan.

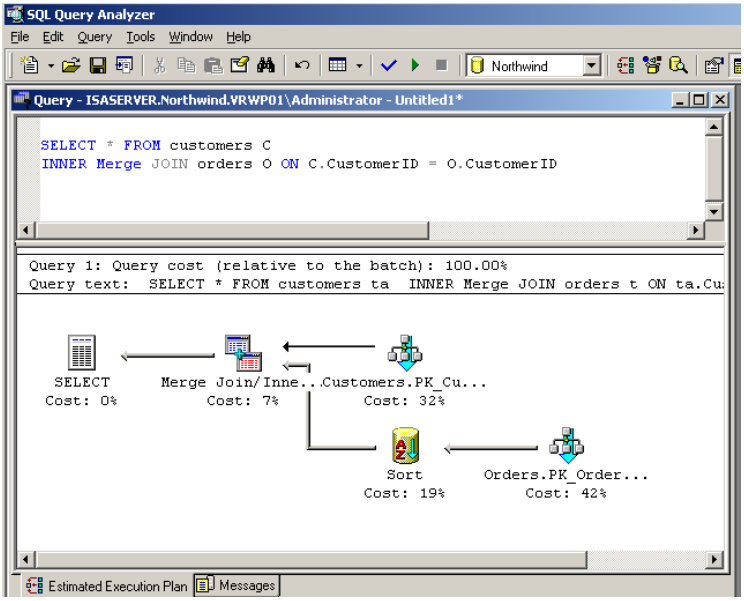


Merge Join Query:

For Merge Join, write following query in SQL Query Analyzer window.

SELECT * FROM customers C

INNER Merge JOIN orders O ON C.CustomerID = O.CustomerID

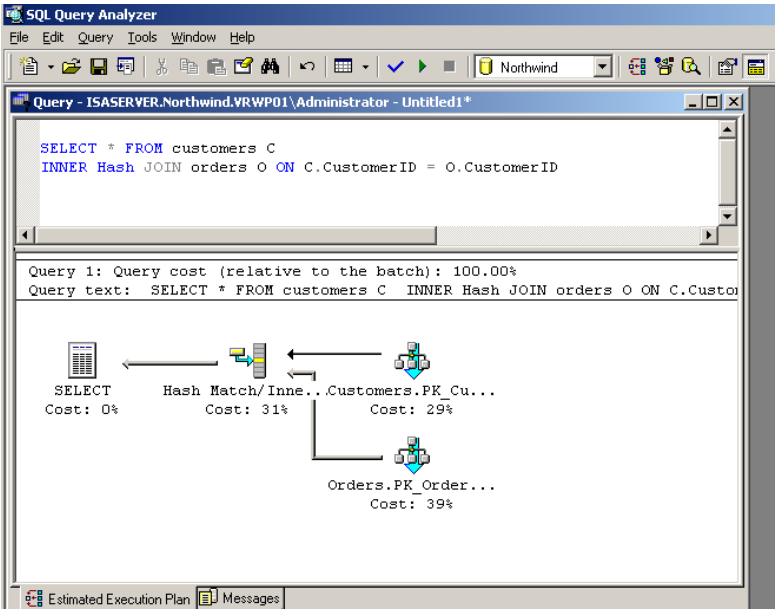


Hash Join Query:

For Merge Join, write following query in SQL Query Analyzer window.

SELECT * FROM customers C

INNER Hash JOIN orders O ON C.CustomerID = O.CustomerID



Lab Exercise: You are required to analyze Nested Loop Join, Merge Join and Hash Join for the given query in terms of which one is efficient in terms of execution time.

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 12

DWH IMPLEMENTATION: Goal Driven Approach

Ralph Kimball's Approach Part I

In this Lab, you have to perform the second part of the semester project which details have been provided in lecture 36. This lab deals with the DWH implementation life cycle that you have already studied in great detail in lectures 32-35.

In lectures 33and 34, you have studied the **Ralph Kimball's approach for data warehouse implementation**. You will use this approach in this lab. Note that, you are not required to do any development and deployment work. As you have studied in lectures that the DWH lifecycle road-map was divided into three parts, you only have to cover these parts i.e. **(i)project planning (ii) user requirement definition and (iii) three parallel tracks**. You are NOT required to discuss or do DWH deployment or do any analytics development.

In this lab, you are required to perform following two tasks:

Task 1: Identify Organization

- Do a complete data warehouse implementation life cycle study.
- Identify a large company/organization that is a prime candidate for a DWH.
- Prepare report_1 giving and explaining any four reasons for selecting a company.
- Submit report_1 and get the company/organization selected approved by the instructor before proceeding ahead.

Task 2: Project Planning

After identifying organization, perform the following task as part of Project Planning. For this,

- Prepare a questionnaire (at least 15 non-trivial questions).
- Identify and contact a key person who will help you.
- Prepare and Submit report_2

Your report should include following:

- Report No.
- Title of course, semester & submission date
- Your Name and roll no.
- Campus and name of city.
- Table of contents.
- 1-page executive summary of the report.
- Description of task 1 and task2.
- Attach (scanned) hard/soft copies of all related material collected and referenced.

Mechanism to Conduct Lab:

Students and teacher communicate through Adobe Connect.

Lab 13

DWH IMPLEMENTATION: Goal Driven Approach

Ralph Kimball's Approach Part II

User requirement definition

- Set an appointment to meet business users.
- Collect answers to questions from business users to understand business requirements (Use your questionnaire which you have prepared in previous lab).
- Compile report of interview and document it in meaningful way.
- Identify business processes.
- Identify Requirements of key processes.
- Prepare report_3 of interview report, business processes and requirements of key business processes.

- Submit report_3.

Note: For details of above tasks, watch video lecture 33 and 36.

Your report should include following:

- Report No.
- Title of course, semester & submission date
- Your Name and roll no.
- Campus and name of city.
- Table of contents.
- 1-page executive summary of the report.
- Interview report, identified business processes and requirements of key business processes.
- Attach (scanned) hard/soft copies of all related material collected and referenced.

Mechanism to Conduct Lab:

- Students and teacher communicate through Adobe Connect.

Lab 14

DIMENSION MODEL OF AIRLINE DATA WAREHOUSE

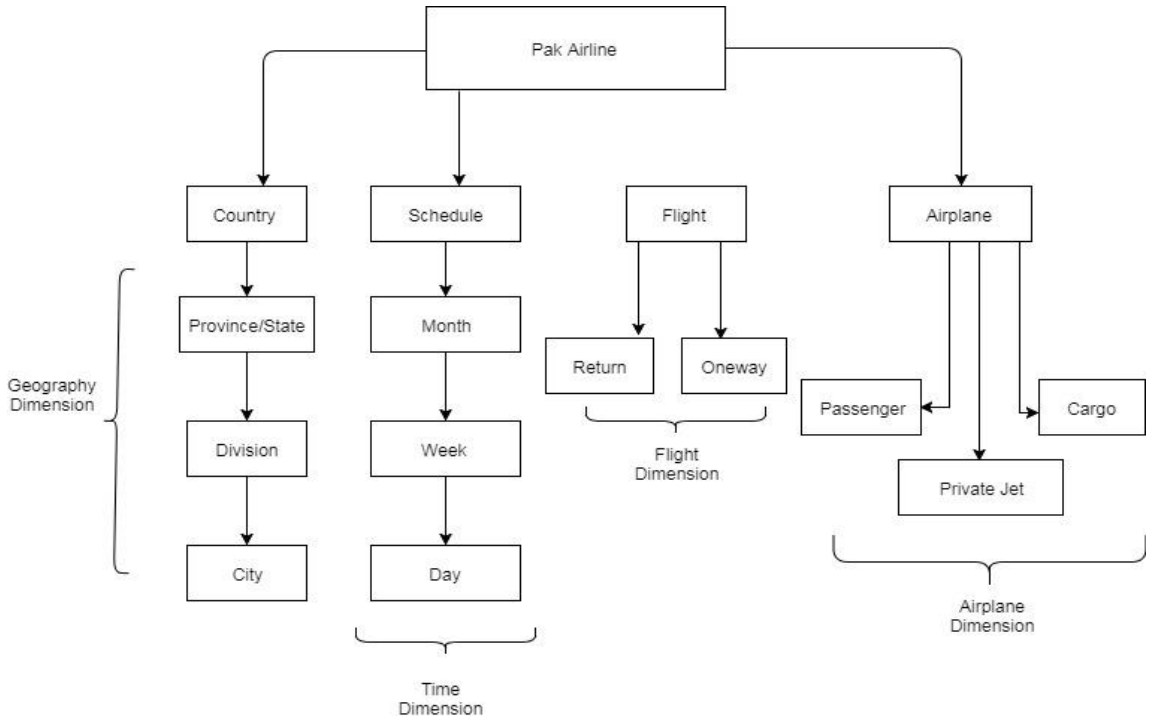
Scenario

"Pak Airline" is an airliner reservation company, which is operating in more than 10 countries. They have developed the airline reservation system to avoid the errors faced in manual system. The staff of the airline use airline reservation system form the tasks such as flight scheduling, ticket reservation, announcements in automated way. Similarly, users/passengers can search for flight schedule according to date and time and fare details. The staff of the airline can manage the reservation systems by flight rout, runway details, flight scheduling and reservation.

Ticket reservation system of the Pak Airline provides the information about schedule of flights, availability of seats, flight number and destination. For reservation of ticket user have to provide its personal information such as name, age, address etc. For payment purpose user will provide credit card number and bank details. Moreover, information about flight number, date of departure, no. of tickets to be booked is also required for confirmation of ticket. Following is the ERD of above airline reservation system.

Question: You are required to draw the Dimension model for the airline data warehouse following above given scenario.

Solution:

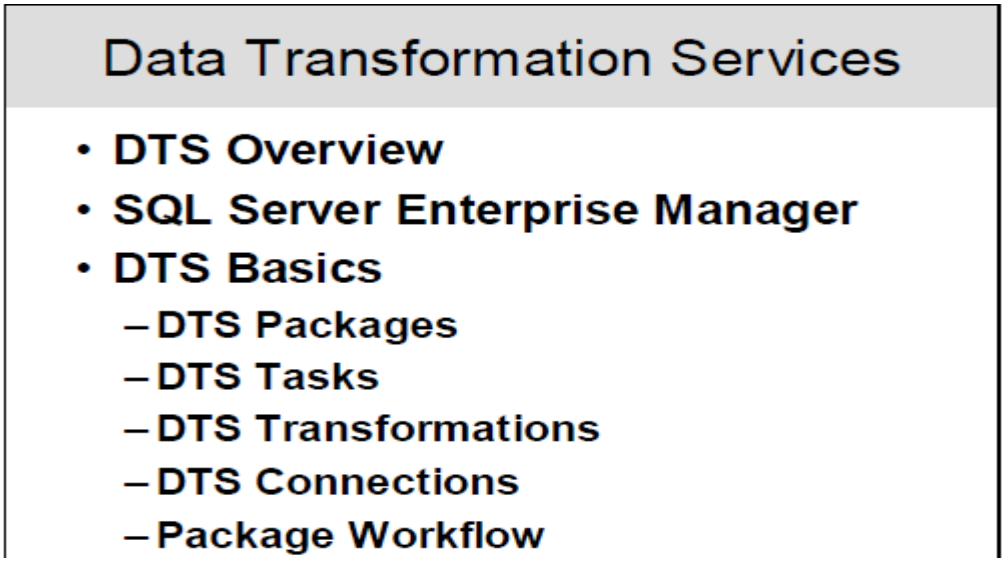


Mechanism to Conduct Lab:

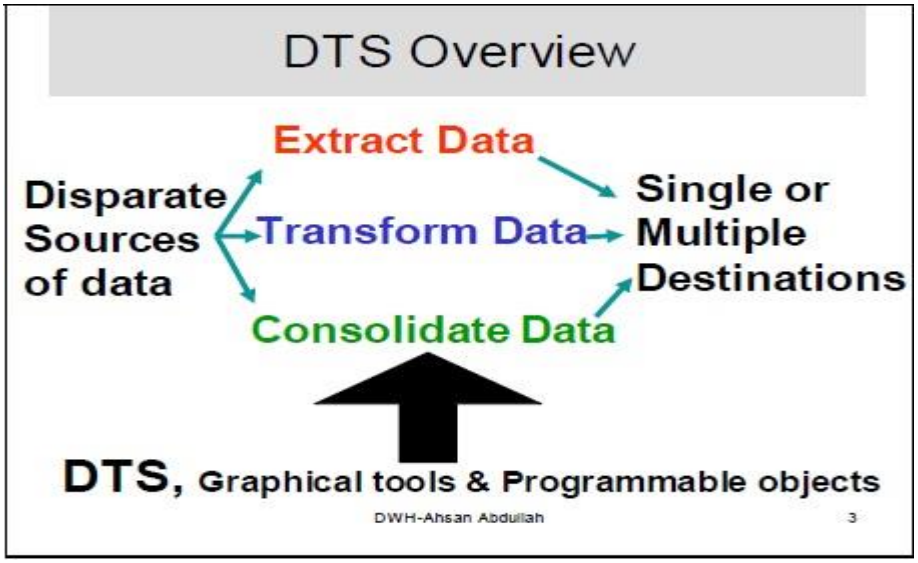
Students and teacher communicate through Adobe Connect.

Lab 15 Part I

DATA TRANSFER SERVICE (STS)



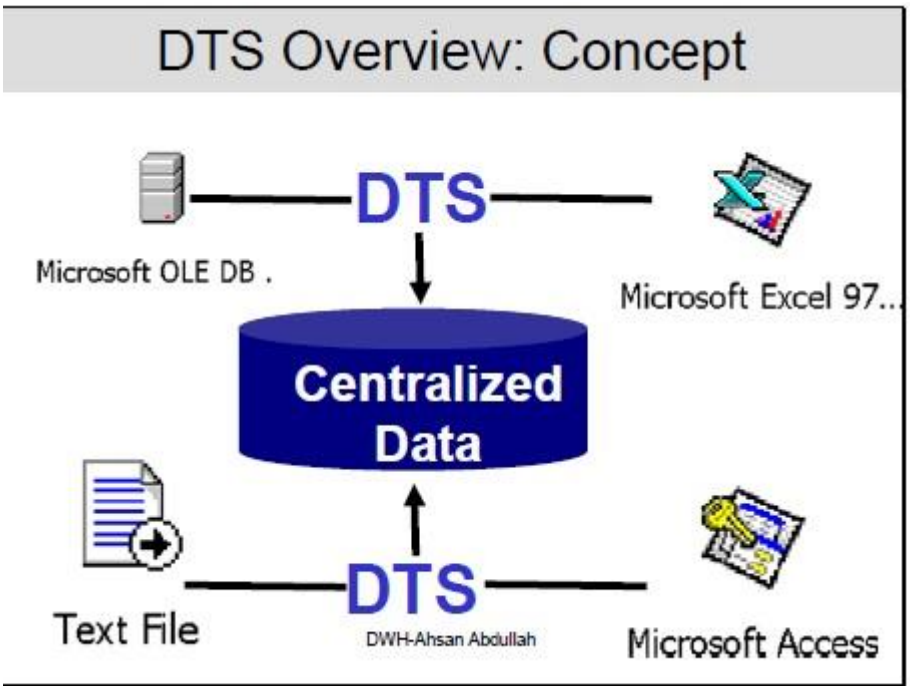
Microsoft® SQL Server™ 2000 Data Transformation Services (DTS) is a set of graphical tools and programmable objects that allow you extract, transform, and consolidate data from disparate sources into single or multiple destinations. SQL Server Enterprise Manager provides an easy access to the tools of DTS.
The purpose of this lecture is to get an understanding of DTS basics, which is necessary to learn the use of DTS tools. These DTS basics describe the capabilities of DTS and summarize the business problems it addresses.



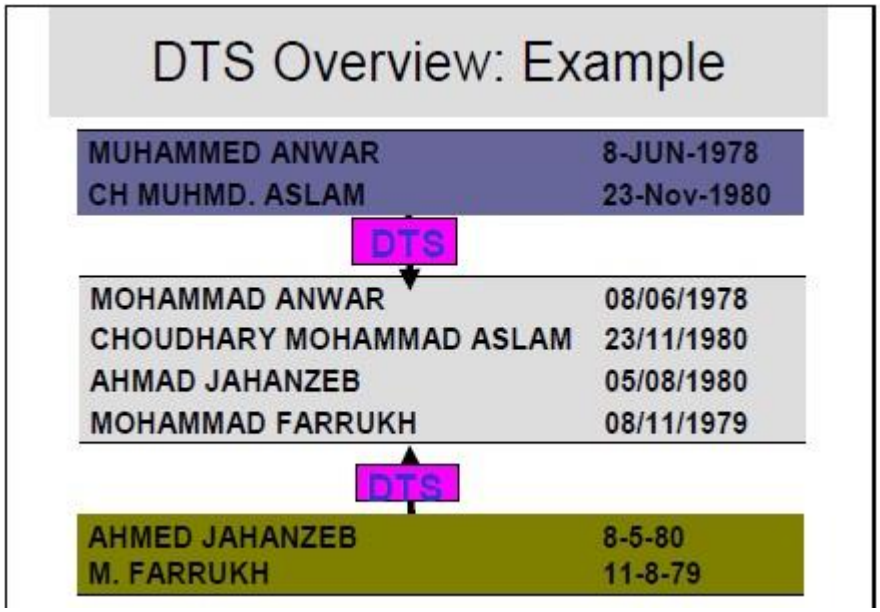
Many organizations need to centralize data to improve corporate decision-making. However, their data may be stored in a variety of formats and in different locations. Data Transformation Services (DTS) address this vital business need by providing a set of tools that let you extract, transform, and consolidate data from disparate sources into single or multiple destinations supported by DTS connectivity.

DTS allows us to connect through any data source or destination that is supported by OLE DB. This wide range of connectivity that is provided by DTS allows us to extract data from wide range of legacy systems. Heterogeneous source systems store data with their local formats and conventions. While consolidating data from variety of sources we need to transform names, addresses, dates etc into a standard format. For example consider a student record management system of a university having four campuses. A campus say ‘A’ follows convention to store city codes “LHR” for Lahore. Another campus say ‘B’ stores names of cities “Lahore”, campus ‘C’ stores city names in block letters ‘LAHORE’, and the last campus ‘D’ store city names as ‘lahore’. When the data from all the four campuses is combined as it is and query is run “How many students belong to ‘Lahore’?” We get the answer only from campus B because no other convention for Lahore matches to the one in query.

To combine data from heterogeneous sources with the purpose of some useful analysis requires transformation of data. Transformation brings data in some standard format. Microsoft SQL Server provides graphical tools to build DTS packages. These tools provide good support for transformations. Complex transformations are achieved through VB Script or Java Script that is loaded in DTS package. Package can also be programmed by using DTS object model instead of using graphical tools but DTS programming is rather complicated.

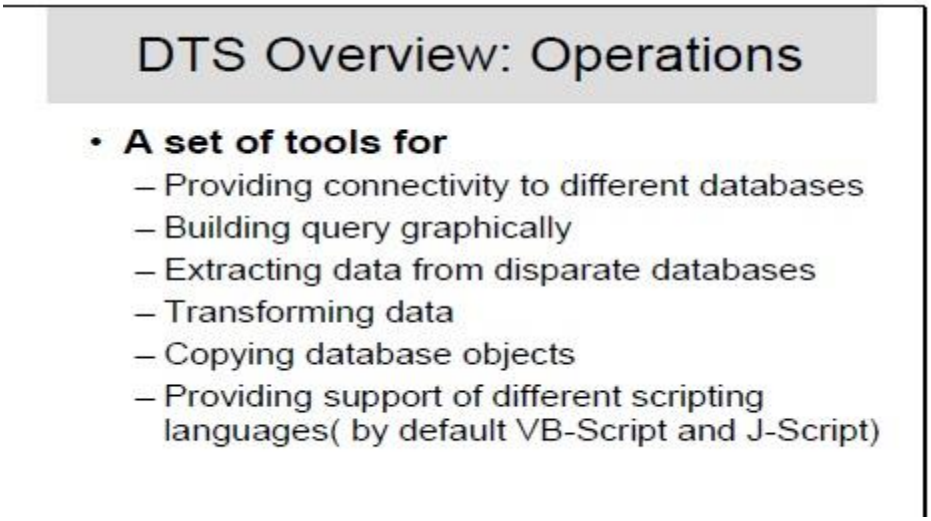


The slide shows the heterogeneous sources of data. Position of DTS while consolidating the data into a single source is also clear from the slide. In legacy systems we may come across the text files as a source of data. Microsoft Access is a database management system, maintains data in tables, and columns validate the input to the system. We often find legal values are stored in these sort of data management systems. But when we deal with text files no validation mechanism for input is there. Therefore we may come across illegal and rubbish values in text files. This makes the process of transformation further complicated.



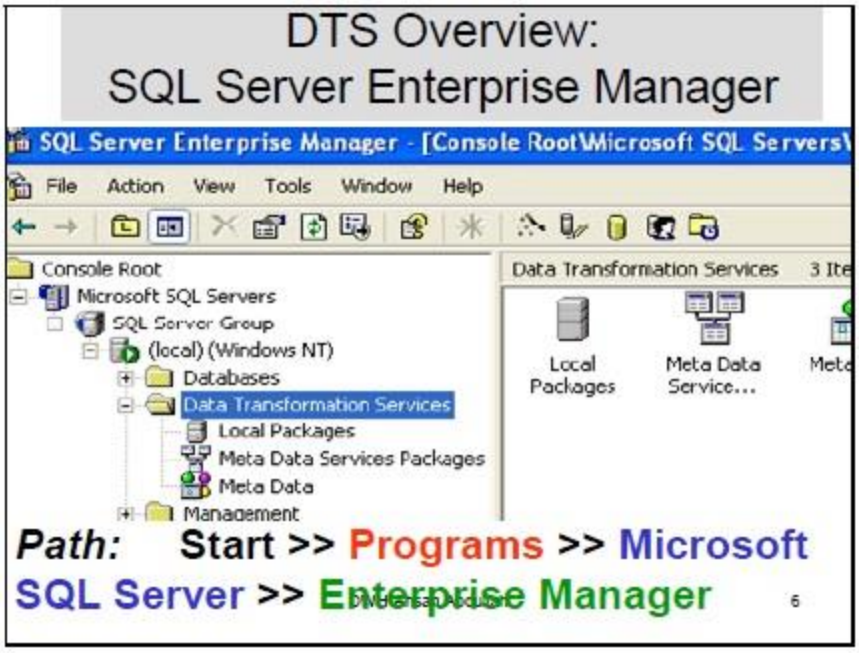
In this slide we may see three data management systems. Data is extracted from two systems, top and bottom, and is loaded into the standardized system shown in the middle. We may see two transformations over here.

First one is name transformation and the other one is date transformation. In the database management system shown at the top we have two names Muhammed Anwer and Choudhary Mohammed Aslam. Whereas in the system shown at the bottom we have two different names Ahmed Jahanzeb and Muhammed Farrukh. Out of four names three names contain Muhammed but with different spellings. Computer cannot identify that the word 'Muhammed' is intended at all the three locations. So while consolidating data names are transformed to standard spellings of names. Similarly Date formats are different in both source systems and it is standardized in destination system (middle one).



DTS contains a set of tools that provides a very easy approach to build a package and execute it. Writing or building a package through programming is a complex task but DTS tools like DTS Designer and Import/Export Wizard do this entire complex task for user just through a single click of button. Not only package building but query building has also very sophisticated support in DTS tools.

Package execution utilities are used to run or execute a package, no matter package is designed through the tools provided by DTS or any external tool like Visual Basic. All these tools can be accessed through the SQL Server Enterprise Manager. Open the node Data Transfer Services in SQL Server Enterprise Manager. Choose the option in which any finished package is saved. Right click the package and get option to execute it.



The Data Transformation Services (DTS) node of the SQL Server Enterprise Manager Console tree provides facilities for accessing DTS tools, manipulating DTS packages, and accessing package information. You can use these facilities to:

• Open a new package in the DTS Import/Export Wizard or DTS Designer. In DTS Designer, you can select and edit an existing package saved to SQL Server, SQL Server 2000 Meta Data Services, or to a structured storage file.

Action >> New Package

- Connect to and import Meta data from a data source, and display the Meta data in the Meta Data node of SQL Server Enterprise Manager.

Right click Meta Data Services Package and select option import Meta Data

- Open a package template in DTS Designer.

Right click the package that is required to be opened in DTS Designer and select the option open in DTS Designer.

- Display the version history of a package, edit a specific package version in DTS Designer, and delete package versions.

Right click the package and select the option Versions.

- Display and manipulate package log information.

Right click the node "Local packages"/"Meta data services packages" in the tree and select the option view logs.

- Set the properties of DTS Designer

Right-click the Data Transformation Services node and click Properties.

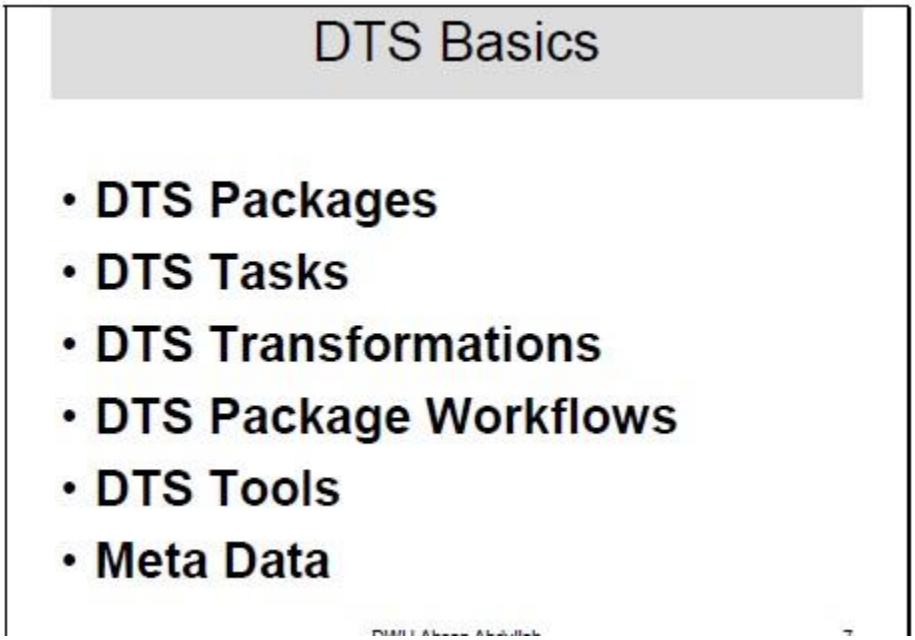
- Execute a package.

Right click the package and click execute

- Schedule a package.

Right click the package and click schedule

SQL Server is becoming one of Microsoft's biggest businesses, as being used by people from wide spectrum of domains. The scope of the SQL Server is so diverse that whole course can be offered and actually such courses are being offered in developed countries.
Microsoft also offers training courses and certifications are expensive enough. For example, "Designing and Implementing OLAP Solutions with MS SQL Server 2000" is a course that provides students with the knowledge and skills necessary to design, implement, and deploy OLAP solutions by using Microsoft SQL Server 2000™ Analysis Services. The importance of the course is well depicted by its cost i.e. $2500+GST.



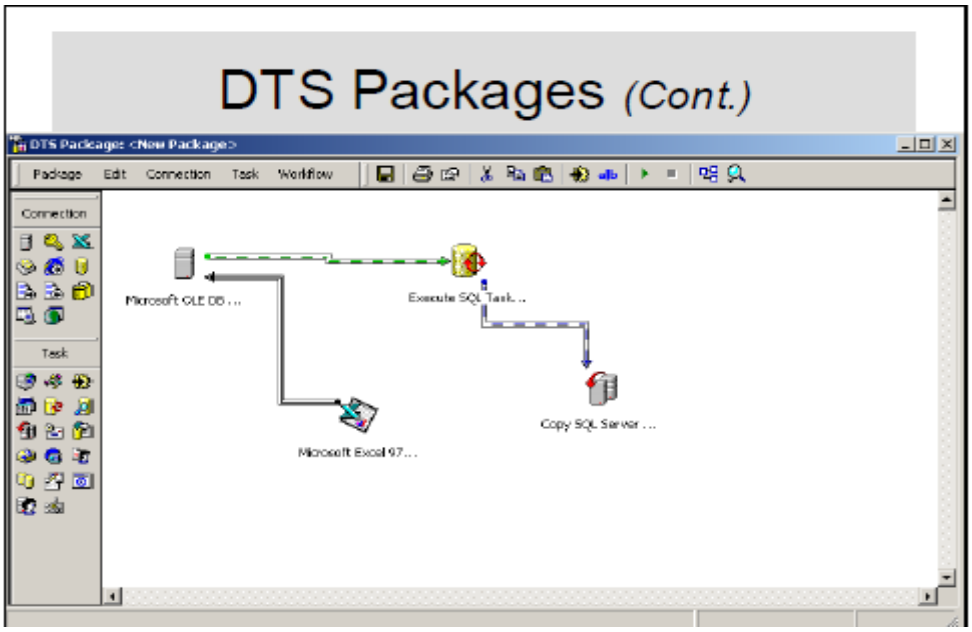
Before learning to use DTS some basic concepts like DTS packages, DTS tasks, transformations and workflows are important to understand. When we want to use computers to perform some particular task through programming, what we do? We write a program in some programming language. Program is a sequence of logical statements that collectively achieve the purpose of the programmer. This analogy is useful in understanding the concept of package and tasks in DTS. DTS package is exactly like a computer program. Like a computer program DTS package is also prepared to achieve some goal. Computer program contains set of instructions whereas DTS package contains set of tasks. Tasks are logically related to each other.

When a computer program is run, some instructions are executed in sequence and some in parallel. Likewise when a DTS package is run some tasks are performed in sequence and some in parallel. The intended goal of a computer program is achieved when all instructions are successfully executed. Similarly the intended goal of a package is achieved when all tasks are successfully accomplished.
DTS task is a unit of work in a package. Tasks can be establishment of connection to source and destination databases, extraction of data from the source, transformation of data, loading of data to the destination, generation of error messages and emails etc.

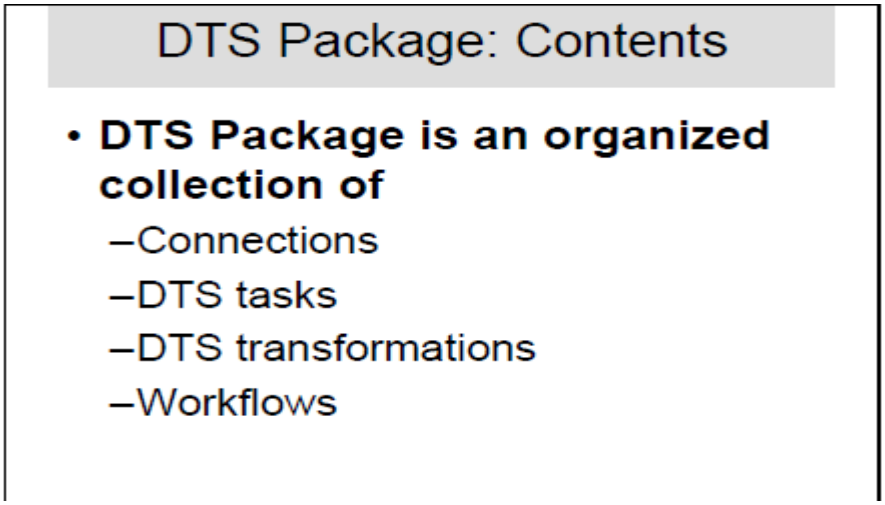
In real world systems when we talk about heterogeneous sources of data there arise a lot of complicated issues. Heterogeneous systems contain data with different storage conventions, different storage formats, different technologies, and different designs etc.

Power of DTS lies in extracting the data from these heterogeneous sources, transforming to some standard format and convention, and finally load data to some different system with totally different parameters like technology, design etc. Microsoft SQL Server provides user-friendly tools to develop DTS Packages. Through graphical editor/ designer or wizards we can put together set of tasks in a package. Order or sequence in which the tasks are required to be performed can be set through conditions like “On

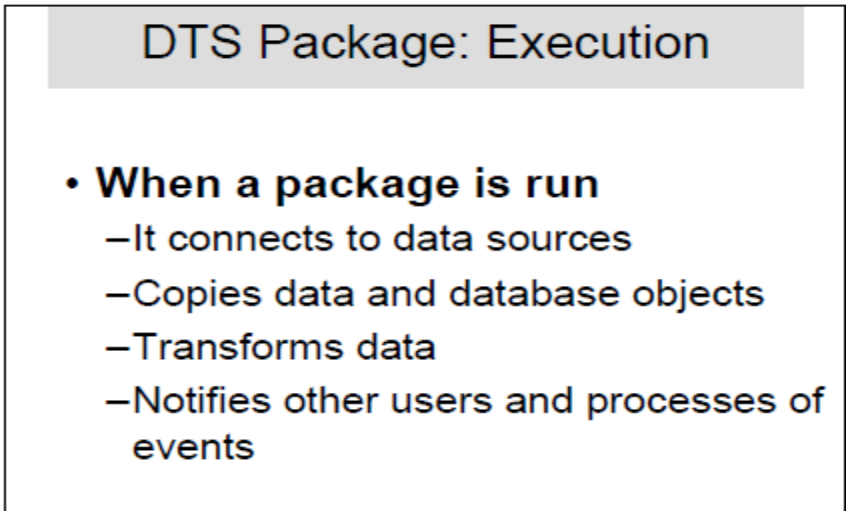
success of task. A task B should be performed otherwise task C should be performed." This order or sequence of execution is called Workflow of a package.
In this lecture we will see these concepts in detail and in subsequent lectures we will develop packages and practically get into the use of DTS functionalities.



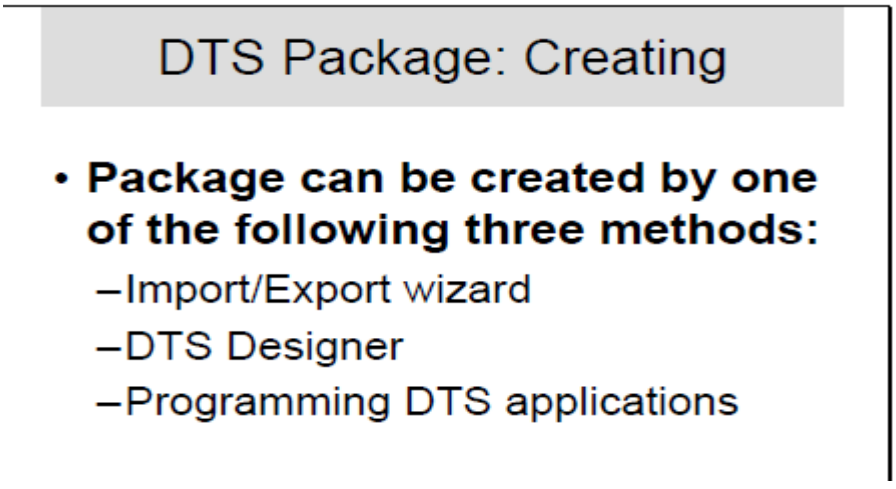
Slide shows how a package looks like. We can only view package as a form of graphical objects as shown in the slide. Here two connections are established. "Microsoft OLEDB Driver" and "Microsoft Excel 97" are connections. Black link between two connections is transformation task. "Execute SQL" and "Copy SQL Server" both are tasks. Green and blue links are workflows. Green link shows 'On the Success of' i.e. on the success of Connection establishment execute task execute SQL. Blue link shows 'On the Failure of' on the failure of the previous task execute another task Copy SQL Server objects.



A DTS package is an organized collection of connections, DTS tasks, DTS transformations, and workflow constraints assembled either with a DTS tool or programmatically and saved to Microsoft® SQL Server™, SQL Server 2000 Meta Data Services, a structured storage file, or a Microsoft Visual Basic® file.
Each package contains one or more steps that are executed sequentially or in parallel when the package is run. When executed, the package connects to the correct data sources, copies data and database objects, transforms data, and notifies other users or processes of events.



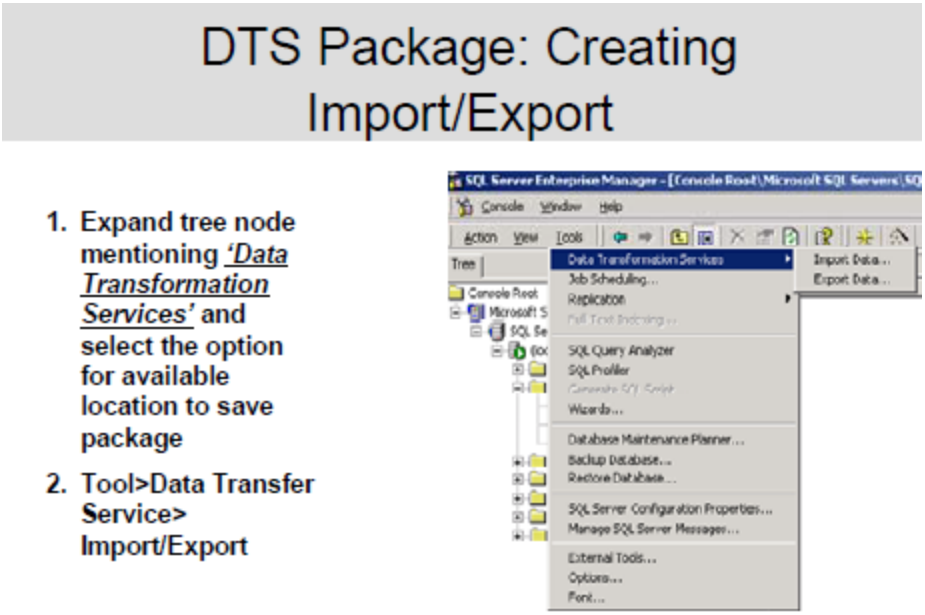
When we run a Data Transformation Services (DTS) package, all of its connections, tasks, transformations, and scripting code are executed in the sequence described by the package workflow.
We can execute a package from:
· Within a DTS tool.
· SQL Server Enterprise Manager.
· Package execution utilities.



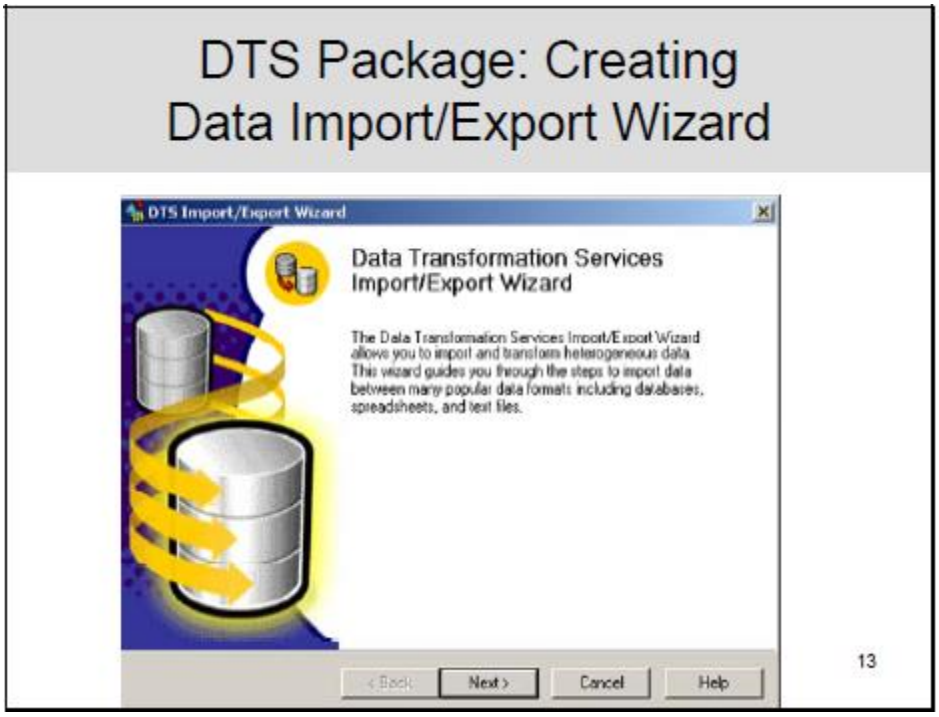
Microsoft SQL Server provides a good support for the tools that are helpful in building a package. Import/Export Wizard and DTS Designer both are the graphical methods of building a package. Both tools provide support to run the package also. Building a package means putting all the tasks that are supposed

to be performed in a particular package together and setting their order of execution or defining workflow. Whereas when we actually run a package all the tasks are actually performed.

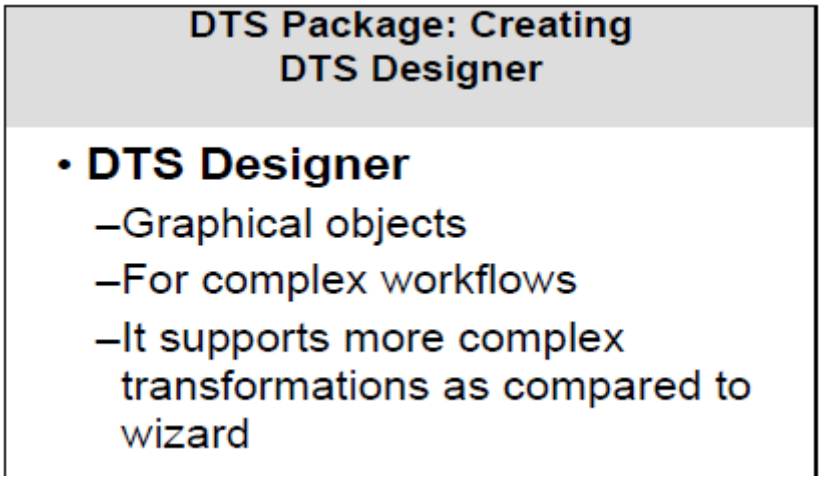
Programming DTS applications without the help of these user-friendly tools is a difficult task. Packages can be programmed using some external tool like Visual Studio in VC++ or VB. Such a programming requires deep understanding of DTS object model.



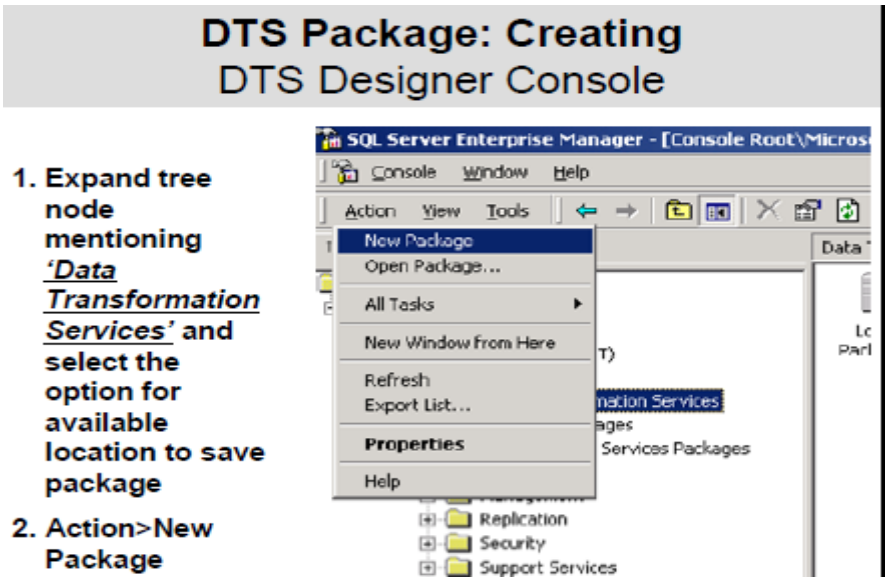
Data Import and Export wizard can be accessed through a number of ways.
1. Start > Programs> Microsoft SQL Server> Data Import/Export Wizard
2. through SQL Server Enterprise Manager
a. In SQL Server Enterprise Manager we can see a Tree view of SQL Server objects and services. Expand Tree, select the node "Data Transformation
Services". We can see two options (discussed earlier) to store Package.
Select any one of them (local OR SQL Server Meta Data Services). Then
Click Tools > Data Transformation Services > Import / Export Data.
b. After Expanding Data Transformation Services node we can click on tool bar to launch the wizard



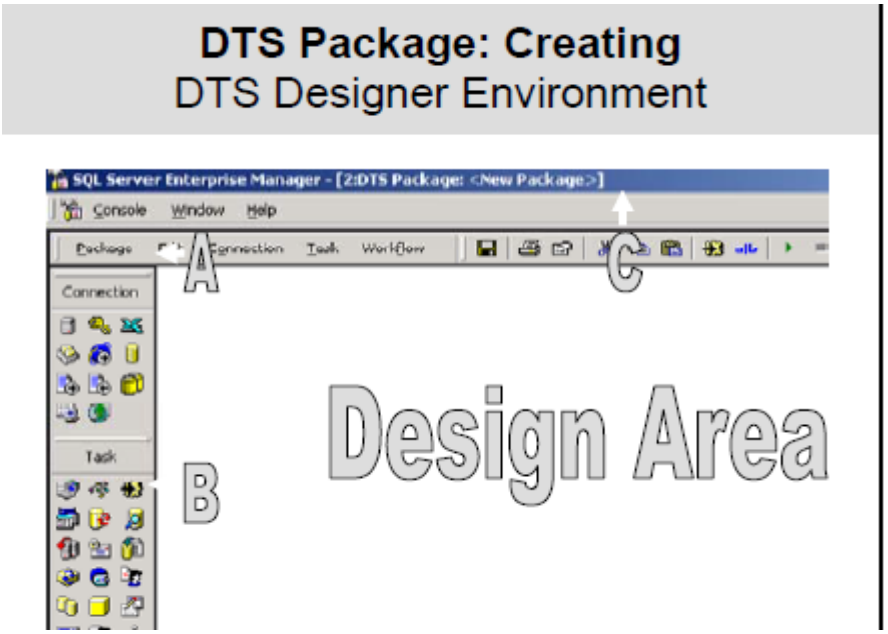
This is how wizard looks like. Just press Next and start working with a user friendly wizard. An easy-to-use tool that guides you, a step at a time, through the process of creating a DTS package. It is recommended for simple data transformation or data movement solutions (for example, importing tabular data into a SQL Server 2000 database). It provides limited support for transformations.



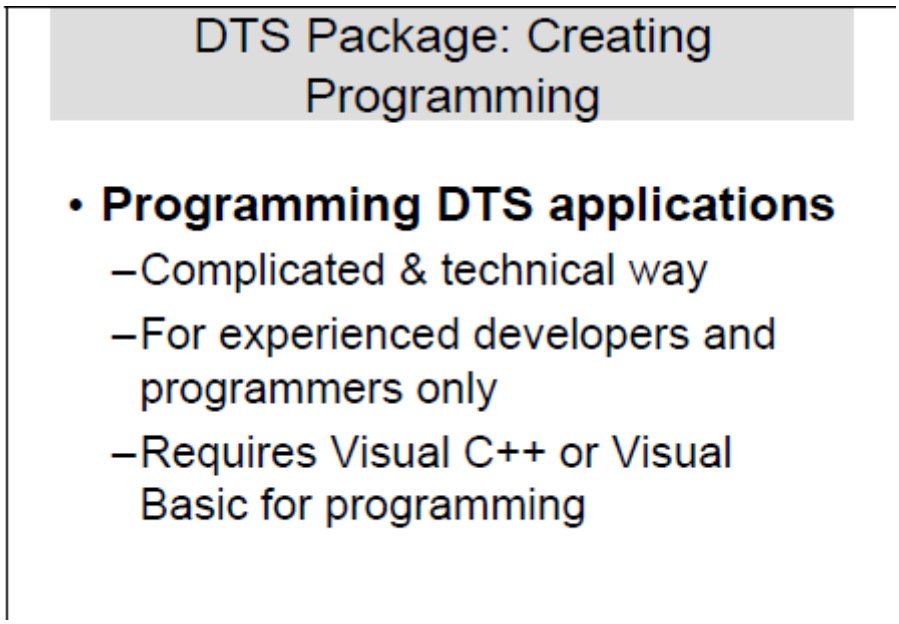
DTS designer is an application that uses graphical objects to help you build packages containing complex workflows. DTS Designer includes a set of model DTS Package Templates, each designed for a specific solution that you can copy and customize for your own installation. It is recommended for sophisticated data transformation solutions requiring multiple connections, complex workflows, and event-driven logic. DTS package templates are geared toward new users who are learning about DTS Designer or more experienced users who want assistance setting up specific DTS functionalities (for example, data driven queries).



DTS Designer can also be accessed through multiple ways.
1. Whenever a saved package is opened by double click or through right click, it is opened in DTS Designer
2. SQL Server Enterprise Manager can also be used to access DTS Designer
a. After Expanding Data Transformation Services node select Action > New Package
b. After Expanding Data Transformation Services node select on toolbar to access DTS Designer

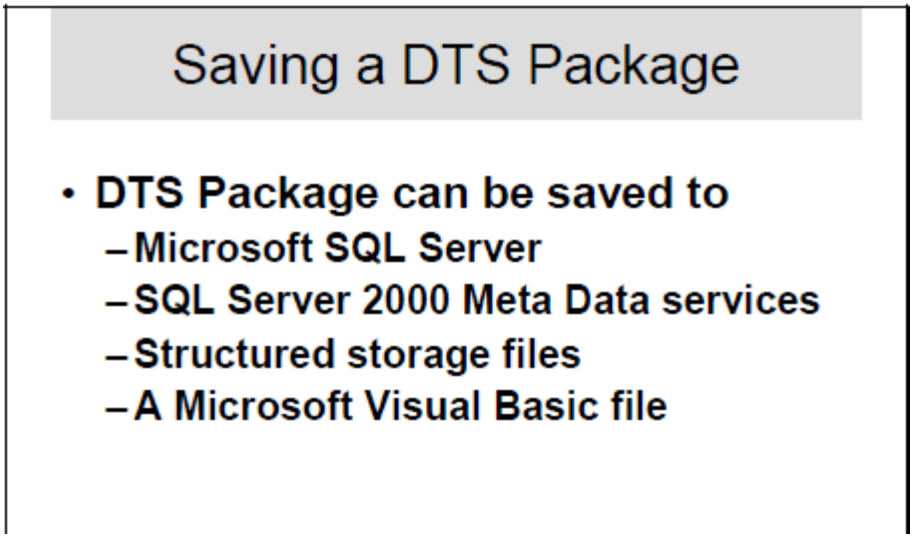


The slide shows environment of DTS Designer. In designer we can see four windows
A. Connection toolbar
B. Task toolbar
C. General toolbar
D. Design Area
A. Connection toolbar
Connection toolbar shows all available connections in the form of icons or symbols.
All OLE DB supported connections are available. To establish a new connection just click the correct icon and drag to design area. Then set properties to your connection.
In case of any difficulty in identifying the connection icon, click on Connection on
Menu bar just above the connection toolbar.
B. Task Toolbar
Tasks toolbar shows icons for all tasks that are supported by DTS. For example is used to set transformation task. This also works as drag and drop. DTS Designer is very friendly to use as it guides user about what to do after picking a certain option.
For new users who do not recognize the tasks through icons, in the top menu bar
'Task' is available.
C. General Toolbar
This toolbar provides general functionality like saving a package, executing a package is used to execute a package
D. Design Area
Design Area is used to design a package through the objects available in the tool bars.

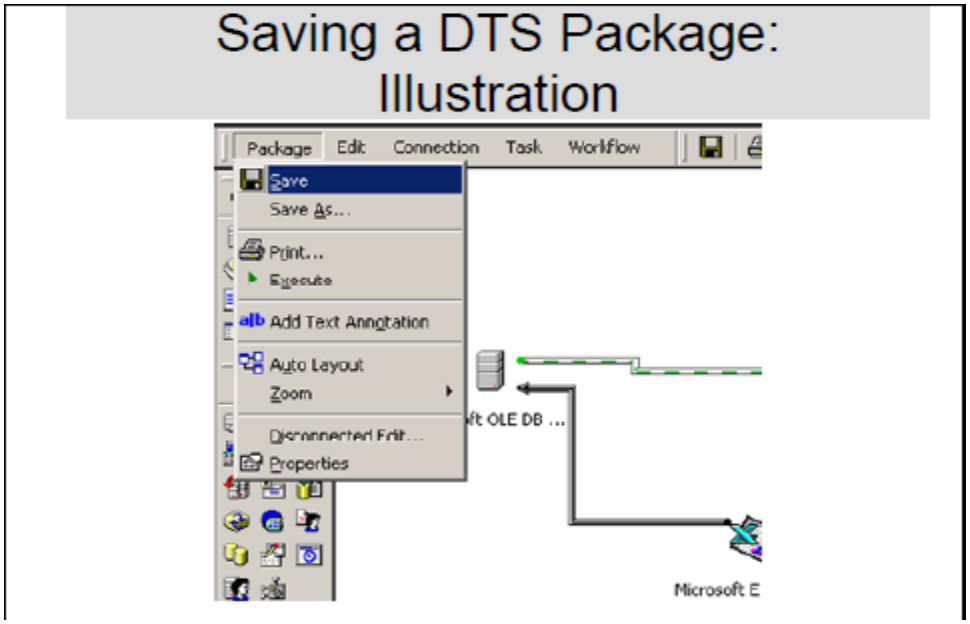


Programming applications that you can use to write and compile a DTS package either in Microsoft Visual Basic® or Microsoft Visual C++®. It is recommended for developers who want to access the DTS object model directly and exert a fine degree of control over package operations. Packages created programmatically can be opened and further customized in DTS Designer. In addition, packages created in the DTS Import/Export

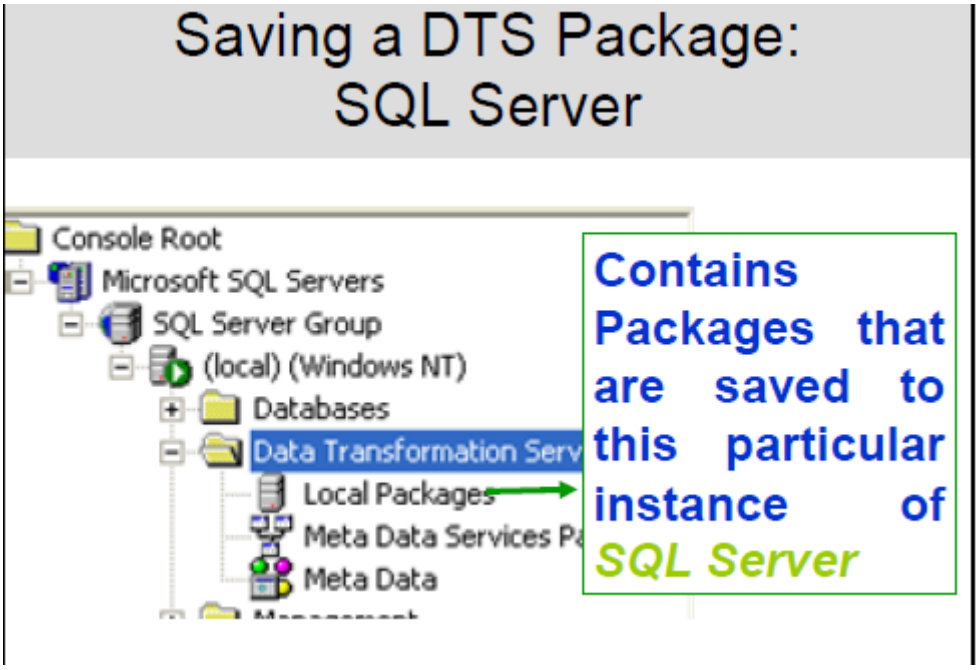
Wizard or DTS Designer can be saved as a Visual Basic program and then opened and further customized in a development environment such as Microsoft Visual Studio®.



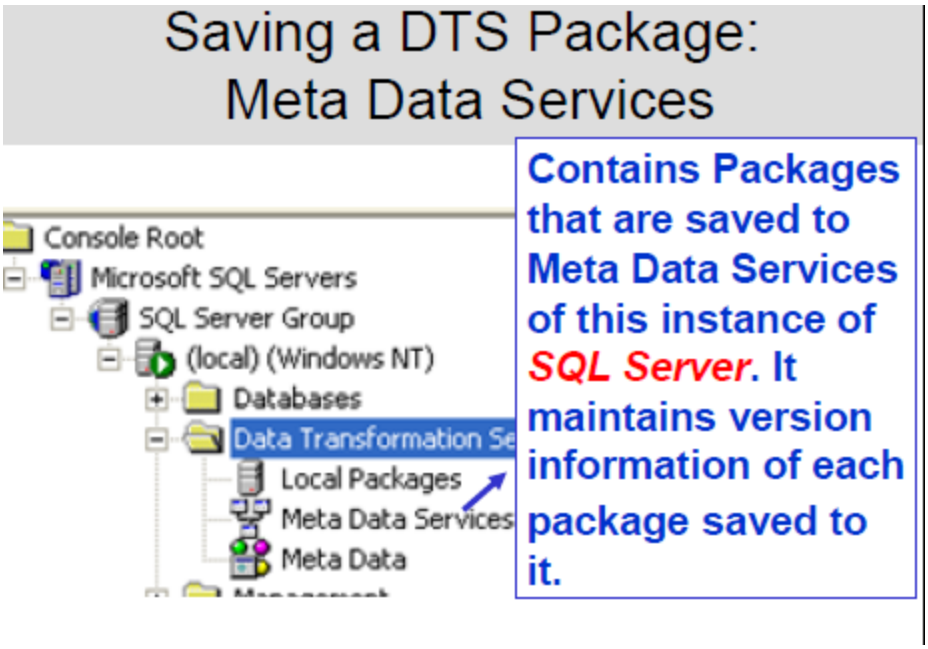
When you save a Data Transformation Services (DTS) package, you save all DTS connections, DTS tasks, DTS transformations, and workflow steps and preserve the graphical layout of these objects on the DTS Designer design While saving a package we get different options as destination location for the package. Package can be saved to Microsoft SQL server. Another option to save a package is SQL Server Meta Data Services. The advantage which we get when we store our package to
SQL Server 2000 Meta Data Services is that we may maintain Meta data information of the databases involved in the packages and we may keep version information of each package. Furthermore package can be stored in a structured file and Microsoft visual basic file.



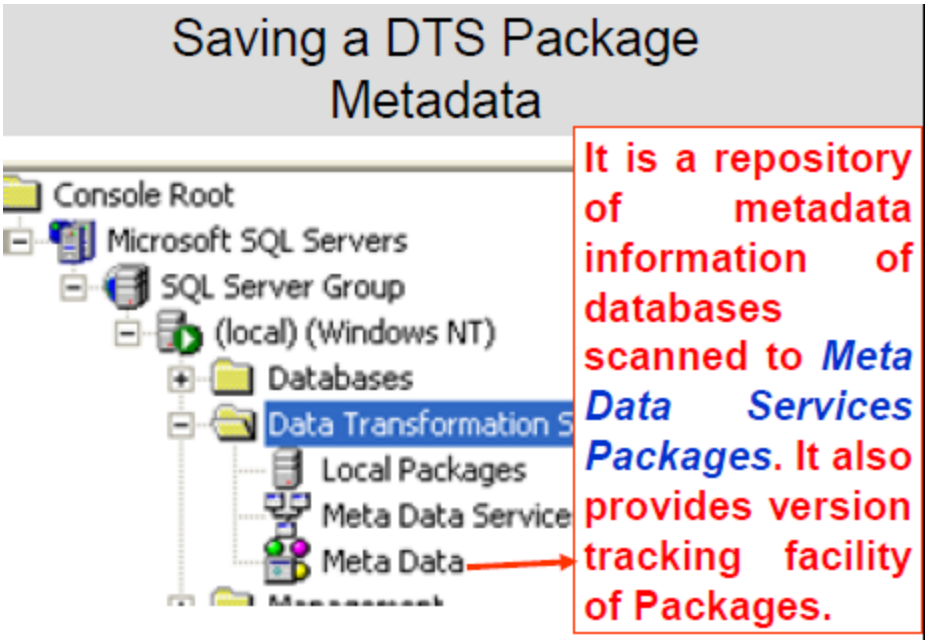
This slide illustrate the package saving process.



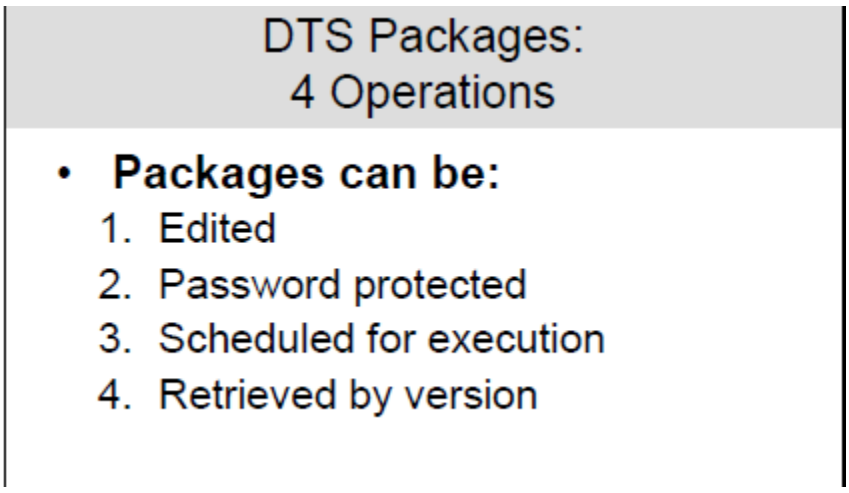
Data Transformation Services node of SQL Server Enterprise Manager contains three options to locate the package saved earlier. The first option is local Packages. These are the packages that are saved to this particular instance of SQL Server. Microsoft SQL
Server may have multiple instances on each machine or over a Network. Local packages are those that are saved to this particular instance of SQL Server.



As it has been discussed earlier, to maintain Metadata information for a package or version information of a package, it may be stored to Meta Data Services. This node contains all those packages that are saved to SQL Server 2000 Meta Data Services.

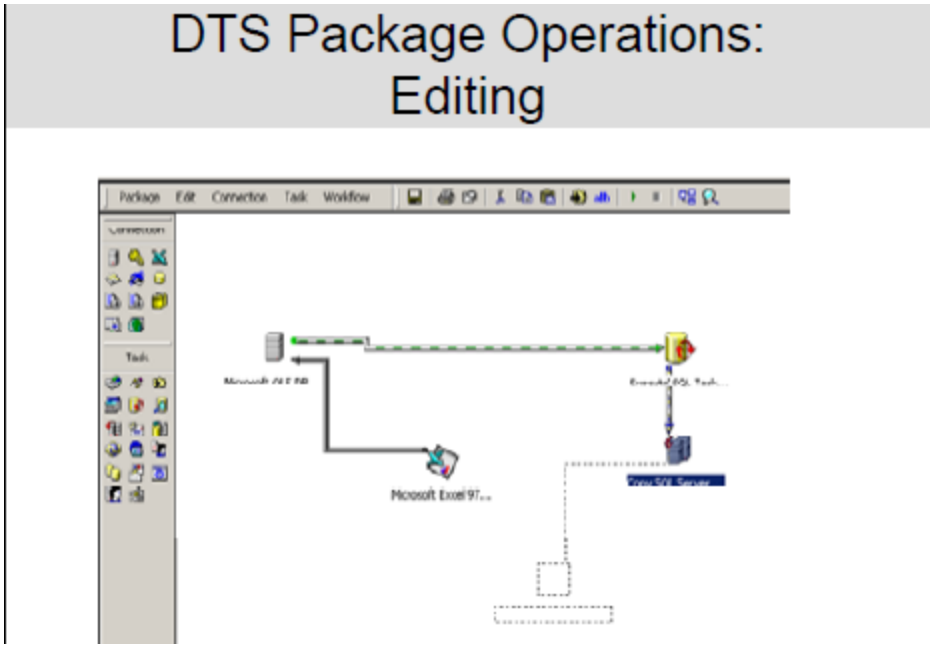


If a package is saved to SQL Server 2000 Metadata Services and is scanned for metadata than its meta data information is maintained in a repository " Meta Data ", that can be found as third option under node Data Transformation Services.

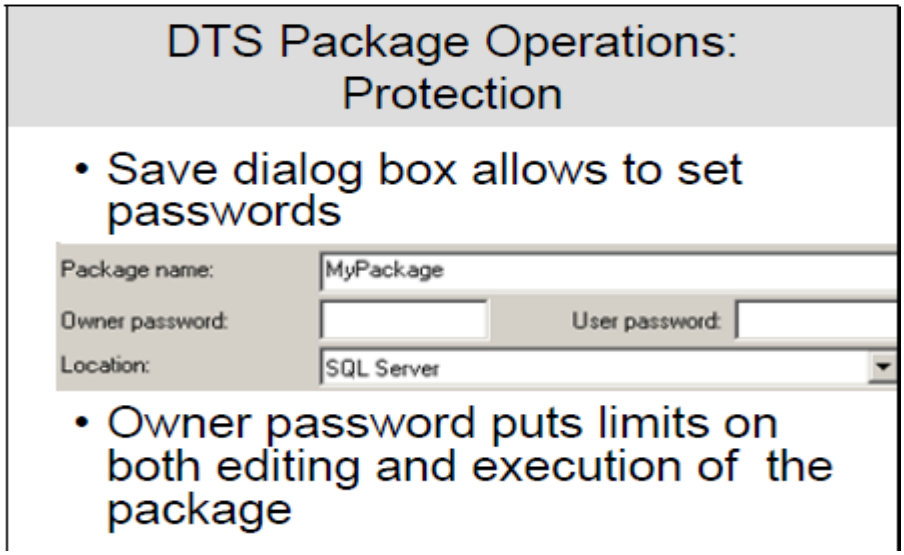


Packages that are required for very complicated tasks are not trivial to build. To develop such packages, DTS Designer or programming tools are used. Once such packages are built they are saved for further use. We may edit these packages later on. For editing purposes either DTS designer or programming is used. After edit a package we may keep both packages that is package before editing and package after editing as two different versions of same package. To maintain version information packages are saved in "SQL

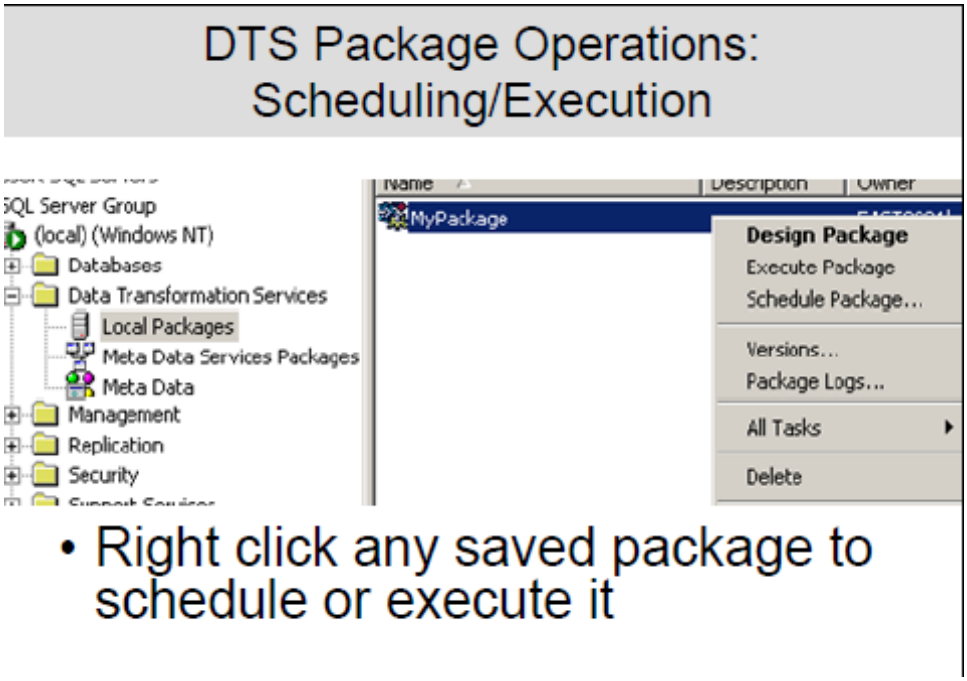
Server 2000 Meta Data Services". Tasks in the package require access to database objects, when package is executed. Packages can be protected through passwords. When a package is built it is not necessary to execute it immediately. We may schedule package to be executed any time later on.
We may prepare a package that is executed after definite intervals. For example we want to update our dataware house every night at 12:00 o' clock, what will we do? We will write a package to update dataware house and schedule it to run at 12:00 o' clock every night.



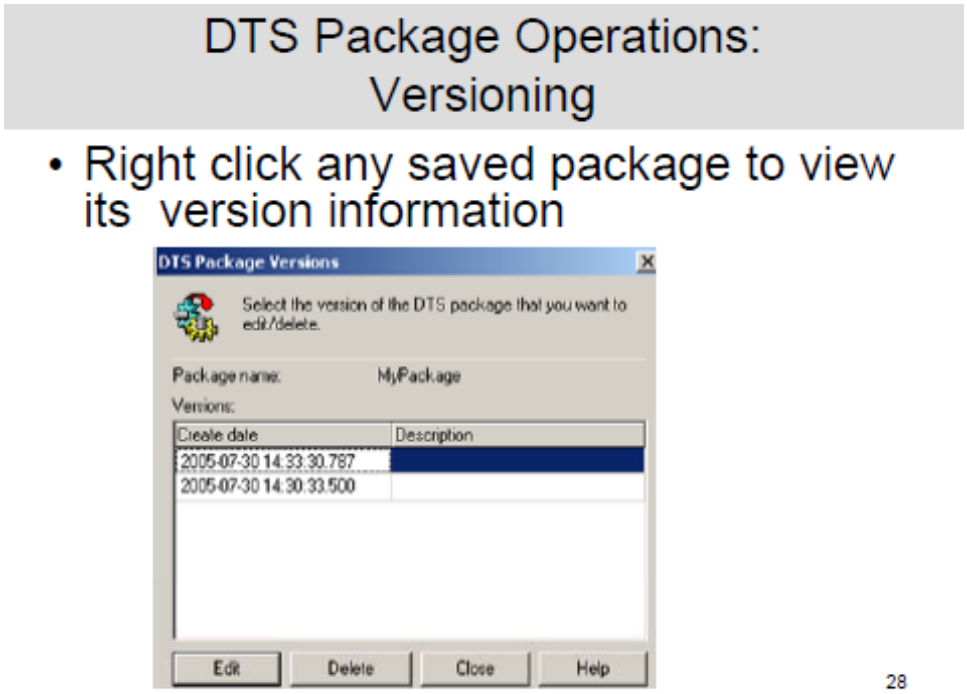
Double click a package to open in designer. Drag and drop objects to edit a package. Designer is the easiest way of editing a package. Even the packages that are created through wizards and are saved, can be edited through designer.



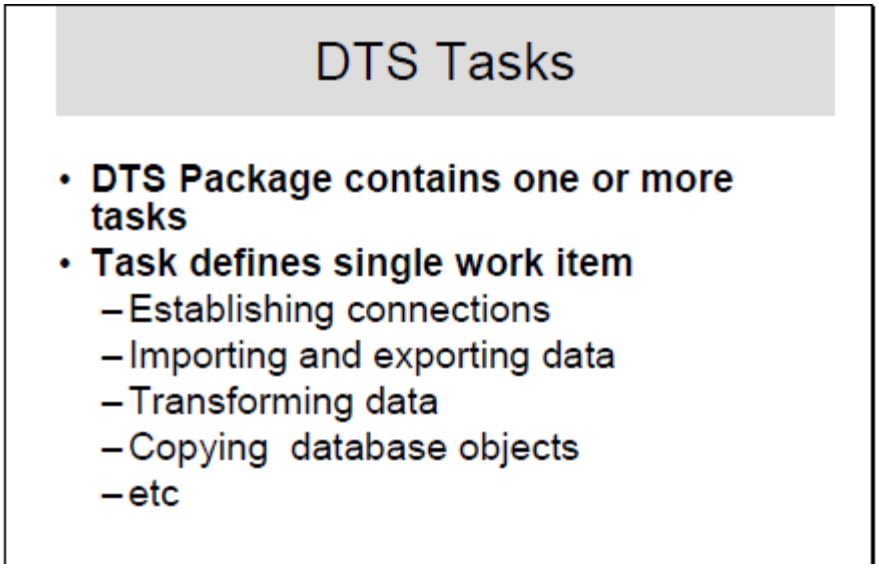
Enter an Owner password. Assigning an Owner password puts limits on who can both edit and run the package. Enter a User password. Assigning a User password puts limits only on who can edit the package. If you create a User password, you must also create an Owner password.



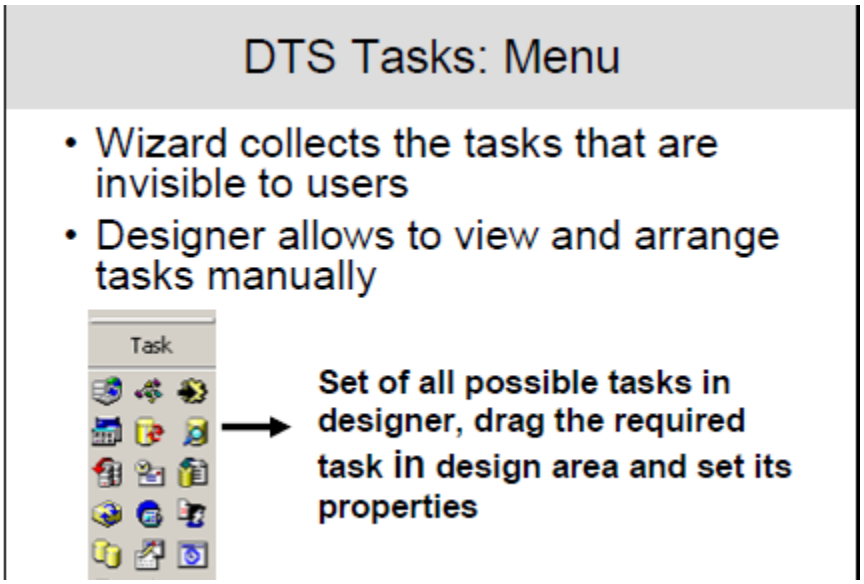
To schedule a package or to execute a package, first right click the package and then select the required option.



If we want to get version information of a package we can see it by right clicking the package and selecting version information. First column contains creation date and the other column contains the description about changes if it is saved with the package.



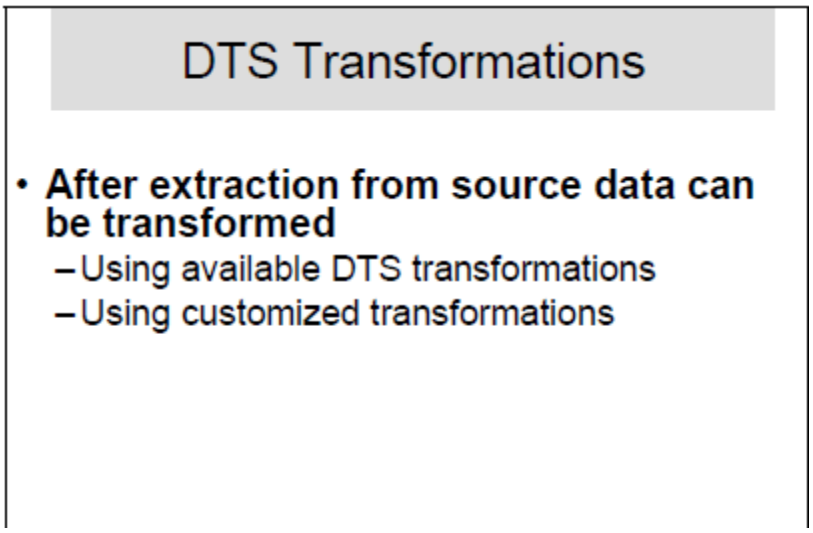
DTS Packages contain a sequence of tasks. When a package is executed these tasks are performed in sequence or in parallel. These tasks are the single work item in a package.
Tasks can be establishing connections, extraction of data from sources, transformations applied on data, loading data to destination, generation of automated email messages to administrator in case of some problem during the package execution.



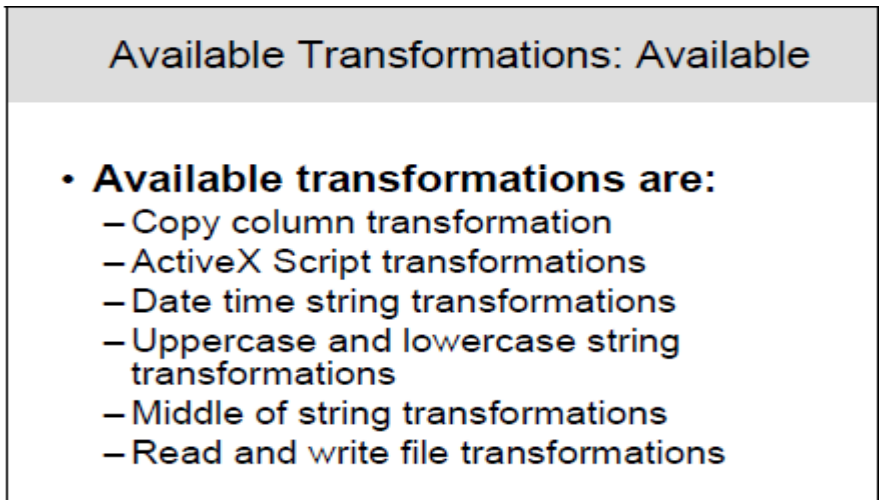
A DTS task is a discrete set of functionality, executed as a single step in a package. Each task defines a work item to be performed as part of the data movement and data transformation process, or as a job to

be executed. DTS supplies a number of tasks that are part of the DTS object model and can be accessed graphically, through DTS Designer, or programmatically. These tasks, which can be configured individually, cover a wide variety of data copying, data transformation, and notification situations.
For example: Importing and exporting data. DTS can import data from a text file or an OLE DB data source (for example, a Microsoft Access 2000 database) into SQL Server. Alternatively, data can be exported from SQL Server to an OLE DB data destination (for example, a Microsoft Excel 2000 spreadsheet). DTS also allows high-speed data loading from text files into SQL Server tables.
Transforming data. DTS Designer includes a Transform Data task that allows you to select data from a data source connection, map the columns of data to a set of transformations, and send the transformed data to a destination connection. DTS Designer also includes a Data Driven Query task that allows you to map data to parameterized queries.
Copying database objects. With DTS, you can transfer indexes, views, logins, stored procedures, triggers, rules, defaults, constraints, and user-defined data types in addition to the data.
In addition, you can generate the scripts to copy the database objects. Sending and receiving messages to and from other users and packages. DTS includes a Send Mail task that allows you to send an e-mail if a package step succeeds or fails. DTS also includes an Execute Package task that allows one package to run another as a package step, and a Message Queue task that allows you to use Message Queuing to send and receive messages between packages.

Executing a set of Transact-SQL statements or Microsoft ActiveX® scripts against a data source. The Execute SQL and ActiveX Script tasks allow you to write your own SQL statements and scripting code and execute them as a step in a package workflow.



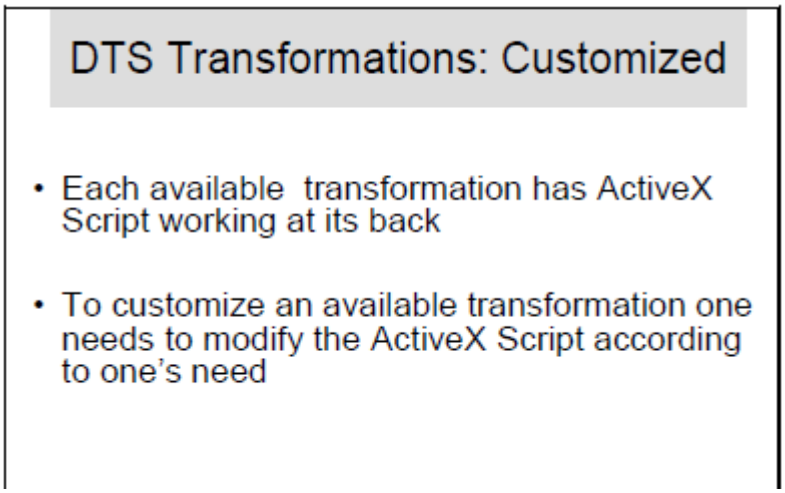
While transferring data from source to destination that may be a single source of truth, data may require to be transformed. Power of DTS tools lies in the support of data transformations. Some transformations are already available with DTS tools and customized transformations can be performed through VB Script or Java Script.



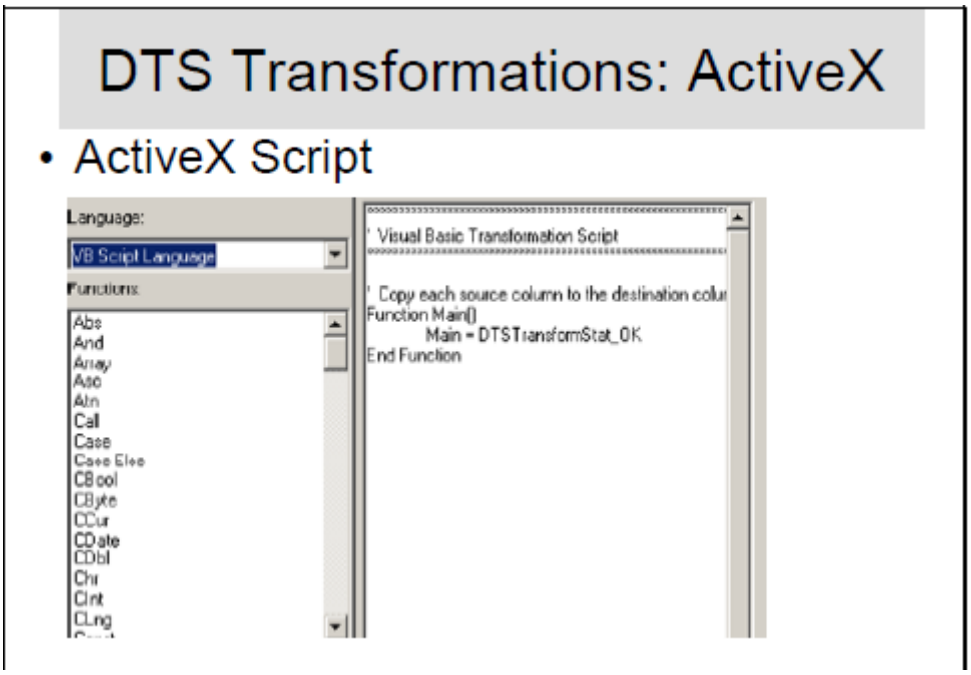
The slide shows the list of transformations that are already available with DTS tools i.e.
DTS Designer and DTS import/export wizard. Wizard has a support of two transformations out of six shown over here:

- Copy column transformation
- Active-X script transformation

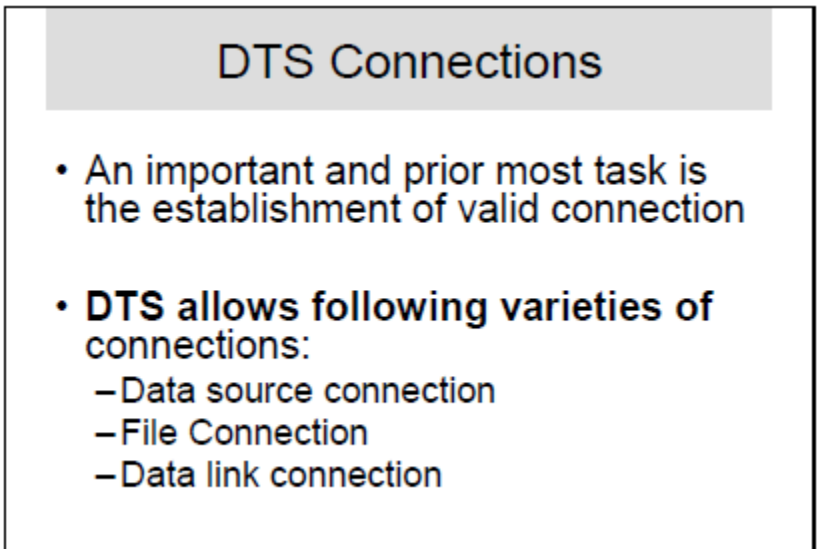
The rest four are accessed through DTS designer and scripts.
Copy Column Transformation: Describes the transformation used to copy source data to the destination.
ActiveX Script Transformation: Explains how to use Microsoft ActiveX® scripts to define column-level transformations.
Date Time String Transformation: Describes the transformation used to convert a source date into a new destination format.
Uppercase String Transformation: Describes the transformation used to convert a string into uppercase characters.
Lowercase String Transformation: Describes the transformation used to convert a string into lowercase characters.
Middle of String Transformation: Describes the transformation used to extract a substring from a source and optionally change its case or trim white space before placing the result in the destination.
Trim String Transformation: Describes the transformation used to remove leading, trailing, or embedded white space from a source string and place the (optionally caseshifted) result in the destination.
Read File Transformation: Describes the transformation used to copy the contents of a file specified by a source column to a destination column.
Write File Transformation: Describes the transformation that creates a new data file for each file named in a source column and initializes the contents of each file from data in a second source column.



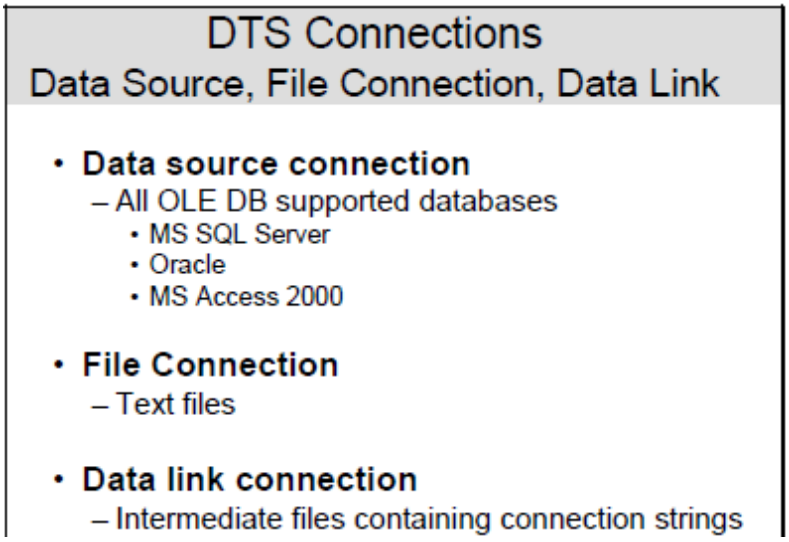
In SQL Server, transformations are applied through running ActiveX Scripts. When we apply an available transformation, tools in SQL Server generate ActiveX Script automatically. This auto generated script can be modified or customized according to our needs. Customized transformations are those in which we customize the auto generated ActiveX Scripts to fulfill our particular need.
For Example:
An available transformation can transform Saad Munir Rao to SAAD MUNIR RAO but if we want to transform it as S. M. Rao then we need to customize the transformation.



Designer provides such an interface to write/customize ActiveX Scripts. To access it we will see it later on.

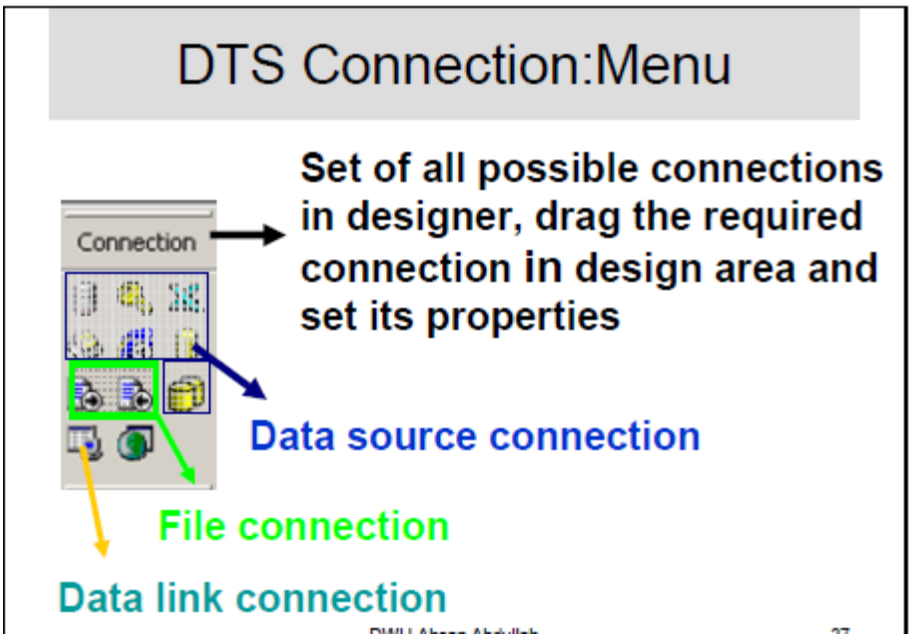


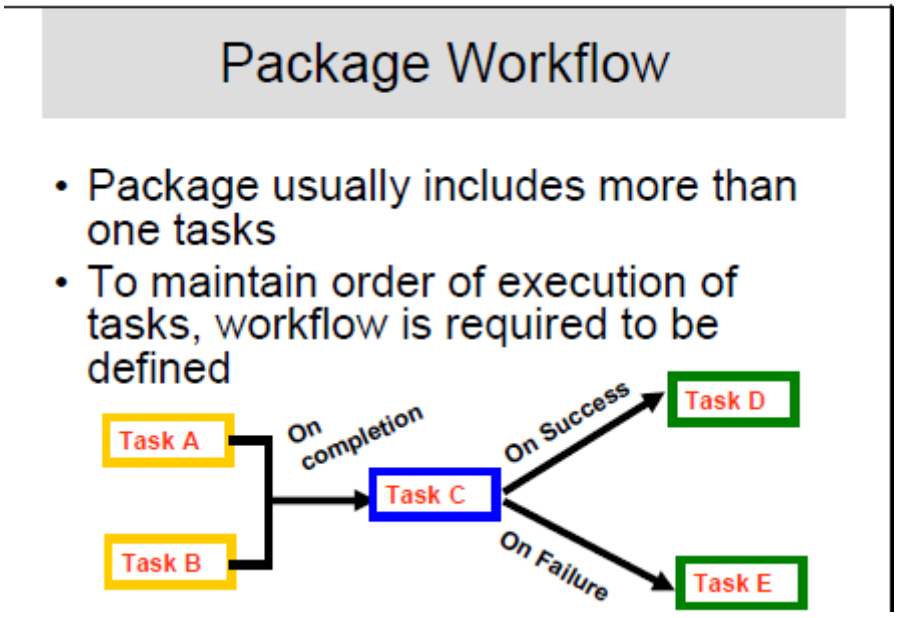
To successfully execute Data Transformation Services (DTS) tasks that copy and transform data, a DTS package must establish valid connections to its source and destination data and to any additional data sources (for example, lookup tables). Because of its OLE DB architecture, DTS allows connections to data stored in a wide variety of OLE DB-compliant formats. In addition, DTS packages usually can connect to data in custom or nonstandard formats if OLE DB providers are available for those data sources and if you use Microsoft® Data Link files to configure those connections.



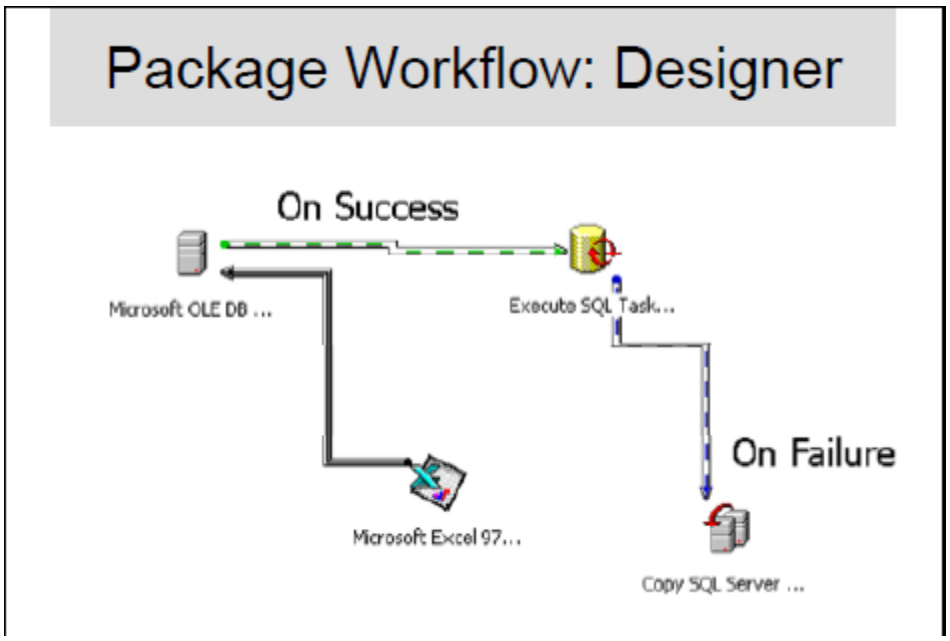
DTS allows the following varieties of connections:
A data source connection. These are connections to: standard databases such as
Microsoft SQL Server™ 2000, Microsoft Access 2000, Oracle, dBase, Paradox; OLE DB connections to ODBC data sources; Microsoft Excel 2000 spreadsheet data; HTML sources; and other OLE DB providers.
A file connection. DTS provides additional support for text files. When specifying a text file connection, you specify the format of the file. For example:
· Whether a text file is in delimited or fixed field format.

Whether the text file is in a Unicode or an ANSI format.
· The row delimiter and column delimiter if the text file is in fixed field format.
· The text qualifier.
Whether the first row contains column names.
A data link connection. These are connections in which an intermediate file outside of SQL Server stores the connection string.





Precedence constraints sequentially link tasks in a package. In DTS, you can use three types of precedence constraints, which can be accessed either through DTS Designer or programmatically:
Unconditional: If you want Task 2 to wait until Task 1 completes, regardless of the outcome, link Task 1 to Task 2 with an unconditional precedence constraint.
On Success: If you want Task 2 to wait until Task 1 has successfully completed, link
Task 1 to Task 2 with an On Success precedence constraint.
On Failure: If you want Task 2 to begin execution only if Task 1 fails to execute successfully, link Task 1 to Task 2 with an On Failure precedence constraint. If you want to run an alternative branch of the workflow when an error is encountered, use this constraint.



This slide shows the process of making workflows using the designer. It provides a graphical interface making the workflow management very easy.

Lab 15 Part II

Lab Data Set

In previous lecture I gave you an overview of the tool to be used for the lab i.e. Data Transformation Services (DTS), MS SQL Server. Now keeping in view the real issue of data acquisition, we will provide you with a simulated data set, so as to make you ready to start exploring the tool. The data is for a multi-campus university having campuses in four major cities. We discussed the details of such a university in Lect-6 of the course i.e. normalization. Each of the campus has its own conventions and norms regarding storing Student information.

Multi-Campus University

- **University has four campuses situated at:**
  i) Lahore
  ii) Karachi
  iii) Islamabad
  iv) Peshawar
- **University Head Office in Islamabad**

Data warehouse is a single source of truth. We have to put all data from different data sources (campuses) at one place in some standard form. The task is not trivial. Different sources of data have a lot of inherent issues of ETL. High level steps given in the slide give just an overview of the task. First of all, we have to identify the source systems. It is quite possible that each campus uses different database systems or same organization at different geographical locations uses different database management systems. To put data into a single source, after extracting from such diverse sources, requires powerful tools especially designed to fulfill the requirements of ETL. We will use Microsoft SQL Server DTS which is a user friendly graphical tool and makes such a complex task doable by some practice. After identification of source systems, it is necessary to study the issues that must be considered before putting all the data together at a single location. Microsoft SQL Server provides a powerful support to perform Extract, Transform and Load (ETL) data from source systems to destination system. Finally certain steps are performed to check and improve quality of data.

In this lab lecture we will look into the data for each of the campuses in detail. This would lead us to identify the core issues that are needed to be taken care of before extracting data from these diverse sources into a single destination.

Degree Programs

- At each campus university has two degree programs:
  (1) BS
  (2) MS
- University started its BS degree program in year 1994 and MS degree program in year 2001

Our Example University offers undergraduate and graduate degrees in all of its campuses. The undergraduate degrees were started in year 1994 and graduate degrees were started in year 2001.

Disciplines for BS

1) Four disciplines at BS level

- Computer Science (CS)
- Computer Engineering (CE)
- System Engineering (SE)
- Telecommunication (TC)

All campuses offer these four disciplines
The slide is self-explanatory.

Disciplines for MS

2) Four disciplines at MS level

- Computer Science (MS-CS)
- Software Project Mgmt. (MS-SPM)
- Networking (MS-NW)
- Telecommunication (MS-TC)

i) Lahore & Karachi campuses offer all the four disciplines

ii) Islamabad offers MS-CS & MS-SPM

iii) Peshawar offers MS-CS & MS-TC

The slide is self-explanatory.

The need

- Four campuses of the University maintain their students record locally
- No standardized way of record management
- Standardized reporting is difficult and time consuming.
- No centralized repository of data
- Head Office wants a central data repository for decision support i.e. a DWH
- We will study the record management at each campus

In this lecture, we will collect data from each campus and figure out the issues as mentioned earlier, our example university has multiple campuses and each campus independently maintains its student records without any meaningful level of coordination. There is no any standardized record management system or agreement among these campuses. Each of the campuses uses its own student record management practices independent of the other campuses. The head office of the university now wants to consolidate the student records from all of the four campuses into a central repository for decision support. Thus they are planning for a DWH.

Students Record Keeping & Mgmt.

- One by one we discuss the record management system specific to each campus of the University
  1) Lahore
  2) Karachi
  3) Islamabad

4) Peshawar

In real life when we need to work with heterogeneous systems from multiple sources then the problems like poor design becomes prominent and significant. In this student record management system none of the database is properly designed and in some cases, there is no database at all. The databases are not normalized. Each of the campus maintains two “tables” to store student information. I have used double quotes as the word table is not used in its literal meaning, especially in the case of a single flat text file.

Student Table:
In each database Student table is used to maintain personal records of the students. This table has only one entry for each student in each campus. A student may have entries in student tables of two campuses in the issues like transfer cases.

Registration Table:
Second table is registration table that maintains the record for course registration. This table contains as many records for each student as many times he/she registered any course. Each campus keeps two tables does not mean that each campus has two files only (one for each table). Each campus maintains its information independent of each other. Lahore campus maintains two text files for each batch i.e. entry taken in a year. For each batch one file contains student information and other file contains registration information. For eleven batches of BS Lahore campus has 22 text files. For four batches of MS Lahore campus contains eight text files. Same is the mechanism used in Peshawar campus to store the data in text files. Islamabad campus has MS Access database with three tables.
Two of these three tables contain student information. One table for MS and the other for BS students. The third table contains Registration data for both degree programs i.e. MS and BS. Karachi campus manages to store all this information in MS Excel sheets. Three Excel Books are maintained. Two out of three contains registration records (one for BS and the other for MS) and the third one contains student records for both degree programs.

Let us discuss “student record management systems” at each of the campuses.

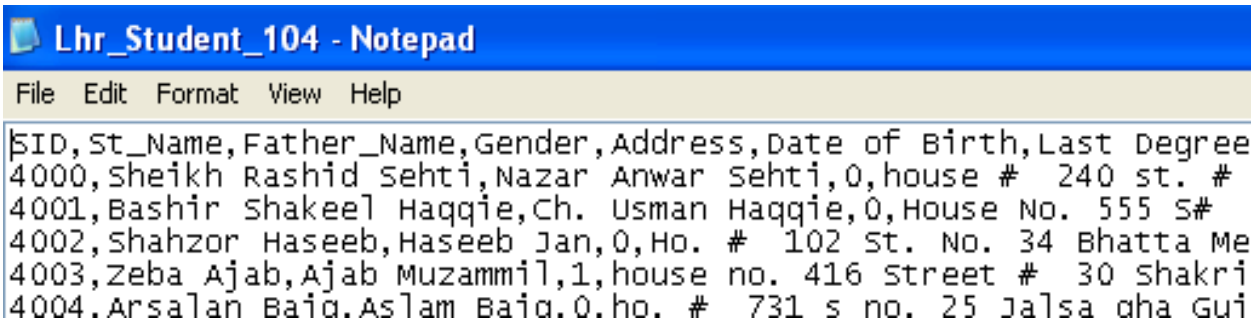
Data from Lahore Campus

- Data at Lahore campus is stored in Text files
- To store data regarding one complete batch 2 text files are used:
  (1) Lhr_Student_batch (Student record)
  (2) Lhr_Detail_batch (Course Reg. record)
- 22 text files for 11 BS batches
- 8 text files for 4 MS batches

The slide is self-explanatory. Here batch is the year the student entry was taken i.e. 94, 95, 104 i.e. year 2004.

Data from Lahore Campus: Sample

- Flat file student data at Lahore campus



The slide shows the screenshot of a sample text file for student records at Lahore campus. We can see that the first row contains the header and the columns are delimited by comma. Let's discuss header of both student and registration tables in detail.

Lahore: Header of Student Table

- *SID:* Student ID
  - A numerical value, starting from 0
  - Starts from 0 individually for both degrees BS & MS
  - It is unique within a degree (BS/MS) but not unique across the degrees
  - Combination of SID and degree is always unique within a campus
- *St_Name:* Student name
- *Father_Name:* Father name

The slide is self explanatory.

Lahore: Header of Student Table

- *Gender:*
  a. 0 for Male
  b. 1 for Female
- *Address:* Permanent Address
- *[Date of Birth]:*
  1. 14-Apr-1980

[Reg Date]: Date on which student was enrolled

This is the convention used for storing some critical data at the Lahore campus. There is no guarantee that the same convention will be used at other campuses too, actually in some cases the converse may be true. We will identify and work on these apparent anomalies in the data profiling phase before we do the actual transformation.

Lahore: Header of Student Table

- [Reg Status]:
  a. 'A' if student was enrolled as new Admission
  b. 'T' if student was enrolled as Transfer case
- [Degree Status]:
  (a) 'C' (complete) if student has graduate
  (b) 'I' for incomplete degree
- *[Last Degree]:*
  a. F.Sc. / A level for BS
  b. M.Sc. / BS / BE for MS

The slide is self-explanatory.

Lahore: Header of Course Reg. Table

- *SID:*
- *Degree:* BS/MS
- *Semester:* e.g. Fall04
- *Course:* Course code
- *Marks:* Out of 100
- *Discipline:* CS/TC/SE/CE

The slide shows the header and sample values for Course registration table at Lahore Campus.

Lahore: Facts about Data

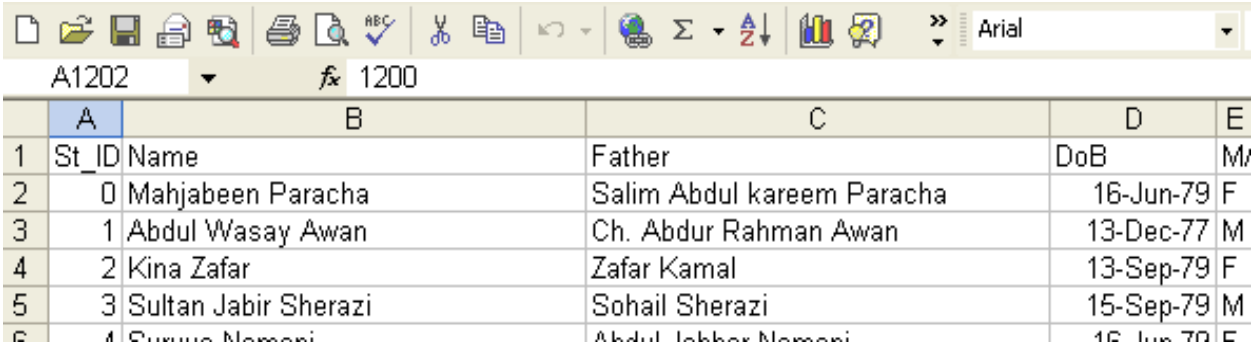
- *Total students = 5,200*
- *Total male students= 3,466*
- *Total BS students= 4,400*
- *Number of graduated students= 3,200*
- *Number of post graduated std.= 600*

The slide shows some of the facts about Lahore campus. These facts can be used for data validation in later steps. However, this has to be taken with a "pinch of salt" because the facts before resolving the data quality issues will most likely be different as compared to the ones after the data has been cleansed.

Data from Karachi Campus

- Data at Karachi campus is stored in MS-Excel books
- Three books are maintained
    - STUDENT_KHR (Student record)
    - Reg_BS_KHR (BS course Reg. record)
    - Reg_MS_KHR (MS course Reg. record)
- STUDENT_KHR keeps two sheets
    - 'BS' for BS student's records
    - 'MS' for MS student's records

The slide is self-explanatory.



	A	B	C	D	E
1	St_ID	Name	Father	DoB	M
2	0	Mahjabeen Paracha	Salim Abdul kareem Paracha	16-Jun-79	F
3	1	Abdul Wasay Awan	Ch. Abdur Rahman Awan	13-Dec-77	M
4	2	Kina Zafar	Zafar Kamal	13-Sep-79	F
5	3	Sultan Jabir Sherazi	Sohail Sherazi	15-Sep-79	M

The slide shows MS Excel screenshot of the sample data for Karachi campus. Let's discuss its header in detail for both student and registration tables.

Karachi: Header of Student Table

- *St_ID:* Student identity
- *Name:* Student name
- *Father:* Father name
- *DoB:* Date of Birth
- *M/F:* Gender (M/F)
- *DoReg:* Date of Registration/Enrollment
- *RStatus:* Status of enrollment (A/T)
- *DStatus:* Status of Degree (C/I)
- *Address:* Permanent address
- *Qualification:* Last degree achieved

The slide is self-explanatory.

Karachi: Header of Course Reg. Table

- *SID:*
- *Courses:* Course code
- *Score:* Out of 100
- *Sem:* e.g. Fall04
- *Disp:* CS/TC/SE/CE

The slide is self-explanatory.

Karachi: Facts About Data

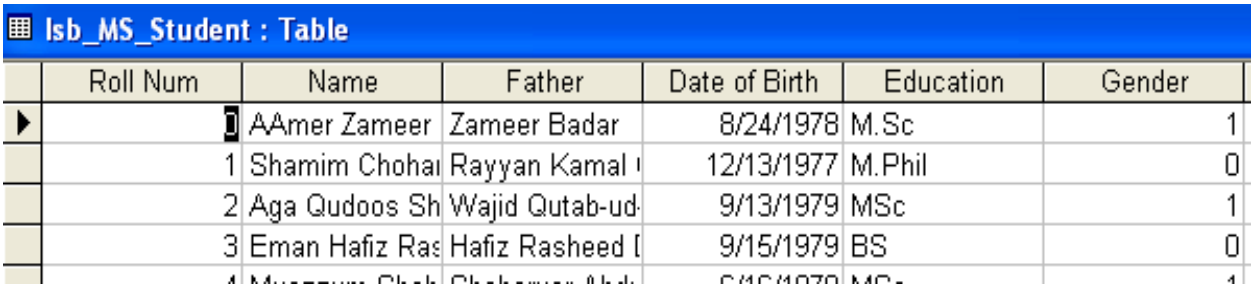
- *Total students = 6,000*
- *Total male students= 4,500*
- *Total BS students= 4,000*
- *Number of graduated students= 3,500*
- *Number of post graduated std.= 1,500*

The slide shows some of the facts about Karachi campus. These facts can be used for data validation in later steps. Again we have to look at the facts keeping in mind that the same may change after data has been cleansed.

Data from Islamabad Campus

- MS-Access is used at Islamabad campus
- Database has three tables
    1. Isb_BS_Student (MS Student record)
    2. Isb_MS_Student (BS Student record)
    3. Registration (All reg. record BS + MS)
- Roll number is also used as primary key in student table

The slide is self-explanatory.



isb_MS_Student : Table

Roll Num	Name	Father	Date of Birth	Education	Gender
0	AAmer Zameer	Zameer Badar	8/24/1978	M.Sc	1
1	Shamim Chohai	Rayyan Kamal	12/13/1977	M.Phil	0
2	Aga Qudoos Sh	Wajid Qutab-ud	9/13/1979	MSc	1
3	Eman Hafiz Ras	Hafiz Rasheed [	9/15/1979	BS	0

The slide shows MS Access screenshot of the sample data for Islamabad campus. Let's discuss its header in detail for both student and registration tables.

Islamabad: Header of Student Table

- *Roll Num:* Student identity
- *Name:* Student name
- *Father:* Father Name
- *Reg Date:* Date of Enrollment
- *Reg Status:* Status of Enrollment (A/T)
- *Degree Status:* Status of Degree (C/I)
- *Date of Birth:* Date of Birth
- *Education:* Last degree achieved
- *Gender:* Gender (Male=1, Female =0)
- *Address:* Permanent address

The slide is self-explanatory.

Islamabad: Header of Course Reg. Table

- *Roll Num:*
- *Course:* Course code
- *Marks:* Out of 100
- *Discipline:* CS/TC/SE/CE
- *Session:* e.g. Fall04

Here we can see that Degree (BS/MS) is missing, whereas same table contains records for both. Only way to differentiate is through discipline attribute.

Islamabad: Facts about Data

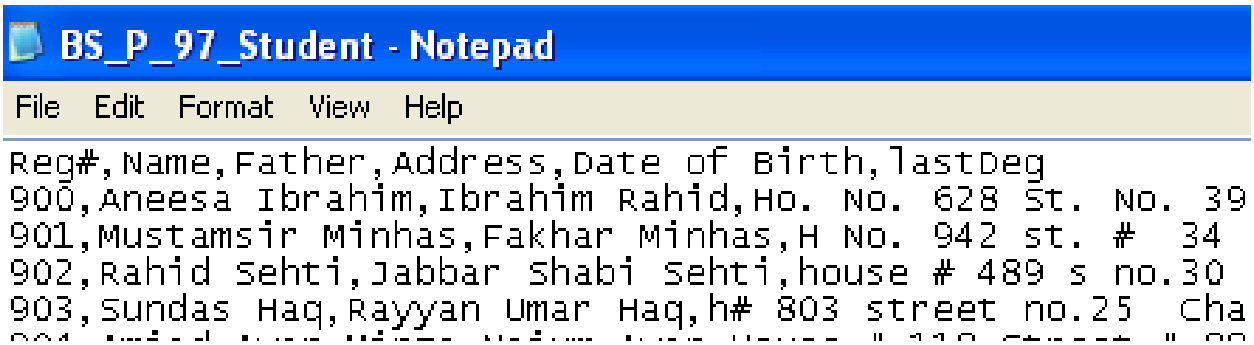
- *Total students = 4,400*
- *Total male students= 3,700*
- *Total BS students= 3,200*
- *Number of graduated students= 2,500*
- *Number of post graduated std.= 900*

The slide shows some of the facts about Islamabad campus. These facts can be used for data validation in later steps.

Data from Peshawar Campus

- Data at Peshawar campus is stored in Text files
- To store data regarding one complete batch 2 text files are used
  (1) Lhr_Student_batch (Student record)
  (2) Lhr_Detail_batch (Course Reg. record)
- 22 text files for 11 BS batches
- 8 text files for 4 MS batches

The slide is self-explanatory.



The slide shows the screenshot of a sample test file for student records at Peshawar campus. We can see that the first row contains the header and the columns are delimited by comma. Let's discuss header of both student and registration tables in detail.

Peshawar: Header of Student Table

- *Reg#:* Student identity
- *Name:* Student name
- *Father:* Father name
- *Address:* Permanent address
- *Date of Birth:* Date of Birth
- *lastDeg:* Last degree achieved
- *Reg Date:* Date of Enrollment
- *Reg Status:* Status of Enrollment (A/T)
- *Degree Status:* Status of Degree (C/I)

The slide is self-explanatory.

Peshawar: Header of Course Reg. Table

- *Reg#:*
- *Courses:* Course code
- *Score:* Out of 100
- *Program:* CS/TC/SE/CE
- *Sem:* Fall/Spring
- *Year:* YYYY e.g. 1999

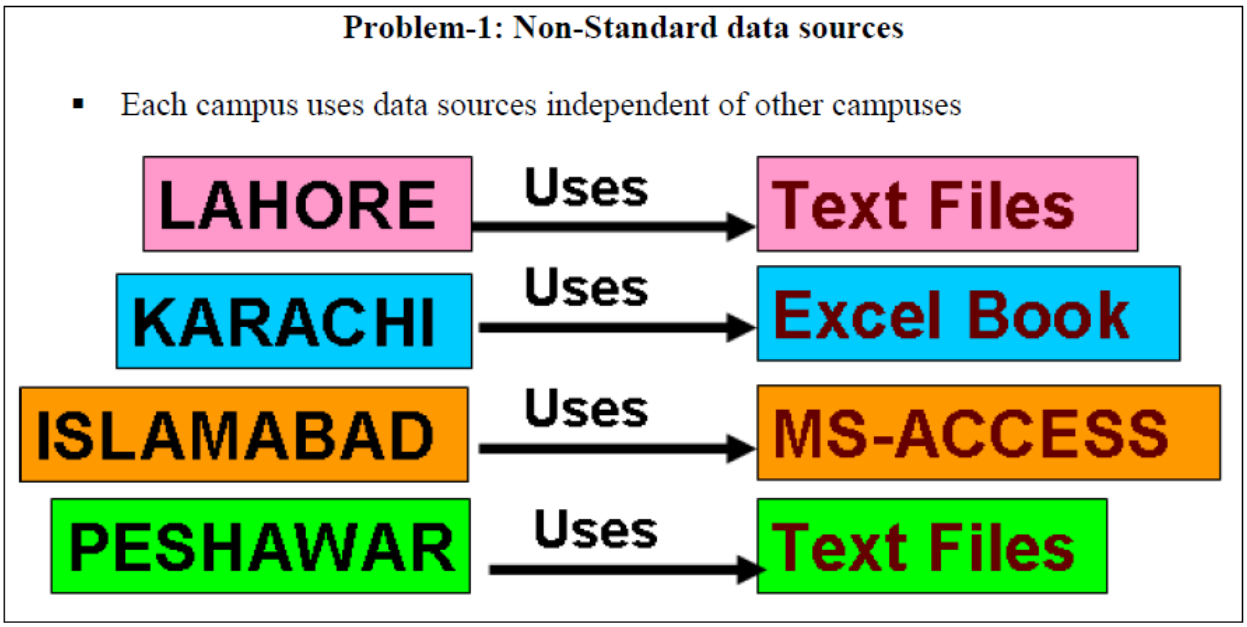
Here we need to identify semester session (fall04) through combination of Sem and Year

Lab Exercise

- Collect demographics for Peshawar campus
- Figure out problems in data at Peshawar campus
- Suggest suitable solutions to the problems identified above

Here is a small exercise. You are required to find the facts for the Peshawar campus. What problems are there in the data? And what, in your opinion, could be possible solutions for those problems.

Now by looking at each of the campus data individually, we found following problems that need to be considered and solved properly before extracting the data and ultimately loading it into the central repository.



The major problem is the inconsistent data sources at different campuses. The slide summarizes the data sources at four campuses. We can see that Lahore and Peshawar campuses are using text files while Islamabad and Karachi campuses are using MS Access and MS Excel respectively.

Problem-2: Non-standard attributes

	Identificaion	Gender (M/F)	Degree
Lahore	SID	Gender (0/1) 4th Col	BS/MS
Karachi	St_ID	M/F 5th Col	Separate books
Islamabad	Roll Num	Gender (1/0) 9th Col	Discipline
Peshawar	Reg#		

The second problem is non standardized attributes across campuses. While looking at the header of data from different campuses we came to know the following problems regarding attributes and is summarized in the table in the slide.

Each of the campuses uses different attribute name for the identification or primary keys e.g. Lahore uses *SID* while Peshawar uses *Reg#* and so on.

Different conventions for representing Gender across the campuses e.g. Lahore campus uses 0/1 while Islamabad uses 1/0 for representing male and female respectively.

Similarly, there are different conventions for representing degree attribute across different campuses.

Problem-3: No Normalized database

- **None** of the campuses uses well designed normalized database
- Each campus uses two "tables":
    (1) One table to store students' personal data
    (2) Second table to store course registration data of each student
- Each campus uses multiple files to store these two tables

Actually Lahore, Karachi and Peshawar campus does not have databases at all, so there is no concept of normalization. These campuses maintain the data in sample shown as follows:
Lhr_detail_94: Is a text file that contains the following details:

SID,Degree,Semester,Course,Marks,Discipline

Lhr_student_94: Is a text file that contains the following details:

SID,St_Name,Father_Name,Gender,Address,Date of Birth,Last Degree _

Note pad: Issues (1)

- Use of text files in record management systems is least suitable
- We cannot run any query on text file
- We cannot validate any input to text file
- Comma is used as a field separator, any erroneous placement of comma can spoil the whole record
- There is no technical way of locating any particular record

Having discussed the three major problems, lets now look at what are the issues regarding the record management tools at individual campuses. This slide and the following four list the issues related to Notepad

Note pad: Issues (2)

- If I want to locate the record of 'Mohammad Ali Nawaz' and I do not know his roll number, what would I do?
- At Lahore campus, academic officer used to do it by "Find" option of text file
- Is it a proper way? Does it work always?
- What about 'Mohamed Ali Nawaz'?

People at different campuses, including the Lahore campus have developed ways and means to answer some questions. But these so called "techniques" have their own inherent limitations. For example, if I want to find the information about a student named 'Mohammad Ali Nawaz' I can use the find command from the notepad, but what if there is a slight change in the spelling? Of course the technique is not going to work.

Note pad: Issues (3)

- If I want to count total students who belong to Multan, can I do it in note pad?
  i) No
- To achieve this purpose, admin at Lahore used to open the file in Microsof**t** Word.
- Then use "Replace with" functionality of Microsoft Word to count total occurrences of Multan.
- In 'Replace With' dialog box if I enter 'Multan' is replaced with 'Multan' & use 'Replace All' option. I can get the total occurrences of Multan. *Interesting*

Some simple questions that can be answered if there was a database cannot be answered, such as the number of students from any particular city. There can be number of short term self-developed ad-hoc mechanisms, but they are not guaranteed to succeed and have their own inherent limitations.

Note pad: Issues (4)

- Some improper ways can work for very limited cases
- We can't collect demographics in note pad
  1. Total number of male students
  2. Students with a particular age
  3. Students with a particular educational background
  4. Students with a particular CGPA
  5. Etc.

Some very simple statistics cannot be collected in the absence of a database as we have a big text file. Some of the examples are number of male students or students of particular age. We can get answer to these problems by parsing the files, but text parsing is not only very slow, but is also very complicated. All these complications and inefficiencies can be reduced, and even removed if he had a database in place.

MS-Excel: Issues (1)

- Karachi campus uses MS-Excel sheets to maintain students record
- MS-Excel is again not basically developed for this purpose
- However, it works somewhat better than note pad, as it can answer to more questions but once again in an improper way
- Both methods adopted for notepad are available here also but it can work more than that

MS Excel is better than having a big text file. For example Excel supports some simple tests and other commands that can help more efficiently answer the questions that could not be answered using a plain text file. But, still Excel is not the right way to store and keep the data for a host of reasons that we discussed in Lect-6 of the theory part.

MS-Excel: Issues (2)

- Now, I can count total number of male or female students?
- I can sort all columns on basis of gender and get all males and females clustered
- I can get student-wise particular scores

- I can get answers to many questions through conditional queries supported by MS-Excel

The slides gives a way of finding answer to some questions, but remember that we are dealing with large data sets, and for such large sets comparison sort which at best is O(n log n) really hurts.

MS-Excel: Issues (3)

- Maintenance of records in MS-Excel is better with respect to the data quality
- concerning issues
- MS-Excel recognizes the correct data type of columns
- It somewhat validates the input, i.e. illegal input is filtered

Some more benefits of Excel. At least there is a column type i.e. not all values are textual in nature, and this helps in the context of data validation.

MS-Access: Issues (1)

- MS-Access is a proper RDBMS and can work well for small databases
- At Islamabad campus, the problem is the poor design of database, not the tool
- SQL of MS-Access is not very powerful, like that of SQL Server, but it works fine to maintain records at campus level

Finally Islamabad campus at least is using the right tool i.e. Access databases, but it works for small personal databases not years of data of a single campus and then pooling together the data of multiple campuses. Thus the problem is not of the poor design (As there is no real design) but of the wrong tool. The correct choice could have been to use MS SQL server which can handle larger workloads more gracefully.

Problem Statement

- We have disparate sources of data
- We have to implement single source of truth i.e. DWH, so that decision makers can be supported to get detailed or summarized university level view, irrespective of particular campus
- In the lab exercises and working us will experience interesting and complicated issues need to be handled while moving towards single standardized source?

Thus, in view of the issues and challenges in our simulated scenario of a multi-campus university, the problem ahead can be summarized as under.

There are disparate and diverse data sources and we have to implement a DWH i.e. single source of truth that can support the decision making at the head office.

Lab 16 Part I

Extracting Data Using Wizard

Steps towards single source of truth

- Identify source systems
- Figure out the issues associated with each source system
- Extract data
- Transform data
- Load data
- Quality checks
-

Data warehouse is a single source of truth. We have to put all departmental data from desperate sources at one place in some standard form. The task is not trivial. Desperate sources of data have a lot of inherent issues. High level steps given in the slide gives just an overview of the task. First of all we have to identify the source systems. It is quite possible that each department uses different database systems or same organization at different geographical locations uses different database management systems. To put data into a single source after extracting from such diverse sources requires powerful tools especially designed to fulfill the purpose. We will use Microsoft SQL Server which is a user friendly graphical tool and makes such a complex task doable by some practice. After identification of source systems, it is necessary to study the issues that must be considered before putting all the data together. Microsoft SQL Server provides a powerful support to perform Extract, Transform and Load (ETL) data from source systems to Destination system. Finally certain steps are performed to check and improve quality of data. In this lab exercise we will perform all the above steps in a detailed manner through powerful support of Microsoft SQL Server.

Example: Student Record System: Diversity

- Identify source systems at different campuses
- Source systems are as follows
    - Lahore campus uses simple Text Files
    - Karachi campus uses MS-Excel workbooks
    - Islamabad campus uses MS-Access DB
    - Peshawar campus uses simple Text Files
    -

The task starts from analysis of source systems at different campuses. Data for Lahore and Peshawar campus is kept in simple text files, for Karachi Ms-Excel books are used and for Islamabad MS-Access is used to keep data. Furthermore table structures, date formats, conventions for gender M/F or 1/0, etc. are different from campus to campus. First we will load the data for each campus in MS-SQL Server as it is, in different databases then before putting all the pieces together all these issues will be addressed.

Example: Student Record System: Issues

- Figure out the issues related with each source system
- Issues include
- Standards and formats of stored data
- Number of tables in different source systems
- Type of columns, their number and ordering in different tables to be combined
-

Here we need to figure out the issues in source systems. As source data is distributed over different campuses therefore the issues like difference in date formats, conventions of storing gender (M/F,0/1,1/0), etc are obvious. Microsoft SQL Server has a good support to resolve these issues.

Extracting University Data

1. Lets start our practical by loading data for the university
2. We have data from four different campuses
3. Initially we will develop four different databases, one for each campus, and load corresponding data
4. Then we will transform and standardized each database
5. Finally we will combine all the four databases to single source of truth (DWH)
6. At each step we will run queries to collect demographics

For loading data for the university, it is required to load the data for four campuses, separately and as it is, into the MS-SQL Server. Once all data is loaded to SQL Server then the tasks of transformation and standardization would be started. First we will transform the database of each of the campuses individually. Then we will standardize the databases of four campuses separately. Finally, the data from four different campuses will be put together.

Extracting Data Using Wizards

- *Import and Export Data Wizard* provides the easiest method of loading data.
- The wizard creates package which is a collection of tasks
- Tasks can be as follows:
    - Establish connection through source / destination systems
    - Creates similar table in SQL Server
    - Extracts data from text files
    - Apply very limited basic transformations if required
    - Loads data into SQL Server table

After addressing the issues we decide to select a suitable tool in SQL Server to resolve these issues. At this stage we are not performing transformations rather we are just copying data from source to destination. For this purpose the easiest method is the use of wizard. Wizard would create package for us including all required tasks as:

- Establishes connection through source / destination systems
- Creates similar table in SQL Server
- Extracts data from text files
- Applies very limited basic transformations, if required
- Loads data into SQL Server table

Extracting Data for Lahore Campus

- **First of all load data for the Lahore campus**
    - Connect to source Text files
    - Connect to Destination SQL Server
    - Create new database ‘Lahore_Campus’
    - Create two tables Student & Registration
    - Load data from the text files containing student information into Student table
    - Load data from the text files containing registration records into Registration table
- **Import/Export Wizard is sufficient to perform all above mentioned tasks easily**

Loading data for Lahore campus includes following tasks:

1. Connect to source Text files

Since there are many text files for Lahore campus, we need to load those text files separately. First of all, select the file that is to be loaded first.

2. Connect to Destination SQL Server

In this case our source system is a text file. For transformation and standardization we will load all data as it is from source file to the SQL server and then through powerful tools of SQL Server, we will perform these intended task.

3. Create new database 'Lahore_Campus'

To load data for four campuses we will develop four separate databases. So, to load data for Lahore campus we will create a new data base named 'Lahore_Campus'.

4. Create two tables Student & Registration

All files containing student information will be loaded in one table Student and all other files containing registration information will be loaded in other table Registration. After this step we will have two populated tables only.

5. Load data from the text files containing student information into Student table.

6. Load data from the text files containing registration records into Registration table

Import/Export Wizard is sufficient to perform all above mentioned tasks easily. So we will use the wizard as it can provide us good functionality in this scenario.

Seven Steps to Extract Data Using Wizard

1. Launch the Wizard
2. Choose a Data Source
3. Choose a Database
    - Specification of file format incase of Text files
4. Specify the Destination
5. Choose Destination Database
    - Selection of existing database or creation of a new database
6. Select a table
    - Selection of existing table or creation of a new table
7. Finalizing and Scheduling the package

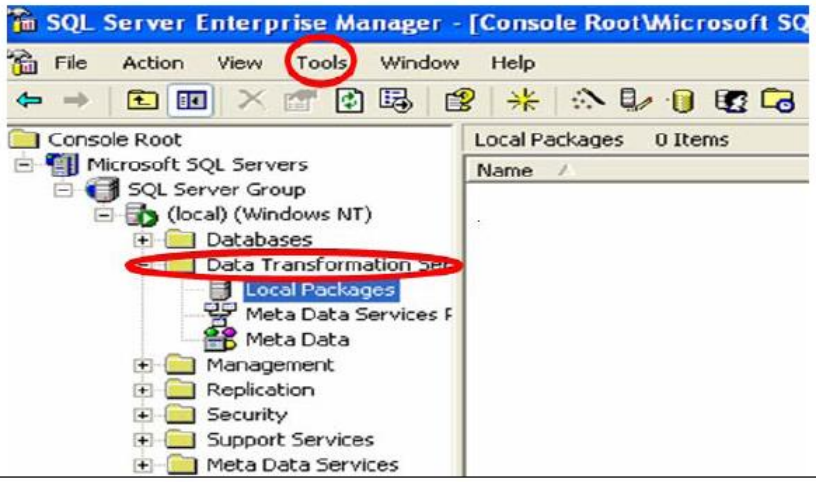
The slide states seven simple steps to create a package for data loading through wizard. Let's discuss each of the steps in detail.

Step1: Launch the wizard(1)

- **Two methods to launch the wizard**
- Start > Programs > Microsoft SQL Server > Import & Export Data
- Start > Programs > Microsoft SQL Server > Enterprise Manager
    - 1) On console root drop Data Transformation Service node
    - Tools > Data Transformation Service > Import/Export data

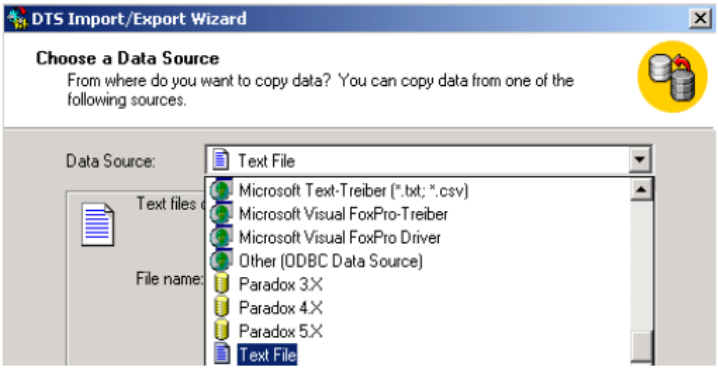
These are two different methods to launch the wizard. We can use either.

Step1: Launch the wizard(2)



The slide shows the main screen of SQL Server enterprise manager. In the left pane we have Console root. We can see Data Transformation services highlighted. Expand the node mentioning Data Transformation Services and then press Tools in the menu bar. This will lead you to launch the wizard to load data.

Step2: Choose a Data Source(1)



After launching the wizard, first of all we will be welcomed by the wizard through a welcome screen having a button 'NEXT'. On pressing 'NEXT' we will see the above window. This window is basically allows to perform step 2 of creating a package through wizard. In this window we can see a drop down menu 'Data Source'. This menu shows all possible connections that are available through SQL Server. Connection will be selected according to source system. In our case, source system is a simple text file. Therefore select the last option text file from this drop down menu.

Step2: Choose a Data Source(2)

- Data source for Lahore is a Text file
- After specifying data source, Browse first file to be loaded from Lahore data
- "Lhr_Student_94.txt", is a text file that contains students data of Lahore campus
- First of all browse this file to load

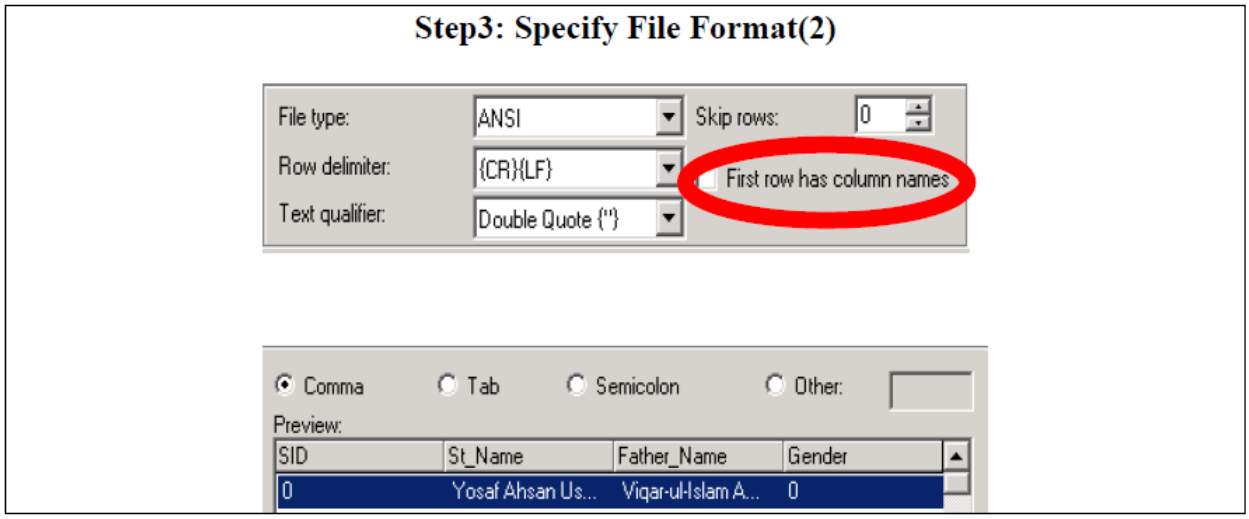
After selecting text file as a source system, we will see a new option on the lower part of same window. The option is to browse the text file considered as a source system. To load data for Lahore campus first of all select the file **'Lhr_Student_94.txt'**. Browse option allows us to have an Open File dialog box, which will let us to locate file through navigation. We will locate the directory in which our source data is placed, select the file 'Lhr_Student-94' from Lahore folder. Then press next to move to step 3 of wizard.

Step3: Specify File Format(1)

- **After specifying file, Wizard asks for file format like**
  - Columns are fixed length fields or delimited by some character
  - First row contains header information (column/field) names or directly data
  - What are the text qualifiers
  - What is the file storage format
- **How many rows, if required, should be skipped from start**

Third step in creation of package through wizard is to specify the format of the file to be loaded. In this new screen we are supposed to provide following information to the SQL Server:

- Whether the source file has fixed length columns or delimited length columns.
- Does the first row contain column name?
- Do we have text qualifiers in our source text file? If yes then select the specified from drop down list menu.
- What is the file storage format?
- Do we want to skip rows from start of file? If yes, then how many rows are supposed to be skipped?
- The answers to all above stated questions are provided through graphical user interface objects like check box, radio buttons and drop down list menus.



Here we can see the screen to input answers of the questions asked on previous slide. Each and every object is self explanatory. First of all you will see the above screen and then on pressing next we need to specify the column delimiter that is used in our text file.

Step4: Specify the Destination(1)

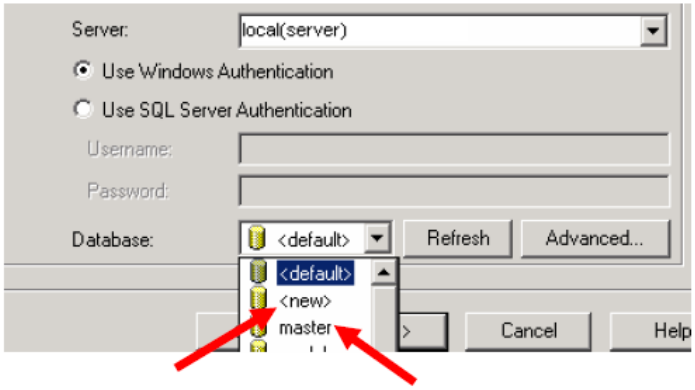
- As many options for destinations are available as were for the source (in step 2)
- By selecting Text files as destination, data extracted from text file would be stored in another text file

- Incase of given scenario we want to load data in SQL Server, which is a default option for destination

Step 4 in creation of a package through wizard is to specify the destination database. This needs to establish a connection with destination system first. For this purpose select the correct connection from the drop down menu **'Destination'** which contains all possible connection options as were available for source in step 2.

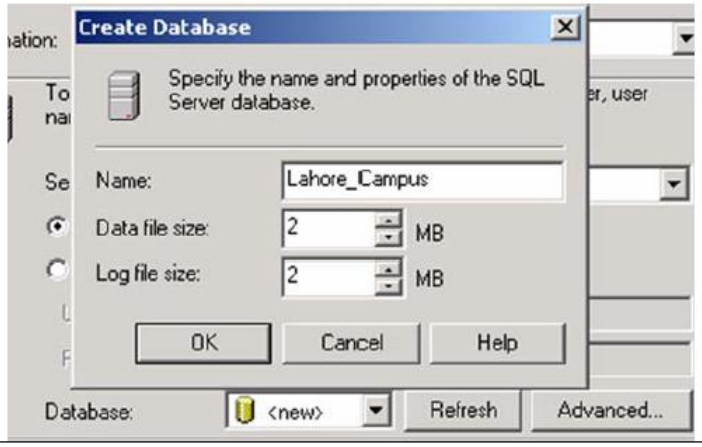
In case we select Text file as a destination then the data extracted from one text file will be loaded in another text file. As we have planned to load data in SQL Server, therefore select SQL server connection, which is by default selected.

Step4: Specify the Destination
Choose Database



On selecting SQL Server as destination we can see options appeared on the same window. From here we are just required to select the destination database in destination system. This can be done through a drop down list menu titled as Database. This drop down list menu is showing all databases available to this user in SQL server. **<new>** option allows us to create a new database. We will select new as we want to create separate database initially for each campus.

Step5a: Creating a New Database



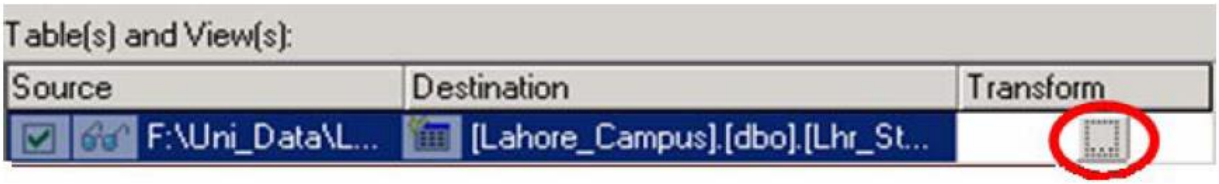
On selecting **<new>** option we can see this Create database pop up dialog box. This dialog box is asking for the name of database and the space needed for that database. Write name as 'Lahore_Campus' and then press OK. Then press NEXT

Step6: Creating a New Table

- Source contains the name of input file
- Destination is a new table with same structure as that of source text file
- By default, name of destination table is same as that of text file i.e. "Lhr_Student_94", Rename it as "Students"

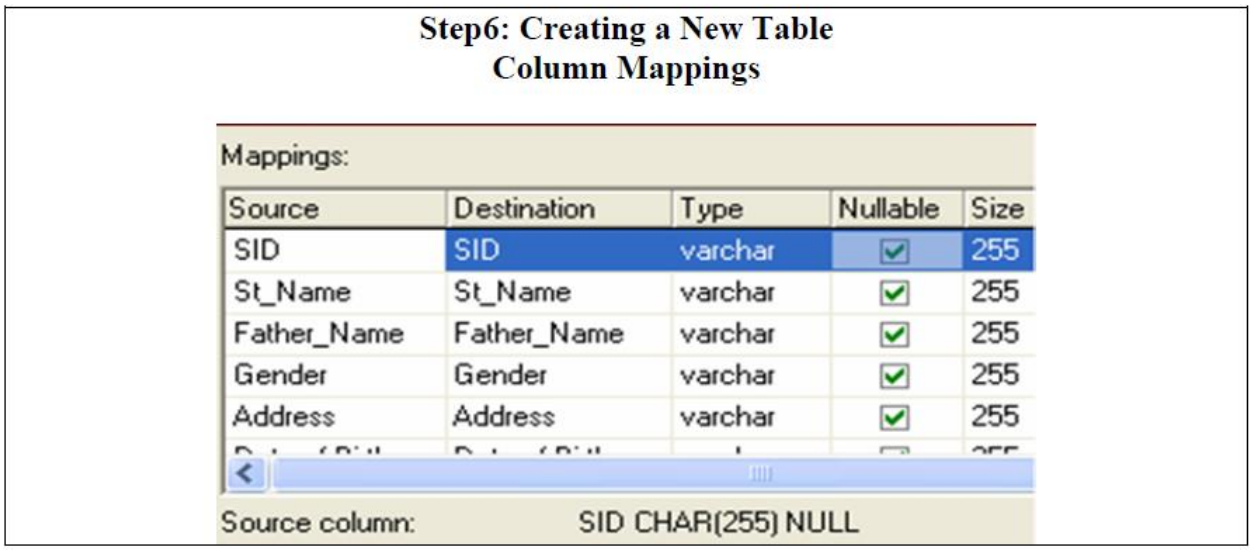
After selecting the destination database from existing databases or new database, we press next. This time we will see the above window on the screen. This window contains 3 columns source, destination and transform. Source contains the name of input file. Destination shows the name of the destination table in destination database which was selected in previous step. The wizard checks the names of all available tables in destination database. Select the table for loading whose name is exactly the same as of source table/ view/ text file. If none of the table has matching name then wizard generates code to create new table with exactly the same name and same number of columns as that of source table/view/text file. Wizard generates table with all columns having same data type i.e. VARCHAR(255) In this case our destination database is empty, as it has not yet physically created. Therefore, wizard selects the name for new table 'Lhr_Student_94' as it is the name of source text file. To rename this text file we can double click the destination highlighted row. As shown in the figure.

Step6: Creating a New Table: Transform



- Name and order of columns in new table can be seen through "Transform"
- Names, data types and order of columns in new table can be changed through "Transform"

Transform column is used to apply transformations that are available through wizard like changing the data types of the columns, order of the columns and so on. If we click the transform button, we can see the mapping between source and destination columns.

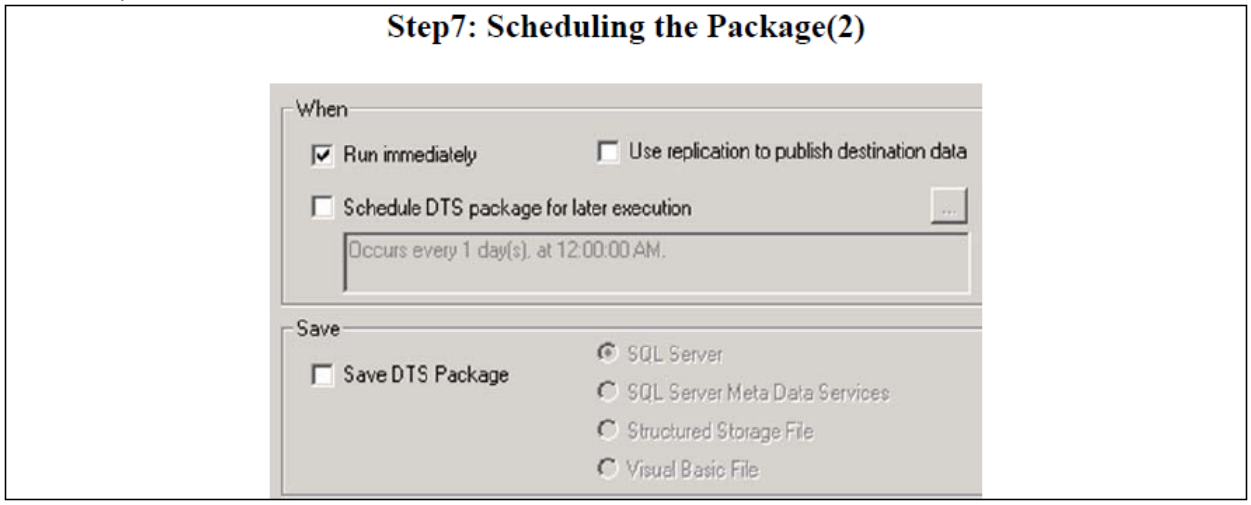


On clicking transform we can view the dialog box showing mapping between source and destination columns. The third column is showing data type of destination column. By default all types are selected as varchar of size 255. By default *Nullable* option is checked that means the corresponding columns can contain null values as well. A pointer variable varchar to hold character array of length 255 requires a lot of memory to be consumed especially when we are dealing with a huge input data. So we should change the type according to required lengths, for example, for Gender one character is enough as gender may be M/F or 0/1.

Step7: Scheduling the Package(1)

- In the preceding six steps, a package has developed including tasks of establishing connections, extraction and loading of data
- Whenever the package would run all the tasks would be executed

After finalizing the mapping we can press next for finalizing the package. On saving the package all steps would be written in a script file. Whenever the script would be run all steps would be executed and tasks would be performed.



The above dialog box provides following options:

1. Run the package immediately
2. Schedule DTS package to run periodically
3. Use of replication to publish destination data (we are not concerned with this option currently)
4. Save package

For saving package we need to select the destination location that can be one of following:

a. SQL Server
b. SQL Server Meta Data Services
c. Structured storage file
d. Visual Basic file

Now, we have successfully completed the package and it is ready to run. It can be executed manually or automatically by scheduled option.

Execution of a package

1. Connection with source (Text file) is established
2. Connection with destination (MS-SQL Server) is established
3. New Database at destination is created
4. New table is created
5. Data is extracted from source
6. Data is loaded to destination

When this package will be executed either automatically due to scheduled option enabled or manually following tasks will be performed:

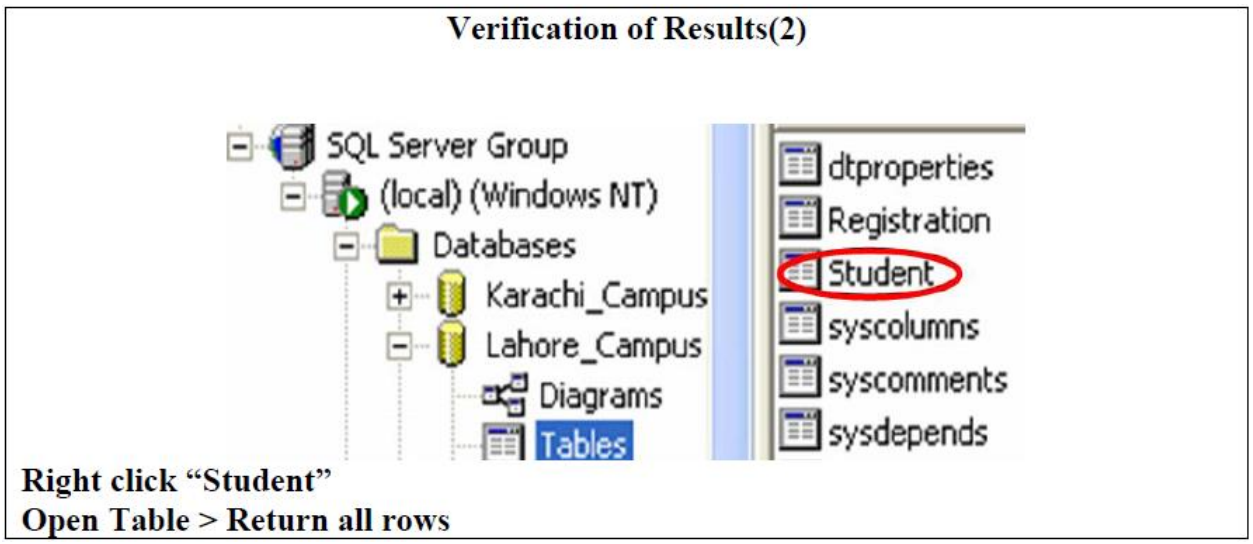
- Connection with source (Text file) is established
- Connection with destination (Ms-SQL Server) is established
- New Database at destination is created
- New table is created
- Data is extracted from source
- Data is loaded to destination

Execution can be completed successfully or it may be stopped and roll backed due to some error. In case of successful completion of execution all the transactions will be committed to the database otherwise, if some error occurs, execution will be terminated abnormally and all transactions will be rolled back. In second case when we will access the database we will find it in the state that was before the execution of package.

Verification of Results(1)

- Results can be verified by view resultant table and its rows
- New Database “Lahore_Campus” can be accessed through SQL Server Enterprise Manager
- Expand the tree on the left pane
- Local > Databases > Lahore_Campus > Student

After successful completion of the package we can verify the results by viewing the destination table and its rows. To access new database ‘Lahore_Campus”, we will use SQL Server Enterprise Manager. In SQL Server Enterprise Manager we will expand the console tree in the left pane and drop the databases, there we can find ‘Lahore_Campus’. Double click the node tables and locate Student table in right pane window.

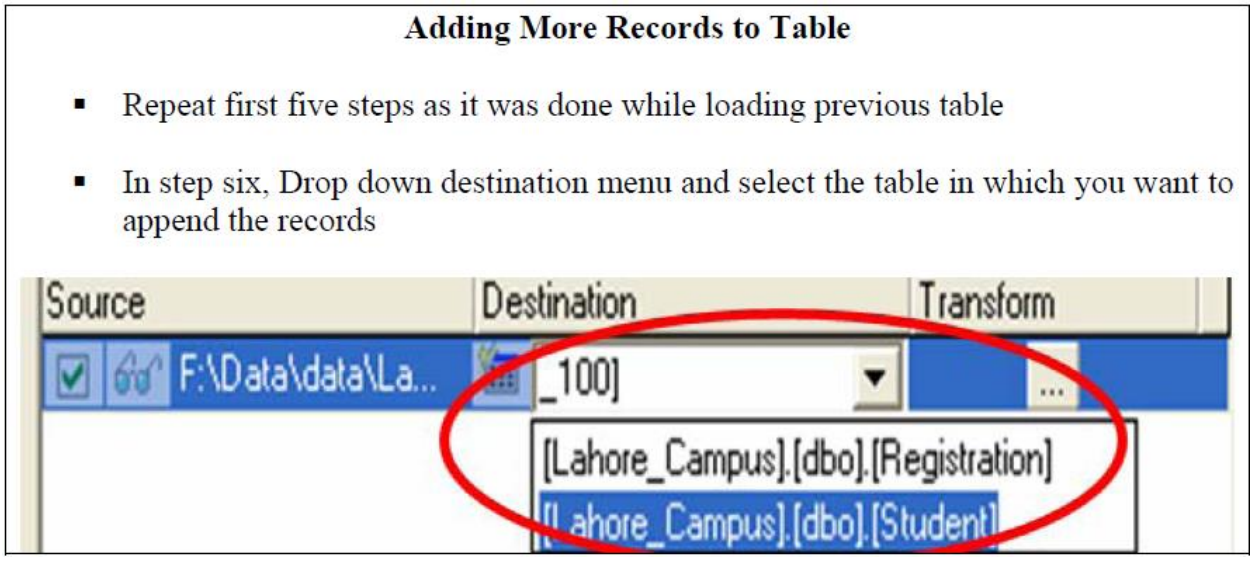


Right click student table and select option '**Return all rows'** in sub menu open Table. This will show you all the rows loaded in the destination table.

Adding More Records to Table

- The Loaded file contains the data for batch 1994 only
- Data for remaining ten batches is stored in ten separate text files
- In SQL Server we will be having only one student table for all batches in a campus, so data for remaining table is required to be added in same table

By this step destination table contains the data for one batch only, as the source file was for the 1994 only. Data for remaining ten batches is stored in separate text files. We are required to add the data for remaining batches in the same table. Following slides guide you for loading data in the same table.



To load data in the same table, we are required to repeat first five steps as it was done before.

1. Launch the Wizard
2. Choose a Data Source
3. Choose a Database

Specification of file format incase of Text files

1. Specify the Destination
2. Choose Destination Database

Selection of existing database "Lahore_Campus"

The difference comes in 6th step when we select destination table. To choose destination table drop the menu in Destination column and locate the name of required table that is "Student". That is all we are required to add records in the same table.

Load All BS &MS Student Records For Lahore

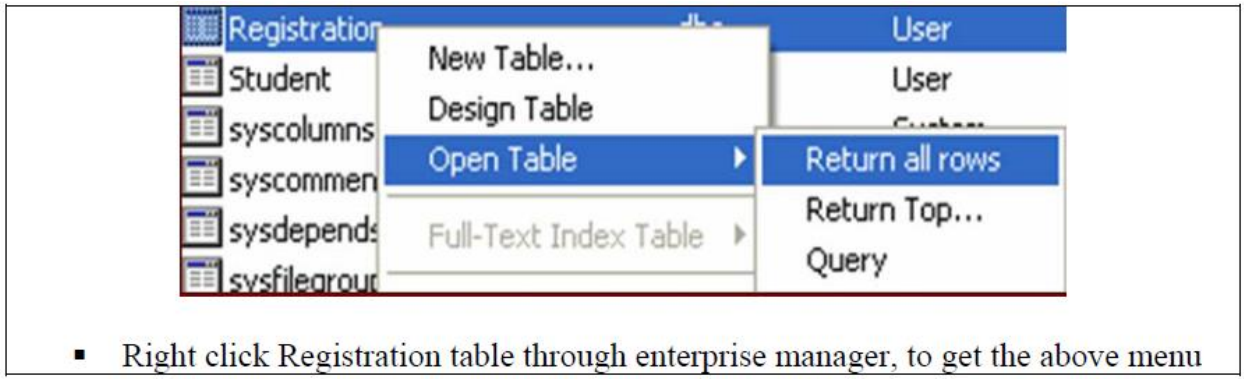
- Repeat the same procedure for remaining nine files for the batches 1996 to 2004
- Then load "Course Registration Details" data in a new table "Registration"
- We have eleven files for Registration data, one for each batch
- Load all files in table, "Registration"

We are required to repeat the same procedure for remaining nine files for the batches 1996 to 2004. After loading all text files containing student data of Lahore campus, load "Course Registration Details" data in a new table "Registration". We can find eleven files for registration data, one for each batch. Load all course registration details files in table, "Registration"

Loading "Course Registration" Records

- Load "Course Registration" records, in the same way
- We have 11 BS + 4 MS text files for Registration data, one for each batch
- Load all files in table, "Registration"
- Registration records already contain a column indicating degree (BS/MS)

Similarly load all text files that contains course registration details records. We have 11 text files for BS registration detail records and 4 text files for MS registration detail records. Create new table registration and load all files in new table. After loading all course registration records for BS and MS we can see that each record would still be identified uniquely as registration table contains a column indicating the degree of a student either BS or MS.



- Right click Registration table through enterprise manager, to get the above menu

In the same way as we did for Student table we may verify the results of package targeting Registration records load.

Verification: "Course Registration" Records

	SI	Degr	Semester	Course	Mark	Dis
▶	0	BS	Fall94	CS-101	97	CS
	0	BS	Fall94	CS-102	65	CS
	0	BS	Fall94	CS-103	85	CS
	0	BS	Fall94	CS-104	50	CS
	0	BS	Fall94	SS-201	62	CS
	0	BS	Fall94	MS-301	53	CS
	1	BS	Fall94	CS-101	74	CS
	1	BS	Fall94	CS-102	78	CS
	1	BS	Fall94	CS-103	58	CS

The above slide shows the output of query "**Return all rows**" from **Registration** table.

Demographics

- Finally all the data for Lahore Campus has been loaded
- Now we can collect demographics on this data through queries

By this time we have two tables in SQL Server, Students & Registration that contain all students and registration records for Lahore Campus respectively. Before starting transformation and quality check of data, it is required to collect demographics so that we can chalk out the way of transformation.

Total students & BS Students

- **Total Number of Students**
  Select COUNT(*) AS Expr1
  From Student
- **5,200**
- **Total BS students**
  SELECT COUNT(*) AS Expr1
  FROM Student
  WHERE ([Last Degree] IN ('F.Sc.', 'FSc', 'A level', 'A-Level', 'HSSC'))
- 4,400

Total number of students

This is the simplest query that counts the total number of students registered in Islamabad campus. For each campus individually, this query provides correct results but for consolidated data for all campuses this type of query will count those students twice who admitted in one campus and then transferred to another campuses as their names would be present in the databases of both campuses. While running queries for consolidated data we must take care of such issues.

Total BS students

Repeating IDs for BS and MS students make such queries complicated which involves the separation of BS and MS students. To count total number of BS students we must separate them from MS students. In **Student** table we do not have any such direct information that can filter BS students from total students. To meet such requirements we devised a methodology of considering degree information. No doubt this is not a perfect way of filtering as the quality of result can suffer a lot due to the presence of outliers. However, in real life when we have to deal with legacy systems we need to face hundreds of such complicated issues due to bad designs and limitations of legacy systems.

In such scenarios we need to devise solutions intelligently through indirect ways.

SIDs vs. Students

- **Total unique SIDs**
  SELECT COUNT(DISTINCT SID) AS Expr1
  FROM Student
- **4,400**

SIDs are exactly equal to the BS students

Total unique SIDs

In this query we counted the distinct student IDs from student table. The answer is exactly equal to the total number of BS students because the student IDs are unique among BS students only and same IDs are repeating for BS students.

Unique identification of each student

Unique identification of each student is possible through combination of degree only. In one campus there can be only one such student who has SID=1 and he is enrolled in BS degree program. Similarly there can be only one student is possible who has again SID=1 and is enrolled for MS degree program. Each SID can be repeated at most twice, one time for BS student and other time for MS student.

Repeating Student IDs for BS and MS

- Students IDs are repeating for MS students
- After loading records of MS students in the same table with BS students, student ID is no more useful to identify each record uniquely

Now we need some more information to be used with IDs to identify each record uniquely

After loading all files for BS and MS, you will find an interesting problem. As the university was managing the records of BS and MS separately, Student ID was used as Primary key to identify students uniquely. Student ID is just an auto-increment sort of number which starts from zero for both BS and MS students. In warehouse environment when we have combined both BS and MS students, Student ID no more uniquely identifies each student in warehouse.

Solution of Repeating IDs

- Problem of repeating IDs can be resolved through the use of ‘Last Degree’ column with ID
- SID + [Last Degree] -> unique record
- 1) SID = ‘100’, [Last Degree] = ‘F.Sc.’ 2) SID = ‘100’, [Last Degree] = ‘M.Sc.’
- 1) is a BS student & 2) is an MS student
- What can be outliers here?

As ID is no more useful we need to add some additional information with ID to identify each student uniquely. If we look at table structure of student table, we can find a column Last degree that can be used

to distinguish BS students from that of MS students. But there is also a little chance of outliers like the students who are doing MS in computer science after MS in physics etc. These outliers, if exists, will be handled separately.

Solution of Repeating IDs *(Cont.)*

- Outlier can be a BS student who has already sixteen years education background (say) in Mathematics and he is again registered for BS computer science
- Such a student can have [last degree] = 'M.Sc.

At this time we will use information like if student ID is 1 and last degree is 'F.Sc.' he is an undergraduate student, and if student ID is 1 and last degree is BS then he is a graduate student who is enrolled in MS.

Male Students

- **Total Number of Male Students**

```
SELECT COUNT(*) AS Expr1
FROM Student
WHERE (Gender = '0')
```

- **3,466**

To find out the total number of male students we counted all records where gender = '0'. This gives 3,466. We can not say any thing about the quality of this result. The quality can be discussed after data profiling but not at this stage. There may be a lot of errors in data, we may have some other male students for whom gender is missing here. We may have some records with noise like '01' or '10' which out of domain of the column. But all such issues can be identified first in data profiling only not before that.

Female BS students in Telecom

- **Total Number of Female students in BS Telecom**

```
SELECT COUNT(DISTINCT r.SID) AS Expr1
FROM Registration r INNER JOIN
Student s ON r.SID = s.SID AND
s.[Last Degree] IN ('F.Sc.', 'FSc',
'HSSC', 'A-Level', 'A level') AND
r.Discipline = 'TC' AND s.Gender = '1'
```

- **365**

This query requires access of both student and registration table. Gender of student can be found from student table where as Discipline can be found from registration table only. To find the answer of such a query we need to have inner join of both tables.

Extracting Data for Karachi Campus

- **Now load data for Karachi Campus**
  1. Connect to source MS-Excel
  2. Connect to Destination SQL Server
  3. Create new database 'Karachi_Campus'
  4. Create two tables Student & Registration
  5. Load data from the Excel worksheet containing student information into Student table
  6. Load data from the Excel worksheets containing registration records into Registration table
- **Import/Export Wizard is sufficient to perform all above mentioned tasks easily**

By this time we have loaded data and collected demographics for Lahore campus only. Now we are required to load data from Karachi campus. For Karachi campus we need to load data from Excel files. Main steps of loading are as follows:

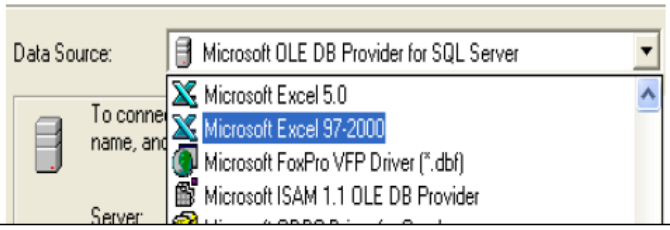
1. Connect to source MS-Excel
2. Connect to Destination SQL Server

3. Create new database ‘Karachi_Campus’
4. Create two tables Student & Registration
5. Load data from the Excel worksheet containing student information into Student table
6. Load data from the Excel worksheets containing registration records into Registration table

Again Import/Export Wizard is sufficient to perform all above mentioned tasks easily

Student data for Karachi

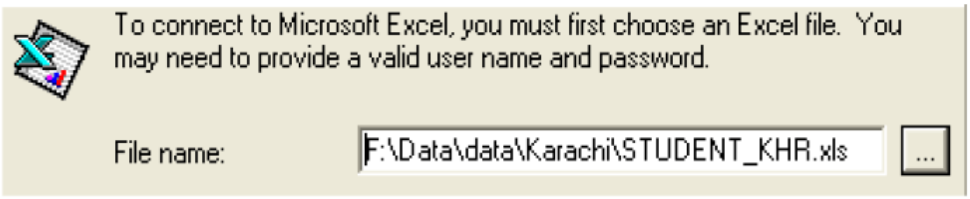
- Step1: Launch the wizard
- Step2a : Choose a data source



Step 1 is same as it was for the text files. In step two we need to select Microsoft Excel 97-2000 as data source.

Student Data for Karachi *(Cont.)*

- Step2b: Browse Student Excel-Worksheet



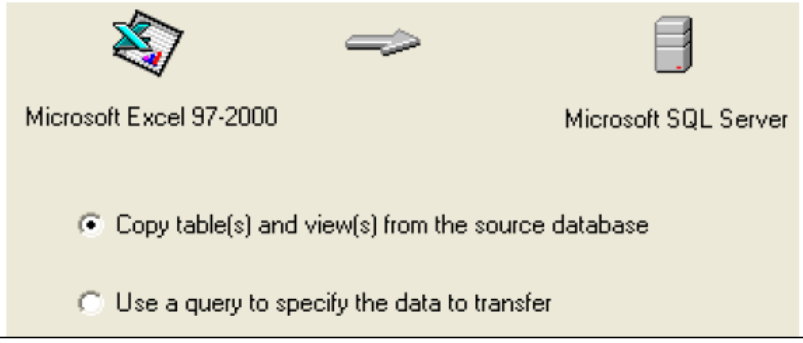
- STUDENT.xls for all BS/MS student records
- Reg_BS_KHR.xls for BS Registration records
- Reg_MS_KHR.xls for MS Registration records

After selection of MS-Excel data source, we are required to locate the data file that contains extension “.xls”. To load student data we need to locate STUDENT.xls data file as it contains data for both BS and MS students.

When we will load registration data we will require to load Reg_BS_KHR.xls file for BS students registration details and Reg_MS_KHR.xls for MS students registration records.

Student Data for Karachi *(Cont.)*

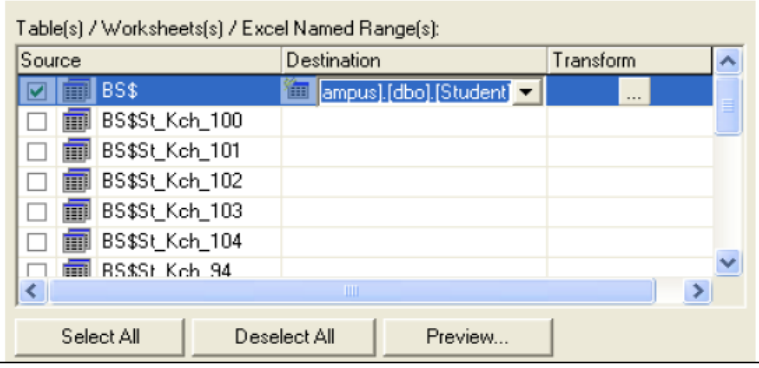
- **Step 3:** Specify Destination
- **Step 4:** Choose/Create Database
  - Karachi_Campus
- **Step 5 a:** Table/View or query



In step3 we select Microsoft SQL Server as destination and in step4 we create new database Karachi_Campus in the same way that we adopted for Lahore_Campus. As some particular sort of queries can be run in MS Excel therefore wizard asks us whether we want to extract data from source through query or we want to copy complete table/view. As we do not want to filter data through query therefore we will select the other option that is copy complete table/view.

Student Data for Karachi *(Cont.)*

- **Step 5b:**
  - Choose BS to copy complete worksheet of BS students



In this screen we can see a lot of worksheets, where as in our Student_KHR.xls file there are only two worksheets, one for BS and one for MS. You can see in the dialog box that the name of worksheet BS is followed by '$'. To load complete worksheet BS select the first option 'BS$'. Following options show logical divisions with in a worksheet. Like in this case we can see an option 'BS$St_Kch_94', it means that by selecting this option only those students will be copied who belong to the batch 1994. Similarly, by selecting the option 'BS$St_Kch_100' only students of batch 2000 will be copied but if we select the option BS$ then all students in worksheet BS irrespective of their batch will be copied.

Student Data for Karachi *(Cont.)*

- **Step 5b *(Cont.)* :**
- In step 5b you can see two columns Source and Destination
- If we want to copy all records from BS worksheet we need to check 'BS$'
- If we want to copy only those records from BS worksheet that belongs to year 2000 we need to choose BS$St_Kch_100
- Similarly BS$St_Kch_101 belongs to records of year 2001 and so on

In step 5b we can see two columns Source and Destination. If we want to copy all records from BS worksheet we need to check 'BS$'. If we want to copy only those records from BS worksheet that belongs to year 2000 we need to choose BS$St_Kch_100. Similarly BS$St_Kch_101 belongs to records of year 2001 and so on

Student Data for Karachi *(Cont.)*

- **Step 5b *(Cont.)* :**
- For Ms-Excel, it is a convention to use '$' after worksheet name like 'BS$' / 'MS$'
- If there are any logical divisions of records within an Excel worksheet, it is written after '$' sign like 'BS$St_Kch_94'
- In the given dataset of Karachi BS Excel worksheet contains data records for eleven years (94–04)and MS Excel worksheet contains data records for 4 years (01-04)
- Records with in an Excel worksheet are logically divided on annual basis

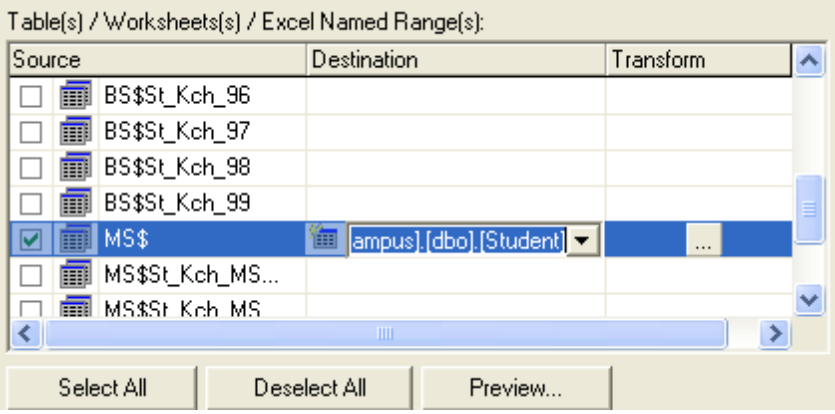
hence we can see options like 'BS$ST_Kch_94'

For MS-Excel, it is a convention to use '$' after worksheet name like 'BS$' / 'MS$'. If there are any logical divisions of records within an Excel worksheet, it is written after '$' sign like 'BS$St_Kch_94'. In the given dataset of Karachi BS Excel worksheet contains data records for eleven years (94–04)and MS Excel worksheet contains data records for 4 years (01-04). Records with in an Excel worksheet are logically divided on annual basis hence we can see options like 'BS$ST_Kch_

Student Data for Karachi *(Cont.)*

- **Step 5b *(Cont.)* :**
- As we want to copy all records from BS worksheet and all records from MS worksheet as well, therefore, we will choose two options from sources BS$ and MS$
- As Records from both worksheets are required to be copied into a single table Student, therefore, rename destinations against both sources as "Students"94'.

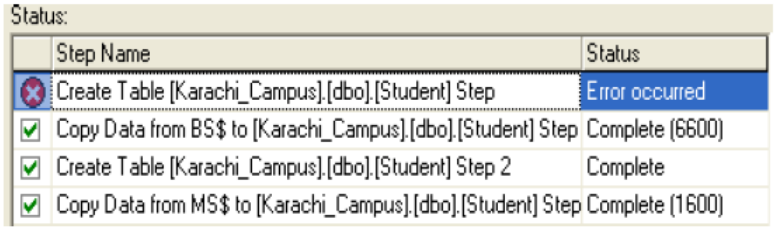
Loading data from MS-Excel workbook provides us facility of loading data in all worksheet through single package. On the same dialog box, scroll down and find option MS$ under the column Source. By this way you can load other worksheet MS in the same table. Locate the row having check box against MS$ and rename the destination as 'Student'. So that all data from both worksheets load in the same table Karachi.



So, finally two options should be checked under the column Source, one is BS$ and the other one is MS$. Against both options the table name should be the same i.e. Student, because our destination is same. We can use third column transform, by pressing button against the selected row. This option provides us the same mapping view that we have seen earlier, while loading text file.

Student Data for Karachi *(Cont.)*

- **Step 6:**



- Run the package immediately
- Status after package completion is as follows:

After this, run the package. After successful execution we can see the above window. We can see that there were four steps in this case:

1) Create table student
2) Load data of Karachi BS students
3) Create table Student
4) Load data of Karachi MS students

As we checked two options therefore package will try to create the two destination tables. But both tables have same names therefore SQL Server database will not allow to create two tables with the same name 'Student'. Hence out of four, Task three will be terminated with an error '*Student table already exists*'. As no new destination table is created for MS student therefore finally all records of MS worksheet will be loaded in the same table 'Student', due to same destination name. Same information can be seen in above dialog, but these errors are not reported in the order in which they occur. Therefore we can see first task failed where as it is third in reality.

Student Data for Karachi *(Cont.)*

- **Step 6 *(Cont.)* :**
  - It can be seen that a task listed first in previous figure is failed
  - Choosing Source 'BS$' and selecting destination table 'Student' means
  - create table student and copy all records from destination worksheet to student table
  - Similarly Choosing source 'MS$' and selecting destination table 'Student' means exactly the same
  - Hence package tries to create table student twice, once for BS and then for MS

It can be seen that a task listed first in previous run failed. Choosing Source 'BS$' and selecting destination table 'Student' means create table student and copy all records from destination worksheet to student table. Similarly Choosing source 'MS$' and selecting destination table 'Student' means exactly the same. Hence package tries to create table student twice, once for BS and then for MS

Student Data for Karachi *(Cont.)*

- **Step 6 *(Cont.)* :**
  - But second time SQL does not allow to create the table with the same name
  - Therefore a task to create table again fails
  - As there exists only one table student therefore all data for BS and MS is copied to the same table
  - Rest of the three tasks are successful
    - Create table student
    - Copy BS records worksheet
    - Copy MS records worksheet

But second time SQL does not allow to create the table with the same name. Therefore a task to create table again fails. As there exists only one table student therefore all data for BS and MS is copied to the same table. Rest of the three tasks are successful:

1. Create table student,
2. Copy BS records worksheet &
3. Copy MS records worksheet

However, their order of reporting is not the same in which they actually occurred.

Registration Data for Karachi

- Similarly load Registration data for Karachi Campus
- There are two Excel workbooks for Registration data of Karachi
  - Reg_BS_KHR
  - Reg_MS_KHR
- Reg_BS_KHR contains six worksheets each containing registration records of two batches
- Reg_MS_KHR contains only single worksheet contianing all records of 4 batches

Now we are required to load registration detailed records for Karachi campus. There are two MS-Excel workbooks, one for BS & the other one for MS. Workbook for BS contains 6 worksheets, each sheet containing records for 2 batches whereas the worksheet for MS contains all records in one worksheet. We need to load all worksheets complete.

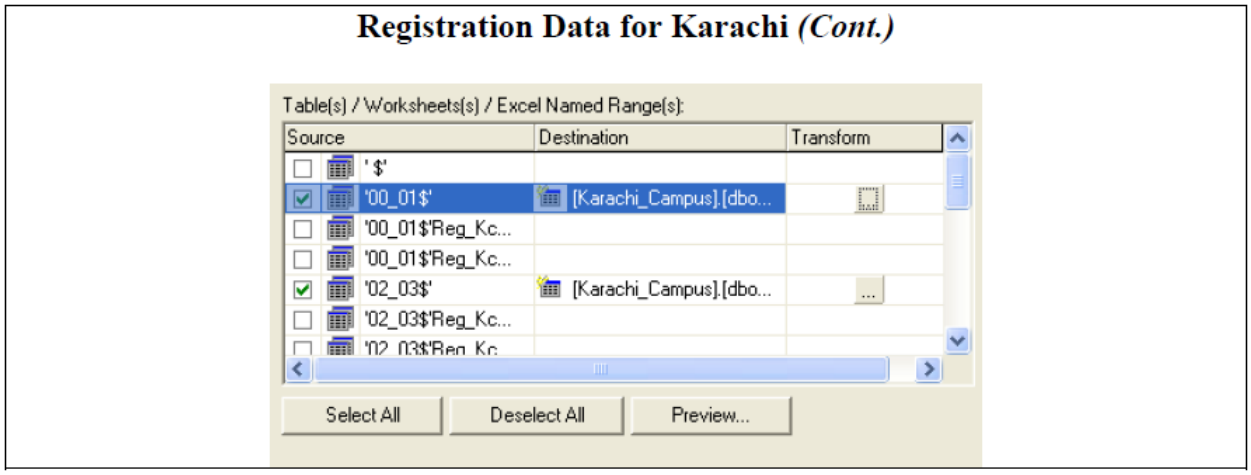
Registration Data for Karachi *(Cont.)*

- First of all load all BS records into a new table 'Registration' in database 'Karachi_Campus'
- Then append all Ms records into the same table 'Registration'
- All these tasks can be performed through wizard in the same way as previous data was loaded

For loading all BS records in a new table 'Registration' we can write only one package.

Then we will develop another package that will append all MS records in Registration table. This all can be done in the similar way as we have done before.

To select complete worksheet to be copied we need to check the row indicating the name of worksheet followed by immediate '$' without any logical division. Then set destination table as registration. All registration records either for BS or for MS should be loaded in the same registration table.



The above slide shows that we are selecting only those options that are showing name of worksheet followed by '$; sign. Against both options we are setting destination as 'Registration'.

Registration Data for Karachi *(Cont.)*

- As the source is again an Excel book therefore we have sources with same convention i.e. Worksheet name followed by '$' and logical division of work sheet
- Check only those sources that corresponds to full worksheet like '94-95$', such sources always miss the name of logical division after '$' sign

As the source is again an Excel book therefore we have sources with same convention i.e. Worksheet name followed by '$' and logical division of work sheet. Check only those sources that correspond to full worksheet like '94-95$', such sources always miss the name of logical division after '$' sign.
Checking each of the check boxes, results in two tasks in the final package. One is to create a new table, always, with the name specified in destination with the same structure as specified by the source table. As we have set same name for destination table that is Registration, therefore only one task of new table creation will succeed rest of all Create Table task will fail and data will be loaded in the same table Registration.

Registration Data for Karachi *(Cont.)*

- Status of completion is as follows

	Step Name	Status
✗	Create Table [Karachi_Campus].[dbo].[Registration] Step	Error occurred
✓	Copy Data from '00_01$' to [Karachi_Campus].[dbo].[Regist...	Complete (57600)
✓	Create Table [Karachi_Campus].[dbo].[Registration] Step 2	Complete
✓	Copy Data from '02_03$' to [Karachi_Campus].[dbo].[Regist...	Complete (36000)
✗	Create Table [Karachi_Campus].[dbo].[Registration] Step 3	Error occurred
✓	Copy Data from '04_05$' to [Karachi_Campus].[dbo].[Regist...	Complete (7200)
✗	Create Table [Karachi_Campus].[dbo].[Registration] Step 4	Error occurred

This time it can be easily identified that only those tasks are failed that attempted to create a new table Registration. Rest of all tasks of copy data copied data into same destination table.

Registration Data for Karachi *(Cont.)*

- All tasks intended to create a new table with the same name 'Registration' are Failed
- All the records are appended to already created single table 'Registration'
- Similarly append the Records for MS students in the same Registration table

All tasks intended to create a new table with the same name 'Registration' are failed. All the records are appended to the already created single table 'Registration'. Similarly append the Records for MS students in the same Registration table

Table(s) / Worksheets(s) / Excel Named Range(s):

Source	Destination	Transform
'01_02_03_04$'	dbo.['01_02_03_04$']	...
'01_02_03_04$'R...	[Karachi_Campus].[dbo].[Registration]	
'01_02_03_04$'R...	[Karachi_Campus].[dbo].[Student]	
'01_02_03_04$'R...		
'01_02_03_04$'R...		
Sheet2$		
Sheet3$		

Select All | Deselect All | Preview...

Similarly load MS data from the workbook **Reg_MS_KHR.** Again set Registration table as destination table and perform remaining steps as it is. As Registration table already exists, again create table task would fail and data will be loaded in the same Registration table, appending new records.

Demographics (1)

- Finally all the data for Karachi Campus has been loaded
- Now we can collect demographics on this data through queries
- First we try the same queries that we used for Lahore Campus

Finally all the data for Karachi campus has been loaded. For collection of demographics, first of all we try the same queries that we used for Lahore campus. As the table names are same i.e. Student and Registration, therefore, those queries should work here also.

Demographics (2)

- **Total Number of Students**
  Select COUNT(*) AS Expr1
  From Student
- **8,200**
- **Total BS students**
  SELECT COUNT(*) AS Expr1
  FROM Student
  WHERE ([Last Degree] IN ('F.Sc.', 'FSc', 'A level', 'A-Level', 'HSSC'))
  Error, [Last Degree] invalid column

Total Number of Students

Select COUNT(*) AS Expr1 From Student;

This query returns 8,200, which is correct. This is exactly the same query that we have already run for Lahore campus. Due to the standardization of table names we get this benefit.

Total BS students

SELECT COUNT(*) AS Expr1
FROM Student
WHERE ([Last Degree] IN ('F.Sc.', 'FSc', 'A level', 'A-Level', 'HSSC'))

This query returns **Error, [Last Degree] invalid column.** Because in Karachi campus student table, there is no column name as [Last Degree], but same information is stored in qualification column.

Demographics (3)

- At Karachi campus, correct column is 'Qualification' instead of [Last Degree], similarly names of others columns are also different from those of Lahore, therefore, same queries are not applicable here
- **Correct Query to count BS students is**
  SELECT COUNT(*) AS Expr1
  FROM Student
  WHERE (Qualification IN ('F.Sc.', 'FSc', 'A level', 'A-Level', 'HSSC'))
- **6,600**

This is one of the characteristic of non standardized and heterogeneous data that same information is stored under different column name. Therefore, we need to standardize all column names and data-types so that data from different databases can be put together. Now, to extract same information for Karachi campus we need to correct the above query as

SELECT COUNT(*) AS Expr1
FROM Student
WHERE (Qualification IN ('F.Sc.', 'FSc', 'A level', 'A-Level', 'HSSC'))

Now there is no problem and query gives correct results (6,600) for Karachi campus.

There may be some outliers but that can not be identified at this stage.

Demographics (4)

- To find total number of male students we use same query that we used for Lahore except Column name 'Gender' which is 'M/F' here, so query is
  SELECT COUNT(*) AS Expr1
  FROM Student
  WHERE ([M/F] = '0')
- **0, Incorrect answer**
- Answer to the query is zero i.e. no male student in Karachi campus which is definitely incorrect

Now we try to find the number of male students at Karachi campus. We submit the same SQL query that we run for Lahore campus. The answer turns out to be zero. There must be some thing incorrect in this query otherwise it can not be zero. To identify the error we look at the data and can easily point out that the error is due to the inconsistent gender storage conventions. At Lahore campus, 1 is a convention to store female and 0 is a convention to store male whereas at Karachi campus M for male and F for female is used. To put all data in single source we need to standardize the data storage conventions.

Demographics (5)

- If we look in the data for Karachi campus we can find out the reason for the error in above query
- For Karachi campus convention to store gender information is M (male) and F (female) instead of 0 (male) and 1 (female)
- **Correct query is**

```
SELECT COUNT(*) AS Expr1
FROM Student
WHERE ([M/F] = 'M')
```

- **5,463**

Following is the correct query that can be used for Karachi campus to count the total number of male students at Karachi campus.

```
SELECT COUNT(*) AS Expr1
FROM Student
WHERE ([M/F] = 'M')
```

The answers to be 5,463.

Demographics (6)

- Total Number of Female students in BS Telecom
- We again need to correct the query which was written for Lahore Campus, and correct query is

```
SELECT COUNT(DISTINCT r.St_ID) AS Expr1
FROM Registration r INNER JOIN
Student s ON r.St_ID = s.St_ID
AND s.Qualification IN ('F.Sc.', 'FSc',
'HSSC', 'A-Level', 'A level') AND r.Disp = 'TC'
AND s.[M/F] = 'F'
```

- 551

Similarly to get the count of total number of female students in BS telecom, we need to correct the column names and data storage conventions. The correct query is as follows:

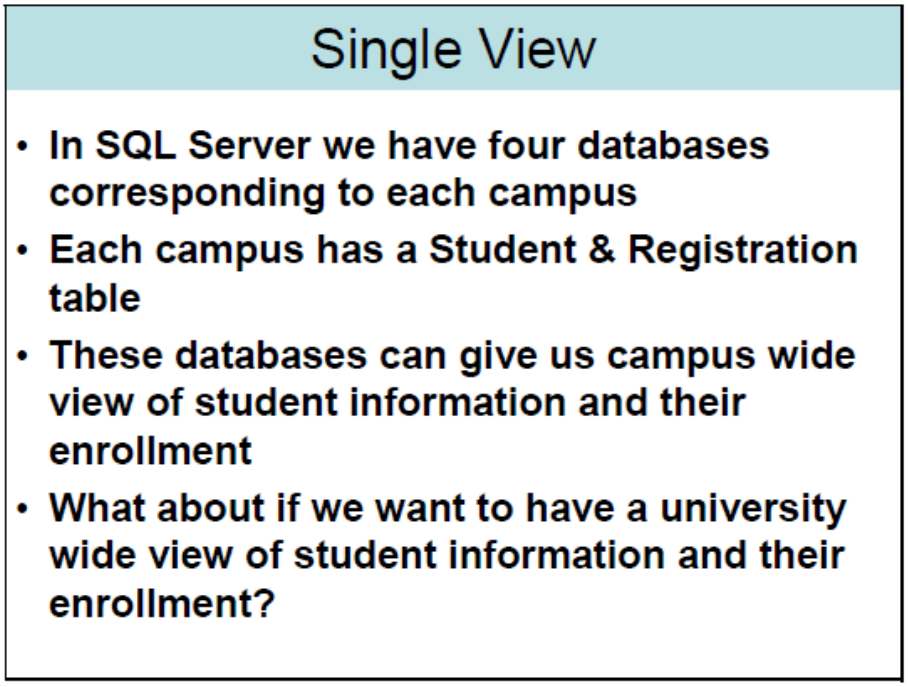
```
SELECT COUNT(DISTINCT r.St_ID) AS Expr1
FROM Registration r INNER JOIN
Student s ON r.St_ID = s.St_ID
AND s.Qualification IN ('F.Sc.', 'FSc','HSSC', 'A-Level', 'A level')
AND r.Disp = 'TC'
AND s.[M/F] = 'F'
```

Answer is 551

Lab 16 Part II

Data Profiling

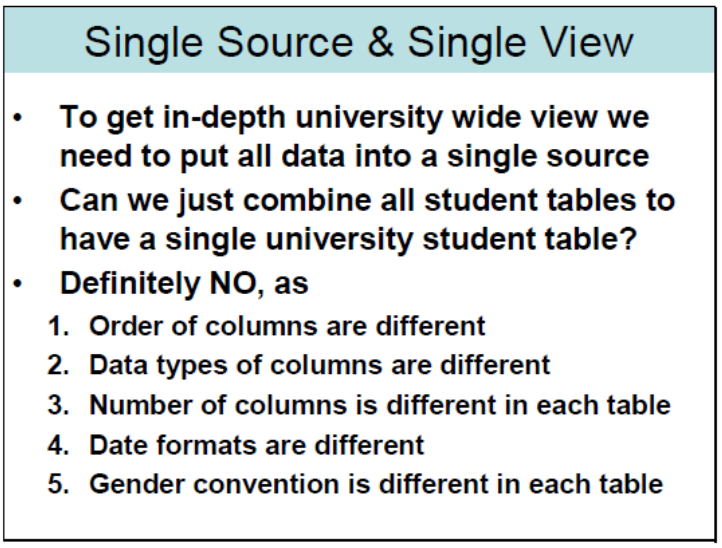
Slide 2



In SQL Server we have four databases corresponding to each campus. Each campus has a Student & Registration table. Now both tables for each campus have been loaded to MS SQL sever. It means by this stage we have four databases (one for each campus) and corresponding to each database we have two tables.

These databases can give us campus wide view of student information and their enrollment. In order to get university wide view of information we have to have consolidated information of all campuses. To consolidate we need to standardize information across all campuses. In this lecture we will step towards consolidation.

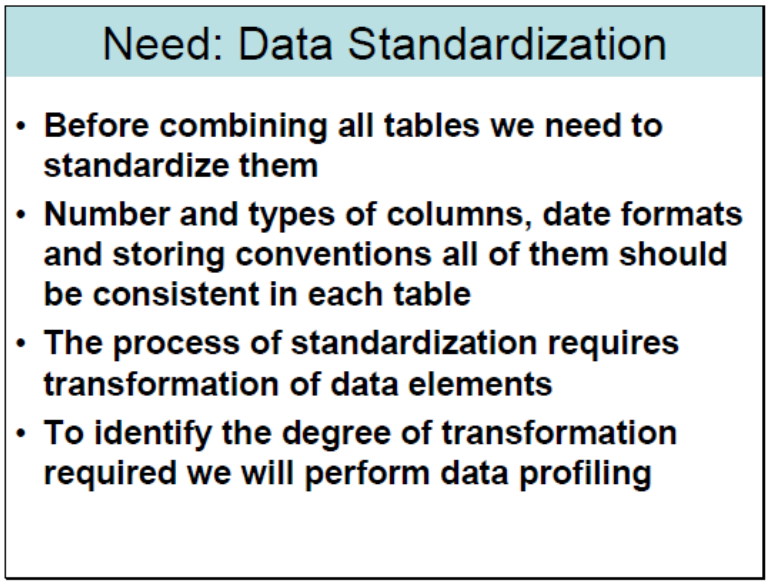
Slide 3



At this time we have four student tables in different SQL databases. To consolidate all tables to get single source of truth we can not just glue all tables because of following facts:

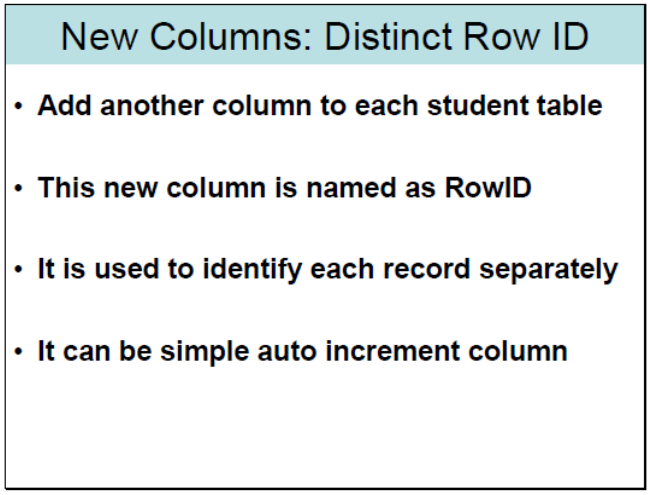
Order of columns are different
Data types of columns are different
Number of columns is different in each table
Date formats are different
Gender convention is different in each table

Slide 4



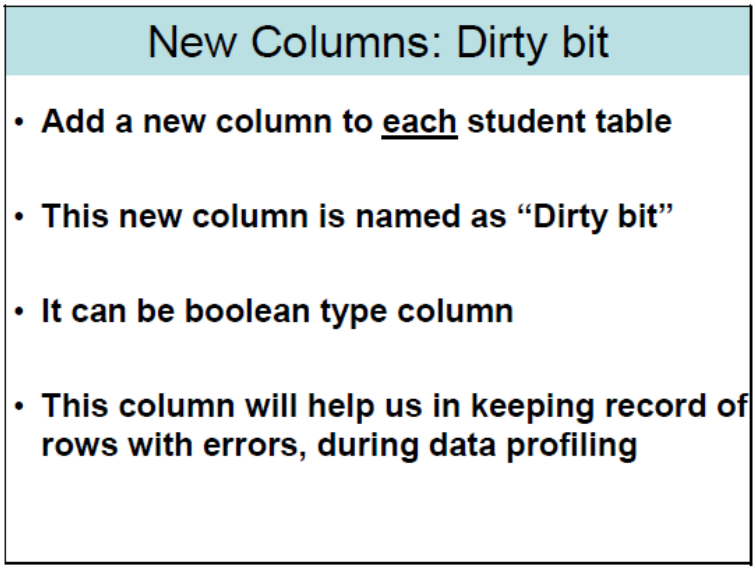
To remove the factors that are not letting us putting all data together we need to standardize all tables. Standardization process involves the consistency of number and types of columns, date formats, and storing conventions across all campuses and more. To standardize we need to transform data elements. To identify the degree of transformation required we need to perform data profiling.

Slide 5



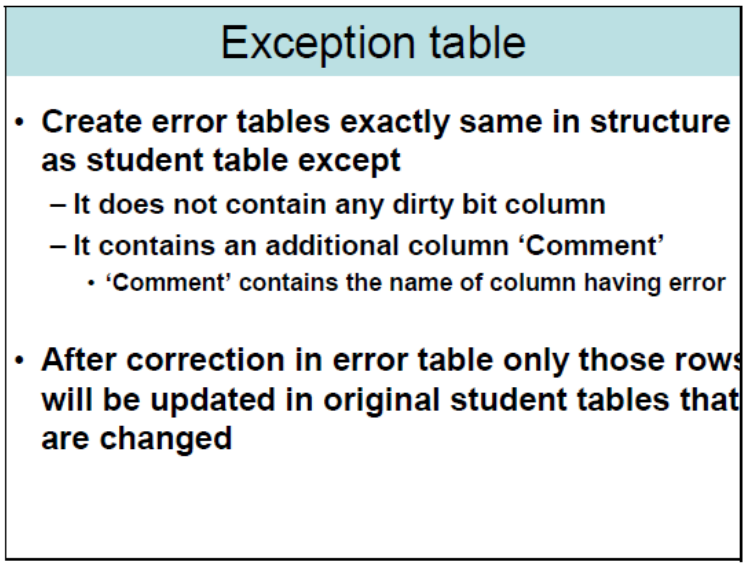
By this time, due to lack of primary key, records can not be identified uniquely. At this stage we need an attribute that can identify each record uniquely. We need such an identifying attribute at this stage because a lot of records in this stage will be put to error or exception table due to error in any of the columns. In the error table they will be corrected and later on after correction the changes will be updated in original student table. For this update we need a column like row id to join error table with student table.

Slide 6



To keep record of the rows that have been inserted into error tables due to certain errors we need an additional column in student table that will serve as dirty bit. Dirty bit of those records is set to true that are inserted in the error table.

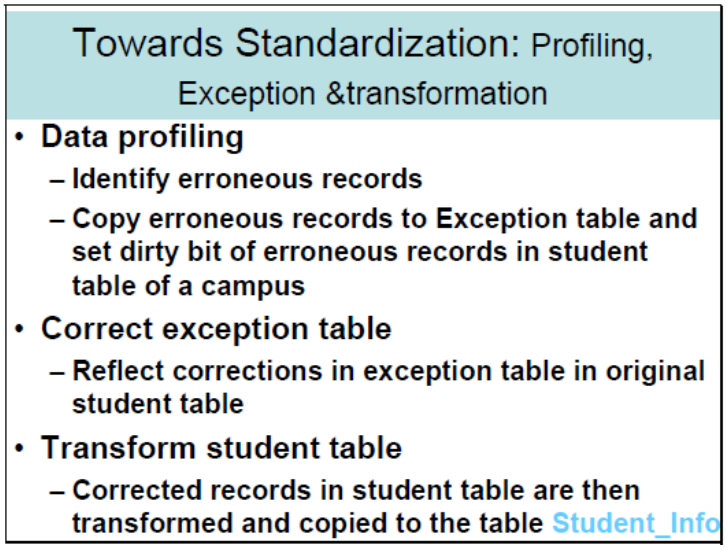
Slide 7



Error or exception table contains the copy of records that have corrupted values for any column in original student table. Error table is the copy of original student table except instead of dirty bit we will have comments column in the error table. In comments column we store the name of columns in which we encountered errors for this particular row. For example consider a record 'R', it has missing gender and incorrect date of birth.

For the record R we will have comment **[Gender], [Date of Birth].** These comments will be used while correction of error tables. After correction of error table we will update corresponding rows in the original student table.

Slide 8



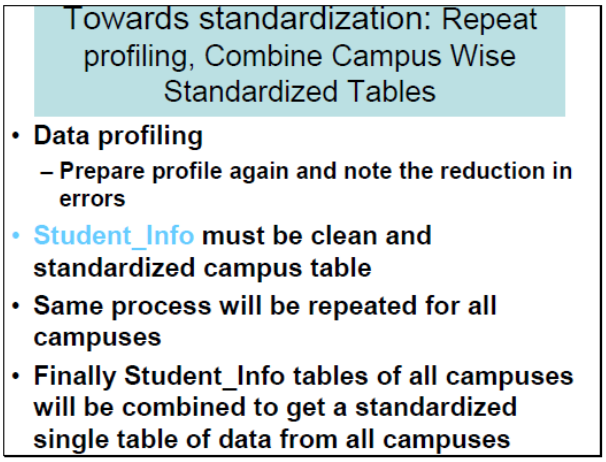
Data profiling is a process which involves gathering of information about column through execution of certain queries with intention to identify erroneous records. In this process we identify the following:

- Total number of values in a column

- Number of distinct values in a column
- Domain of a column
- Values out of domain of a column
- Validation of business rules

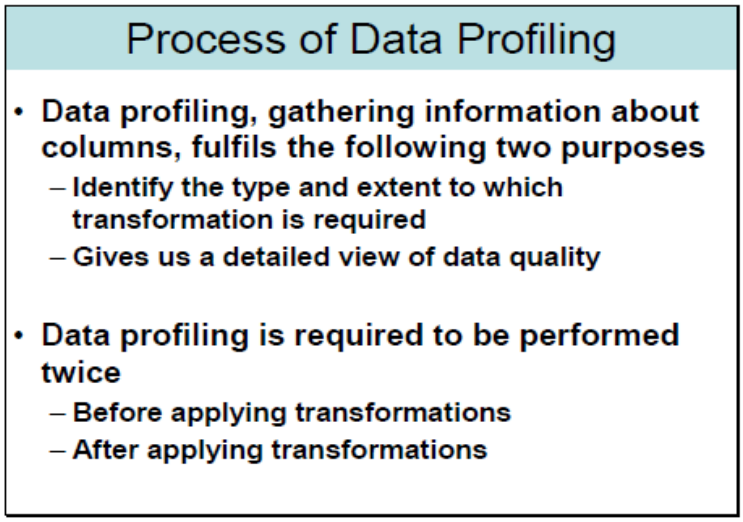
We run different SQL queries to get the answers of above questions. During this process we can identify the erroneous records. Whenever we will come across an erroneous record we will just copy it in error or exception table and set the dirty bit of record in the actual student table. Then we will correct the exception table. After this profiling process we will transform the records and load them into a new table Student_Info.

Slide 9



After transforming all records, we will perform data profiling again and identify the reduction in the number of errors. Student_Info must be clean and standardized campus table. After performing same steps for all campuses we will just put Student_Info tables from four campuses together to get consolidated source.

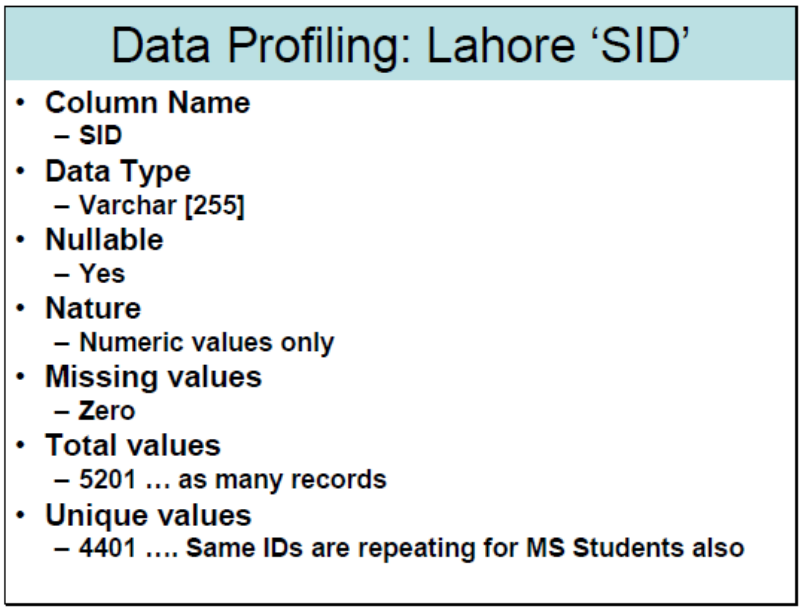
Slide 10



Data profiling is a process of gathering information about columns, It must fulfil the following purposes
• Identify the type and extent to which the transformation is required
• The number of columns which are required to be transformed and which transformation is required, meaning date format or gender convention.
• It should provide us a detailed view about the quality of data. The number of erroneous values and the number of values out of domain.

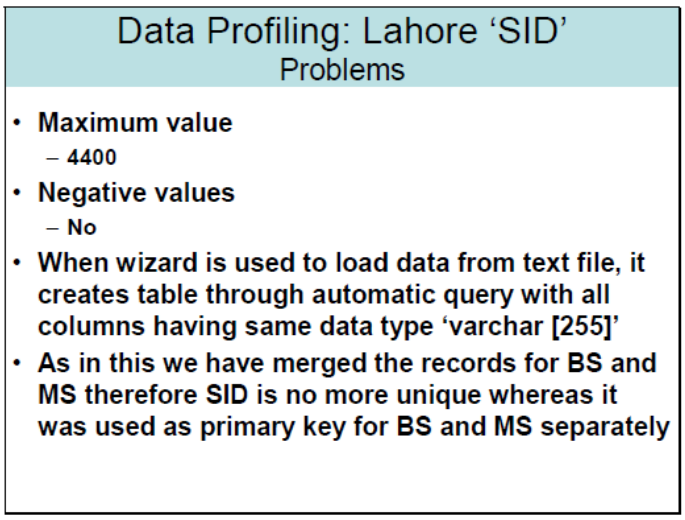
To judge effectiveness of transformation we perform data profiling twice. One before transformation and the other after transformation.

Slide 11



The slide shows data profiling for Student ID. The results can help us in identifying that what transformations should be applied. Like we can see from here that data type is highly incompatible. To store SID we should not have varchar[255], we can use varchar[10] at maximum. To know whether column is unique or not, we can compare total values with distinct / unique values. If both values are same then its mean that our column is unique. Similarly we can also identify and count the total number of null values.

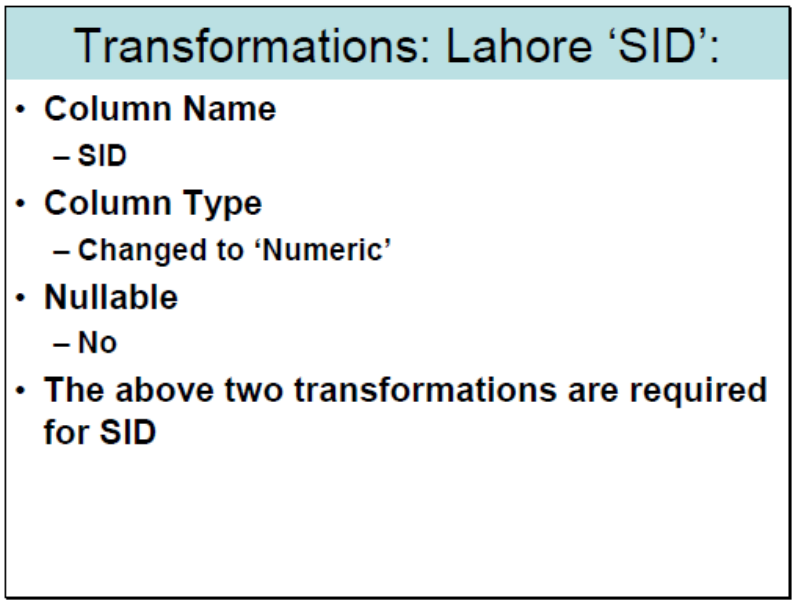
Slide 12



Two major problems that are identified are:

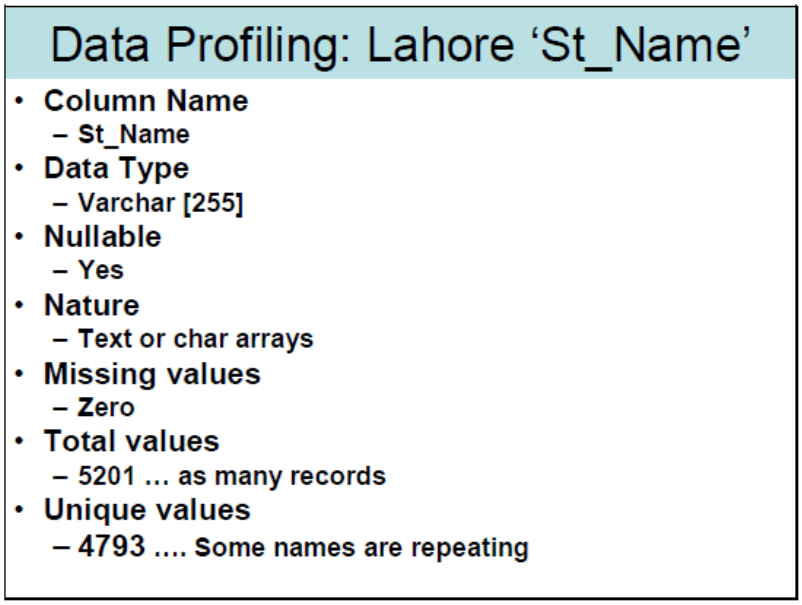
- Varchar[255] is very large to store IDs, names and even addresses, we must need to change it.
- There is no unique column that can be used as primary key

Slide 13



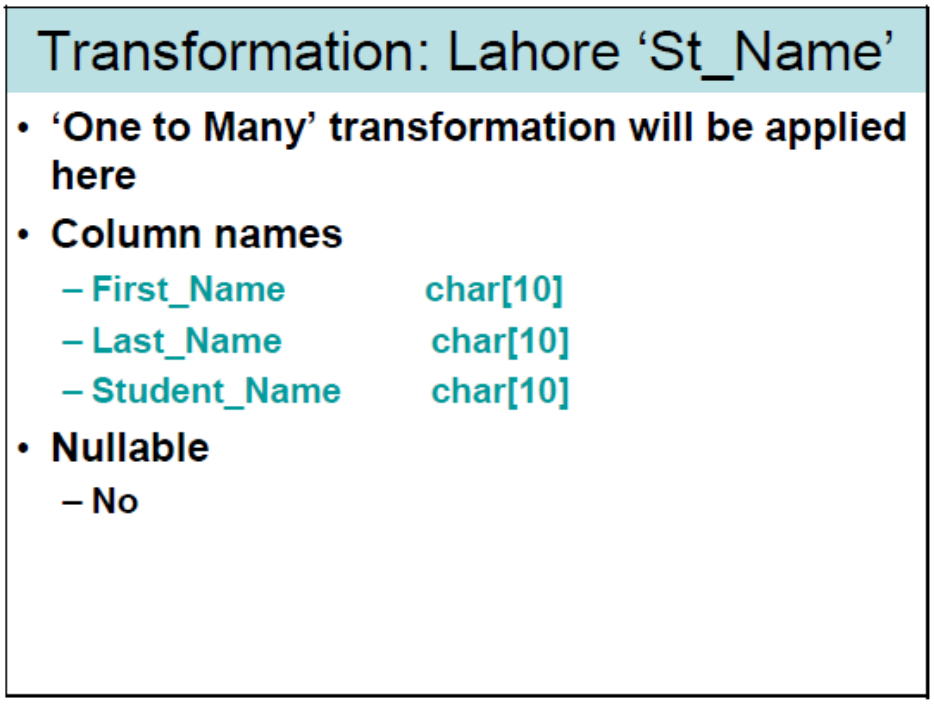
The slide shows the transformations that are suggested for the first column that is SID. First of all its type should be changed to Numeric, as varchar[255] does not make any sense. Secondly we need to change nullability option to no, as ID of each student must be present there. NULLs in ID can cause great trouble while joining tables.

Slide 14



The above slide shows the profiling output of St_Name column. The slide shows that no names are missing and some of the names are repeating.

Slide 15

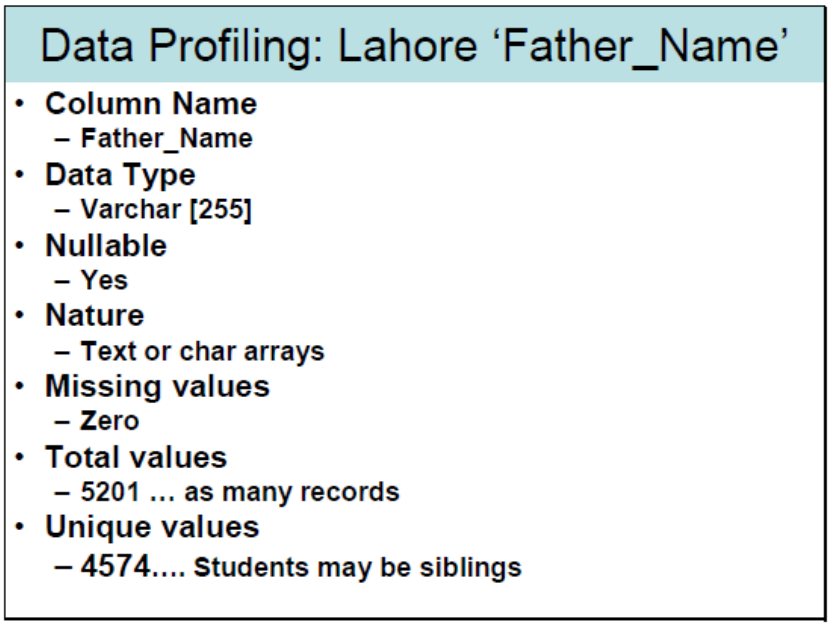


To store information we need one to many transformations of names. We need to transform name of each student into 3 columns

• First Name

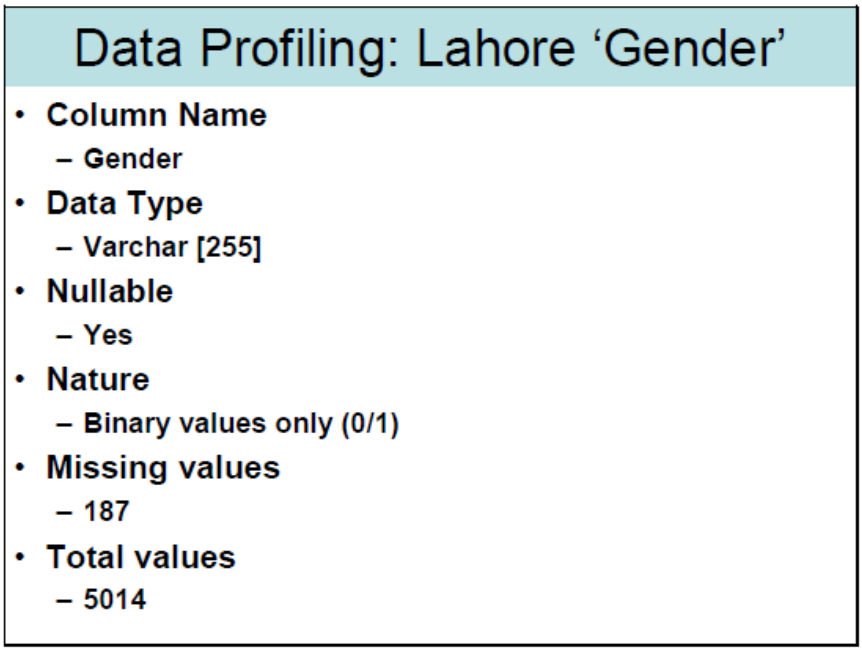
• Last Name
• Student Name (middle part of name)
This type of transformation requires scripts. We will write VB Scripts for such transformations.

Slide 16



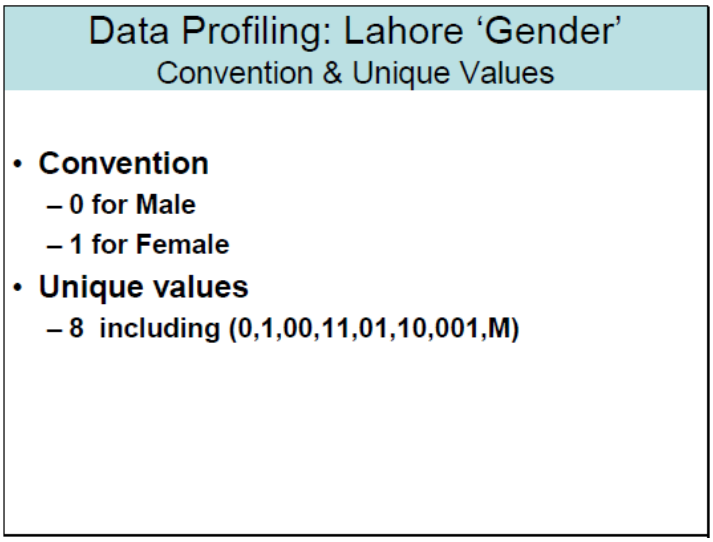
The slide shows the profiling for Father's name.

Slide 17



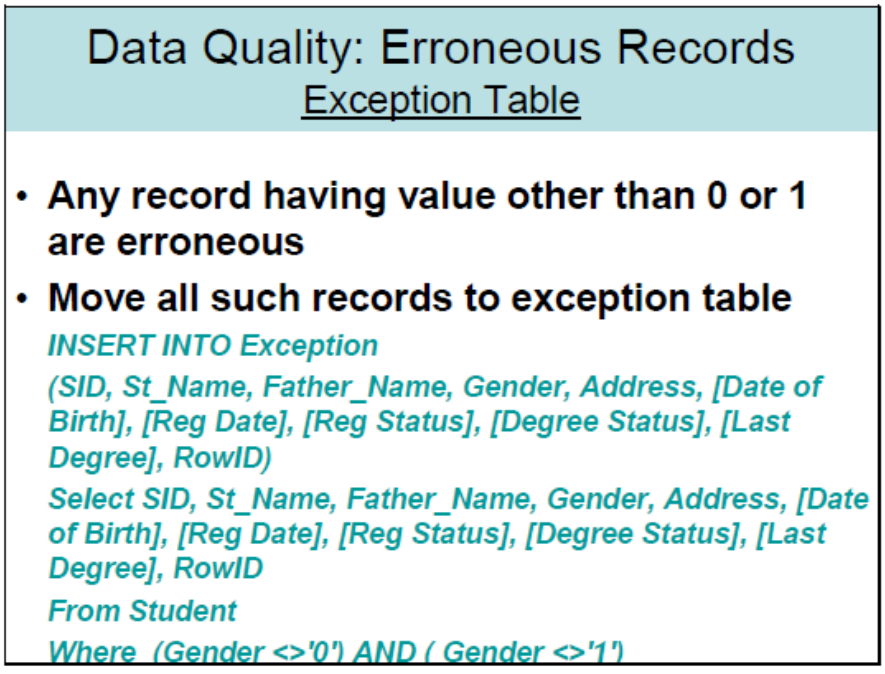
The slide shows the profiling details for column Gender.

Slide 18



There should be only two unique values in this column (0 and 1). But while profiling we have found that there are 8 unique values. Its mean there are a lot of rows corrupted in original data. Some may be typos or missing values etc. We need to clean data to make it usable for analysis purposes.

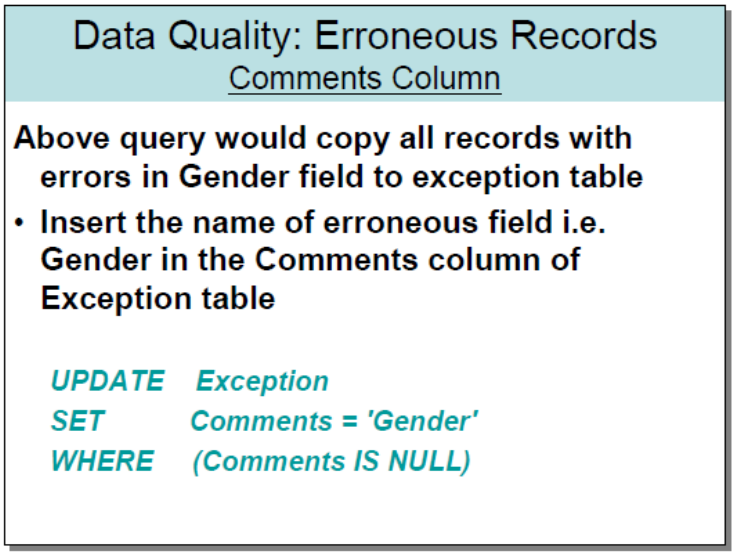
Slide 19



Legal values are just 0 and 1. All other values are noise and required to be cleaned. To clean values put all corrupted rows to exception table. Following query can be used to move all corrupted rows to exception table.

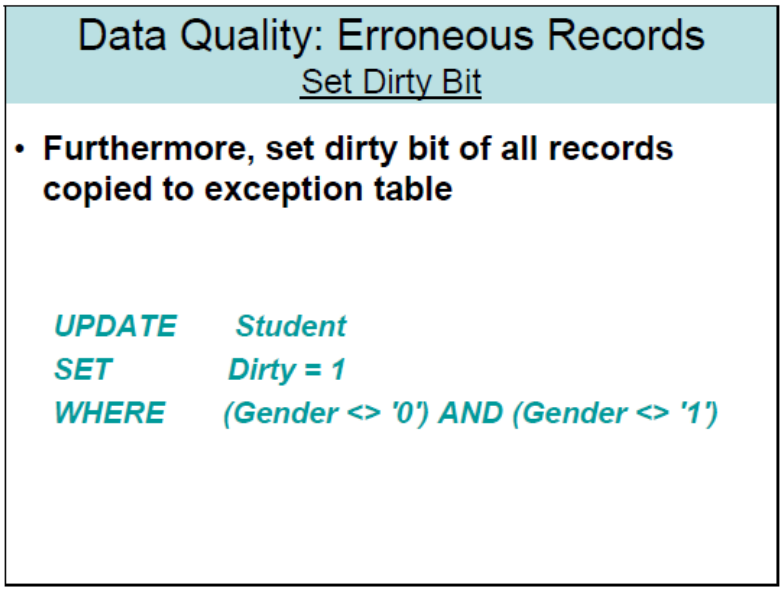
```
INSERT INTO Exception
   (SID, St_Name, Father_Name, Gender, Address, [Date of Birth], [Reg   Date], [Reg
   Status], [Degree Status], [Last Degree], RowID)
Select      SID,  St_Name,  Father_Name,  Gender,  Address,  [Date  of  Birth],
[Reg Date], [Reg Status], [Degree Status], [Last Degree], RowID
From Student
Where  (Gender <>'0') AND ( Gender <>'1')
```

Slide 20



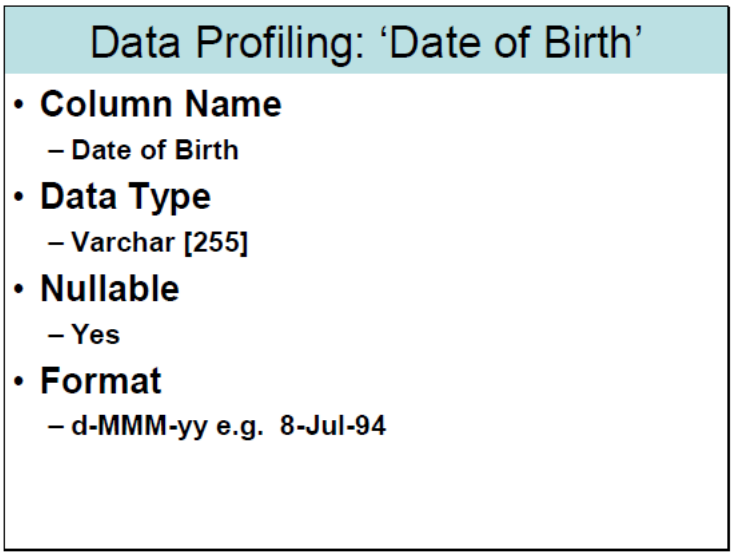
As the Comment column was added just to store the name of column having error, therefore, above query is required to be executed to update Comment column to enter the name of corrupted column against each row.
This query is required to be run right after each Insert query to exception table, otherwise we will lose the information 'What was wrong in a particular row?' in exception table.

Slide 21



At the same time we need to update each corrupted record in original student table for Lahore campus by setting its dirty bit. So that we can maintain the record which rows are copied in error table. Dirty bit says that this record has at least one field corrupted. It does not show how many fields are corrupted.

Slide 23

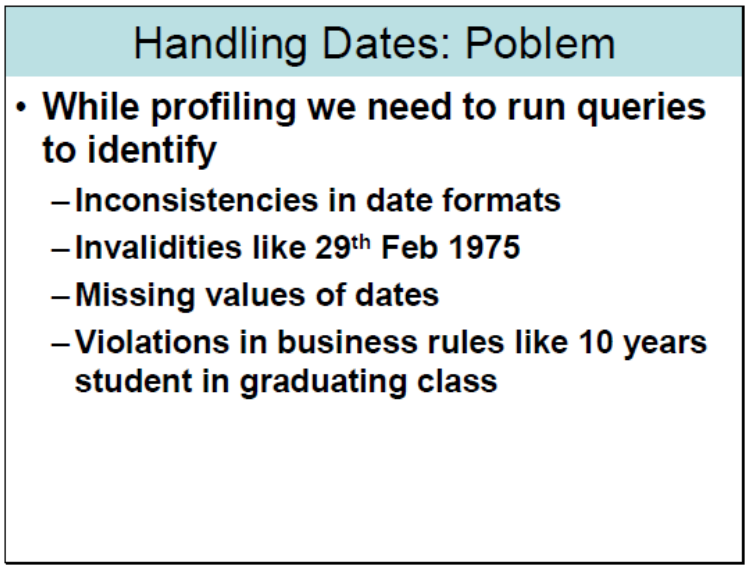


The slide shows the output of profiling of column 'Date of Birth'. Again column type is incorrect, column can contain null values, and format of date is **d-MMM-yy.**

Profiling Date needs to change our flow of work. For columns before this we just did profiling and no transformation has been done yet. We can not profile date without transforming it. So we need to transform date first and then profile. The problem is, when we loaded all records to SQL server from text files they are loaded as strings (character arrays). While profiling we may want to get the range of dates

(minimum and maximum dates). We can not identify date ranges until or unless we transform it as date data type.

Slide 24

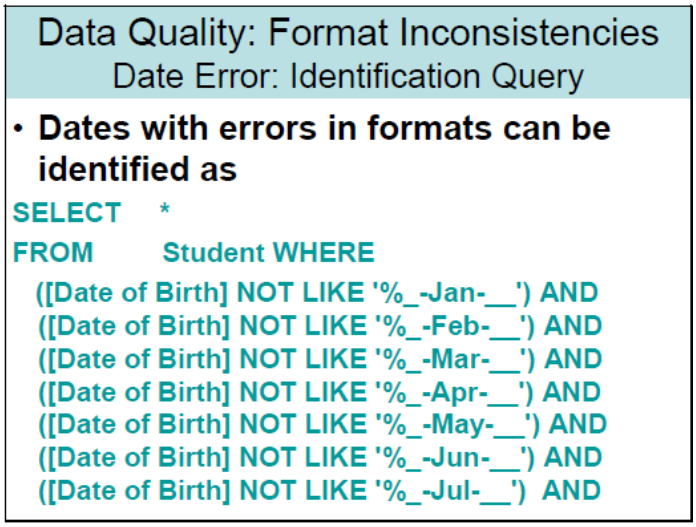


By this time you must have got the idea that data profiling is a powerful method to have an idea about the quality of data. While profiling data we need to run queries to identify:

- Inconsistencies in date formats
- Invalidities like 29th Feb 1975
- Missing values of dates
- Violations in business rules like 10 years student in graduating class

Not only the values out of domain of columns hit the quality of data but certain values in the domain of column may cause the quality to suffer. Like the values cause the violation of business rules. While profiling data we must have a set of rules regarding to our business like we can not allow any student less than 16 years to be admitted for BS. While profiling we need to identify the records violating such rules means if we find a 10 years old student in graduated students then its the violation of business rule and there must be some error either in date of birth of student or date of graduation. Similarly any student has date of graduation less than date of birth then again it is considered as noise while both dates are in valid domain. Such dates are required to be identified at the time of profiling and needs to be dealt separately in error table.

Slide 25

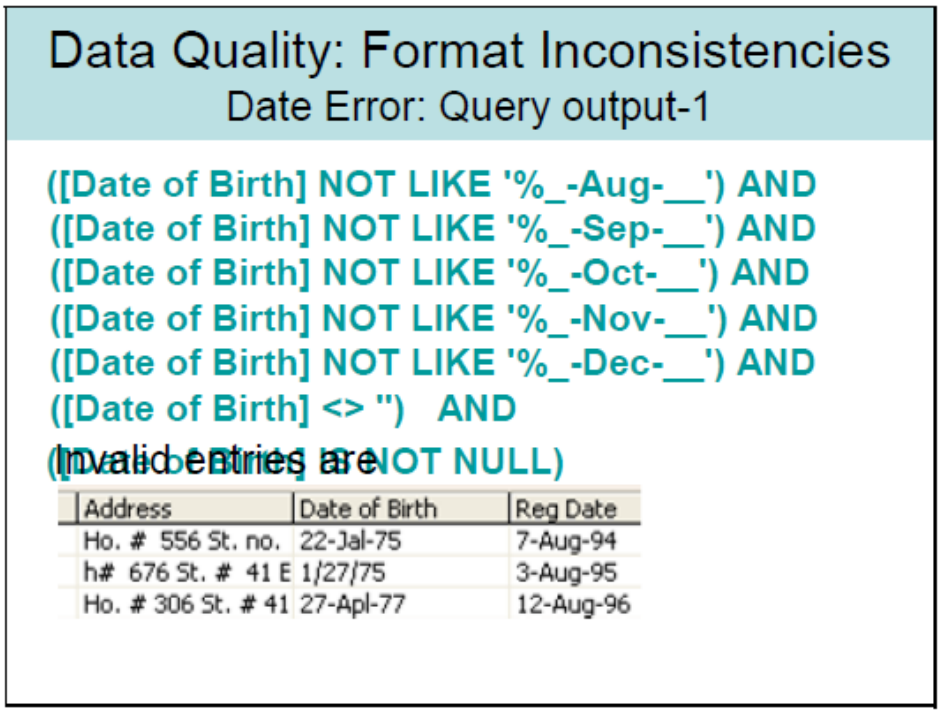


Errors in the format of dates can be identified through following query.

```
SELECT    *
FROM      Student
WHERE     ([Date of Birth] NOT LIKE '%_-Jan-__')
AND       ([Date of Birth] NOT LIKE '%_-Feb-__')
AND       ([Date of Birth] NOT LIKE '%_-Mar-__')
AND       ([Date of Birth] NOT LIKE '%_-Apr-__')
AND       ([Date of Birth] NOT LIKE '%_-May-__')
AND       ([Date of Birth] NOT LIKE '%_-Jun-__')
AND       ([Date of Birth] NOT LIKE '%_-Jul-__')
AND       ([Date of Birth] NOT LIKE '%_-Aug-__')
AND       ([Date of Birth] NOT LIKE '%_-Sep-__')
AND       ([Date of Birth] NOT LIKE '%_-Oct-__')
AND       ([Date of Birth] NOT LIKE '%_-Nov-__')
AND       ([Date of Birth] NOT LIKE '%_-Dec-__')
AND       ([Date of Birth] <> '')  AND      ([Date of Birth] IS NOT NULL)
```

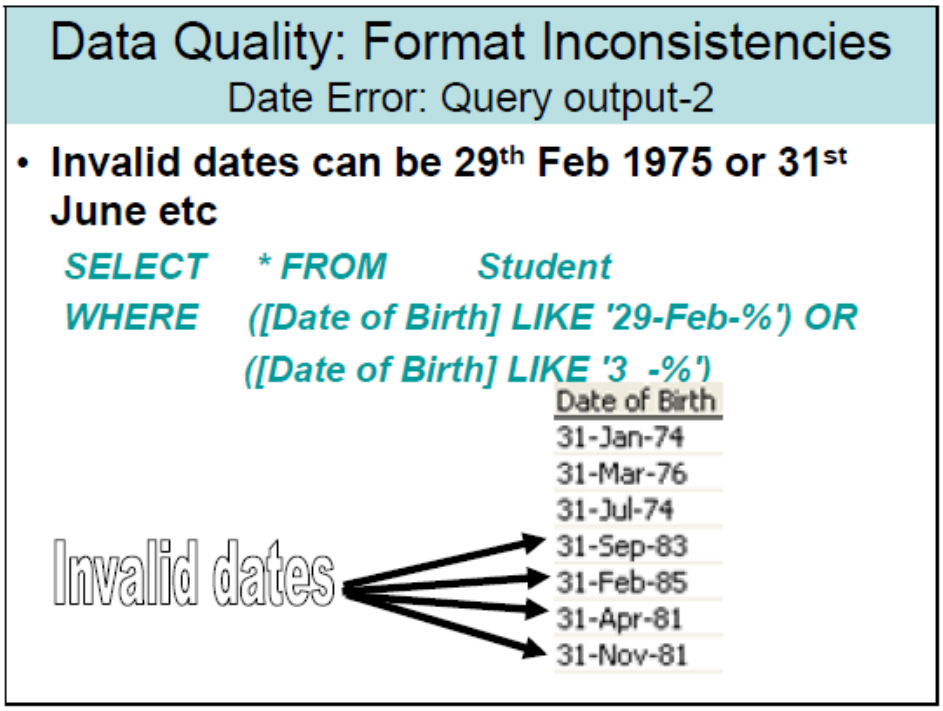
The above query can not identify all invalid dates like 30-Feb-2005. But any date with invalid format, or having illegal month can be identified through the above query.

Slide 26



The output of the above query shows 3 invalid dates. 22-Jal-75,1/27/75,27-Apl-77. These are invalid dates as their format are not correct. While transforming dates from string to date format, all dates must be legal. If any illegal date comes the package will be terminated and all transactions will be rolled back. So through sequence of different query we try to identify all invalid dates so that our package can complete its execution successfully.

Slide 27

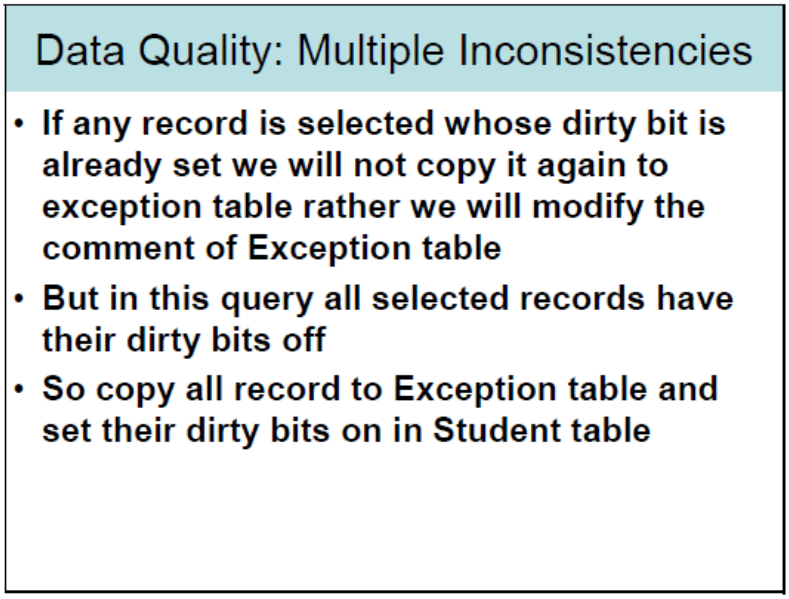


To identify invalid dates like 29th Feb 1975 or 31st June, we can run the following query

```
SELECT    * FROM      Student
WHERE     ([Date of Birth] LIKE '29-Feb-%') OR
          ([Date of Birth] LIKE '3_-%')
```

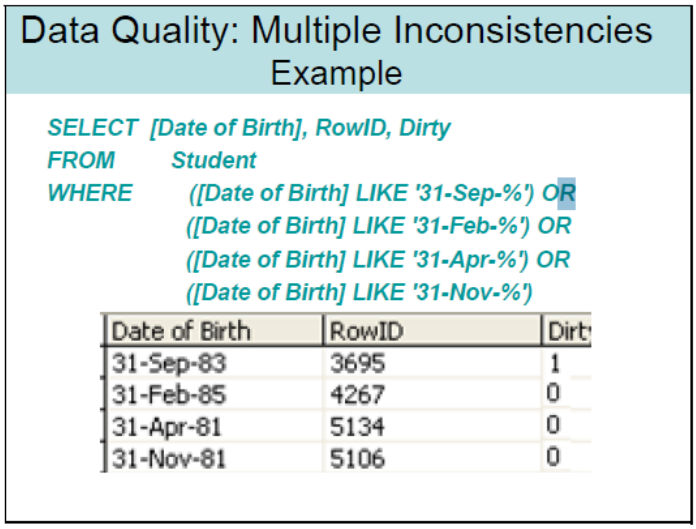
There is no finalized methodology to identify errors in dates. It is up to the designer what sequence of queries he/she generates to identify the errors in dates because the purpose is to identify all erroneous dates before running the package. Otherwise, package's execution will be terminated and all transactions caused due to that package will be rolled back.

Slide 28



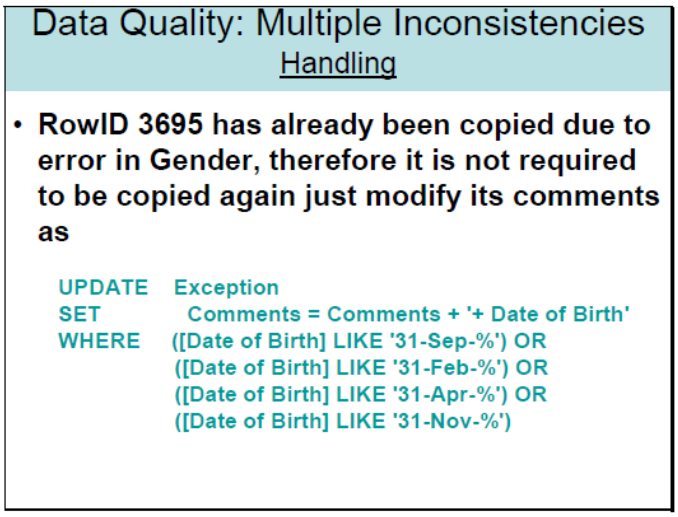
It has been discussed earlier, how we will work with error table. Now if we meet a record whose dirty bit is already set then we will not copy the record again but we will modify the comment in exception table. We will append the name of second erroneous column with the name of column already present there.

Slide 29



We can use following query as well. The above query and the query in the previous slide both are aimed at identify the invalidities in dates. There may be some other set of queries that can be used with the same intention to identify invalid dates. Good queries are those that identify and output only erroneous records and all erroneous records.

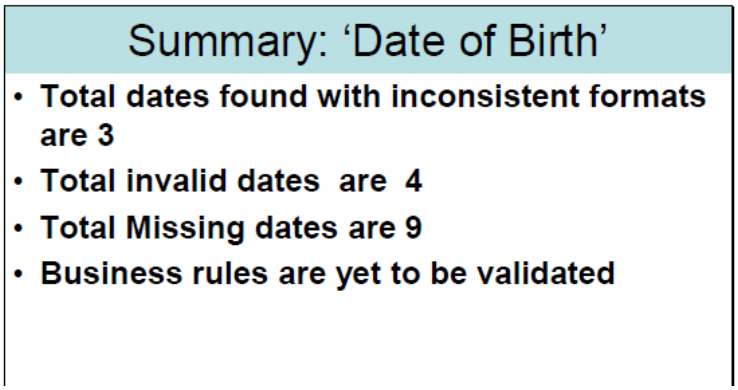
Slide 30



For the records whose dirty bit is already on, we do not need to enter the record again but to modify the comment only through following query.

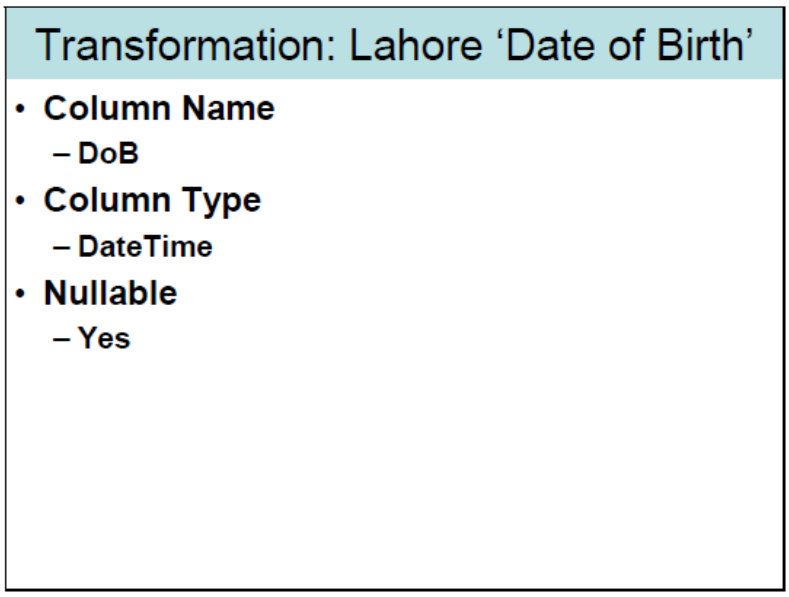
```
UPDATE    Exception
SET       Comments = Comments + '+ Date of Birth'
WHERE     ([Date of Birth] LIKE '31-Sep-%') OR
          ([Date of Birth] LIKE '31-Feb-%') OR
          ([Date of Birth] LIKE '31-Apr-%') OR
          ([Date of Birth] LIKE '31-Nov-%')
```

Slide 31



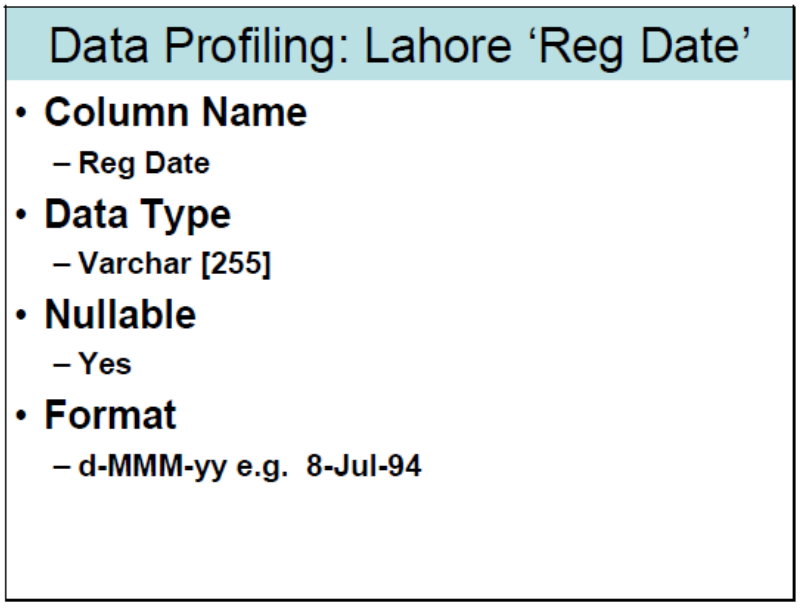
Slide shows the summary of data profiling of 'Date of Birth'. Business rule can be as
At the time of joining for BS, student must be at least 16 years old
At the time of joining for MS, student must be at least 20 years old
Business rules are yet to be validated because to validate business rule we have to have date of registration as well.

Slide 32



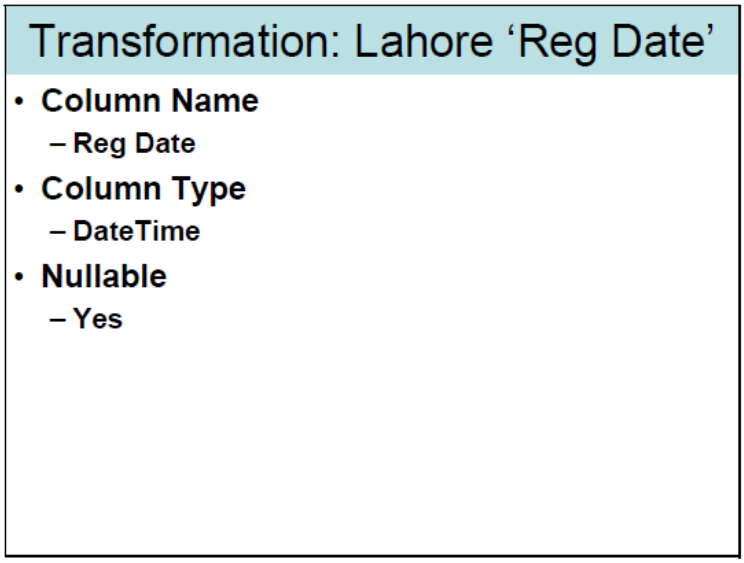
Following transformations are suggested for Date of Birth:
• Change the name of column as DoB
• Change column type as DateTime format
• Allow null values as there are some nulls and empty strings in the original data

Slide 33



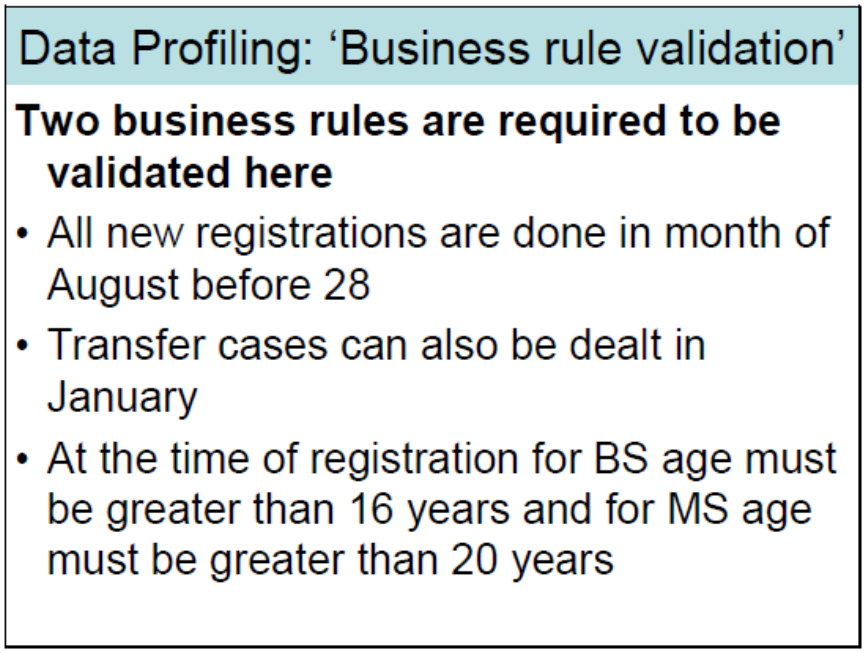
Now we need to profile date of registration column. The same procedure that we have done for Date of Birth is required to be repeated.

Slide 34



The slide shows the transformations that are required for Registration date.

Slide 35



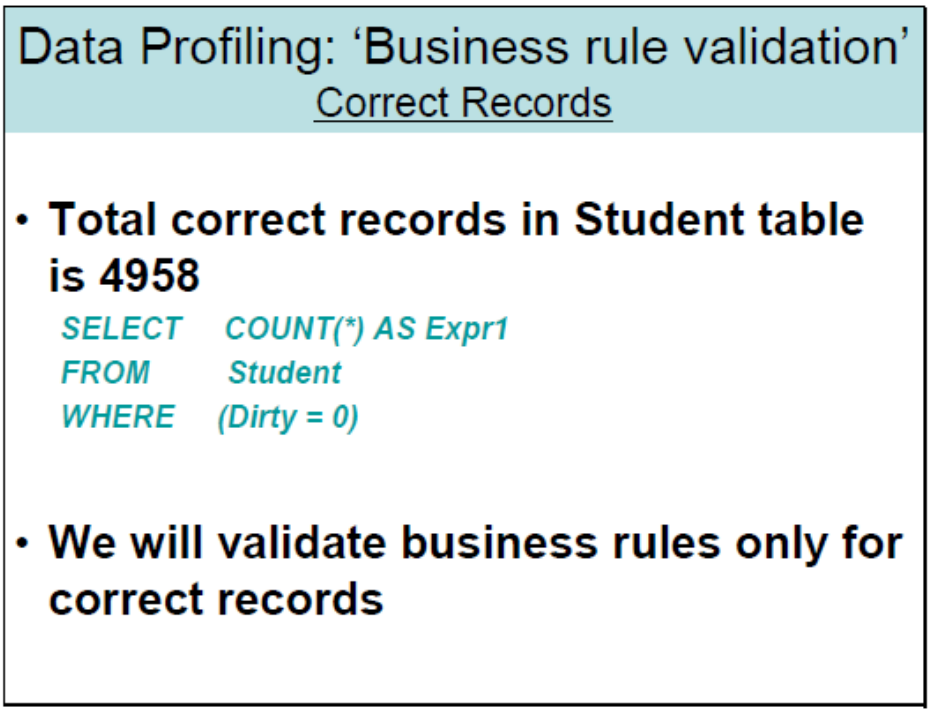
Now as we have profiled date of registration as well therefore we can validate the following business rules

• All new registrations are done in month of August before 28th
• Transfer cases can also be dealt in January

• At the time of registration for BS age must be greater than 16 years and for MS age must be greater than 20 years

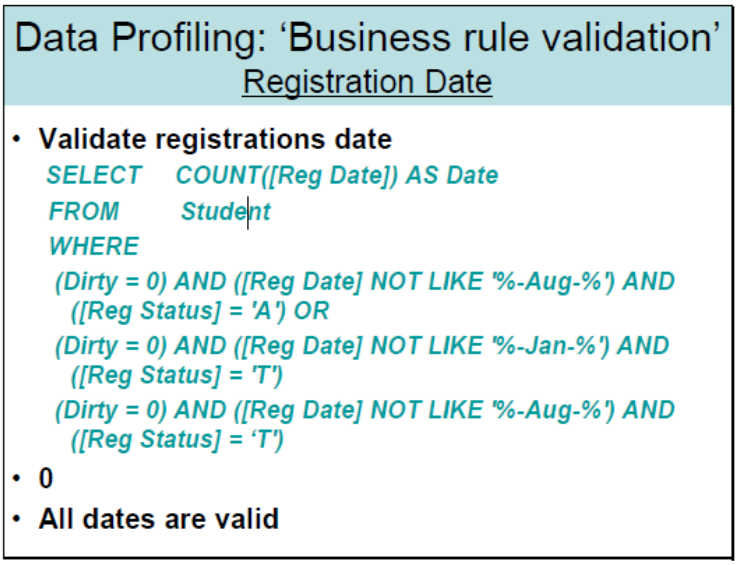
There can be a lot of business rules that are supposed to be validated at this stage. It totally depends upon the business. Other than business rules we also validate some logical rules at this stage, like date of registration can not be less than date of birth. No one could register for a degree before birth. Similarly date of birth can not be 100 years back and so on. Such rules can be devised keeping in mind the nature of business.

Slide 36



At this stage we have only correct records in Student table. All records with errors in dates have been moved to exception table. Now we will validate the correct records for given set of business rules and general logical rules. Records violating any of the rules are required to copy in exception table.

Slide 37



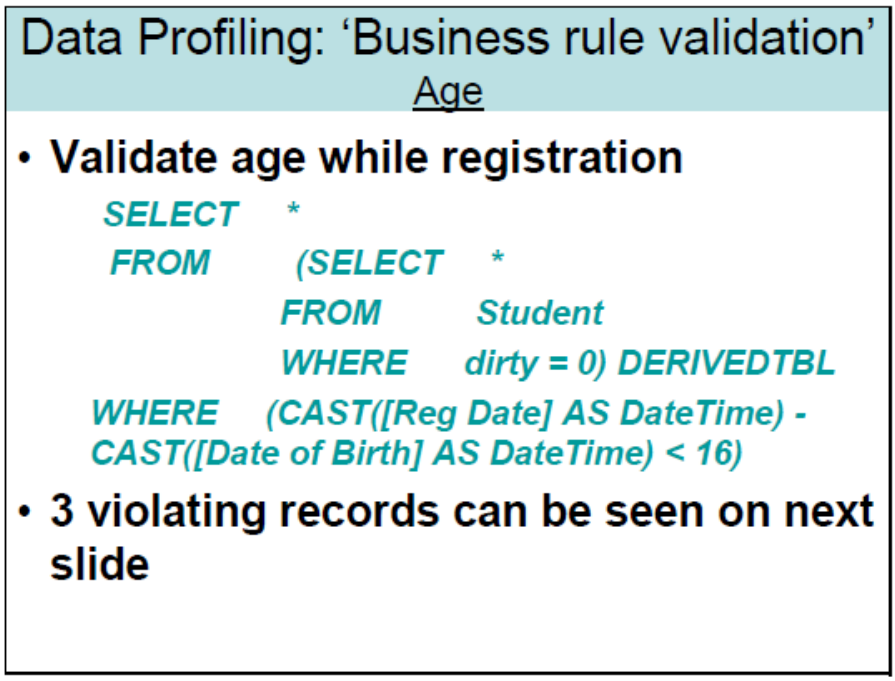
The slide shows the query that can violate the following two rules:

- **All new registrations are done in month of August before 28th**
- **Transfer cases can also be dealt in January**

We can see that the first check ,'dirty bit = 0', is giving us the records having no errors. The following query can identify all records that have registration date in any month other than August and January. Those records will be erroneous records. Records having registration date in January and Registration status is transfer case is legal. Similarly in August both Transfer cases and new admissions are legal.

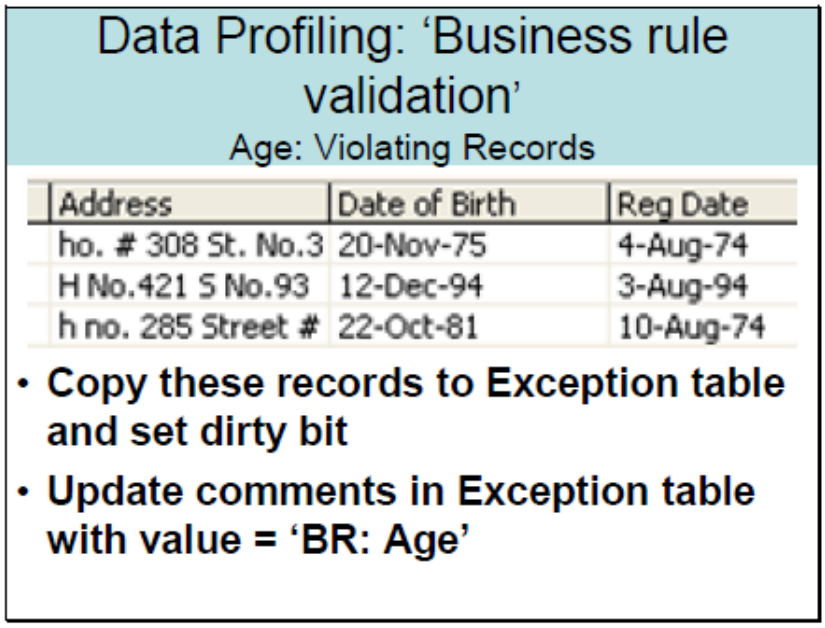
```
SELECT COUNT([Reg Date]) AS Date
FROM Student
WHERE
        (Dirty = 0) AND ([Reg Date] NOT LIKE '%-Aug-%') AND ([Reg Status] = 'A') OR
    (Dirty = 0) AND ([Reg Date] NOT LIKE '%-Jan-%') AND ([Reg Status] = 'T')
        (Dirty = 0) AND ([Reg Date] NOT LIKE '%-Aug-%') AND ([Reg Status] = 'T')
```

Slide38



The slide shows the query that can be used to validate the following business rule:
At the time of registration for BS age must be greater than 16 years and for MS age must be greater than 20 years
The query again first identifies the correct records through dirty bit. Then checks which of the students are less than 16 years at the time of registration. To apply '+' or '-' operator on dates we need to cast date strings into date data type by using CAST function. If there comes any invalid date the query terminates with error.

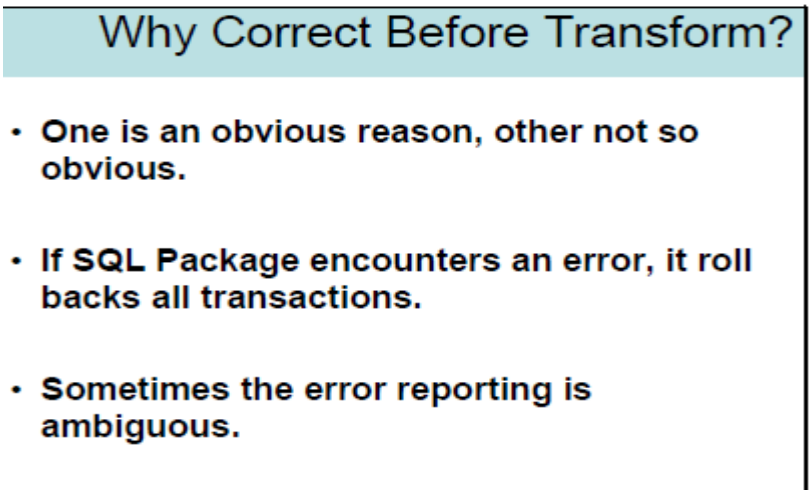
Slide 39



On the slide we can see three records that are violating business rules. In the first record date of birth is 20-Nov-75 and date of Registration is 4-Aug-74, which is obviously impossible. Here, one thing is obvious that date of registration is incorrect as we have data starting from 1994 only. Either date of birth is correct or not, nothing can be said without consulting golden copy (Original copy). We will copy these records to exception table and set the dirty bit for these records. For such cases we will update comment by appending 'BR:Age' (Business rule: Age).

Lab 16 Part III

Data Transformation & Standardization



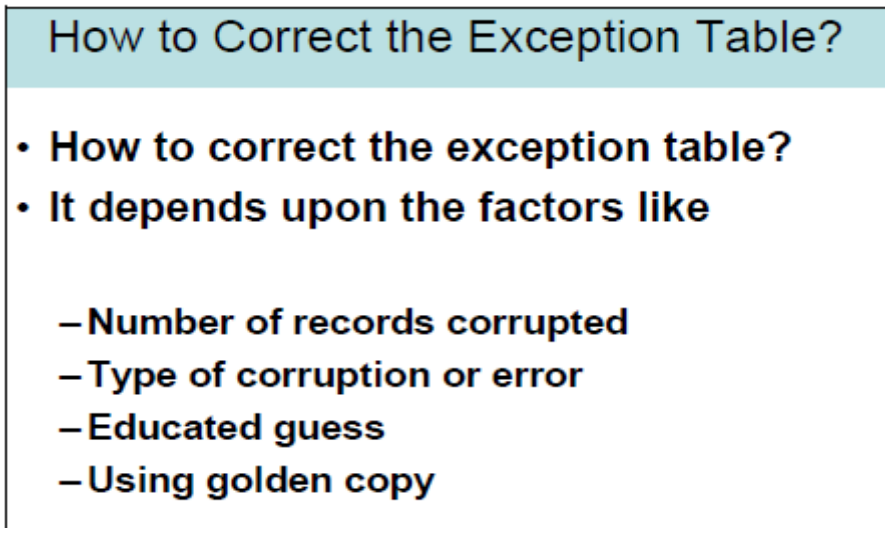
After completion of data profiling for Lahore campus data, we are required to correct the records in the error table. After corrections the corresponding records in the actual table are required to be updated. The methodology to correct the exception table is dependent on the following factors:

Number of records corrupted

If the number of corrupted records are very large in number then we need to update it through SQL Queries or some other automated way, otherwise, if they are less in number then they can be updated through inspection or manual checking.

Type of corruption or error

If the dates are missing we must need to consult golden copy. If gender is missing we are not required to consult golden copy. In many cases name can help us in identifying the gender of the person.



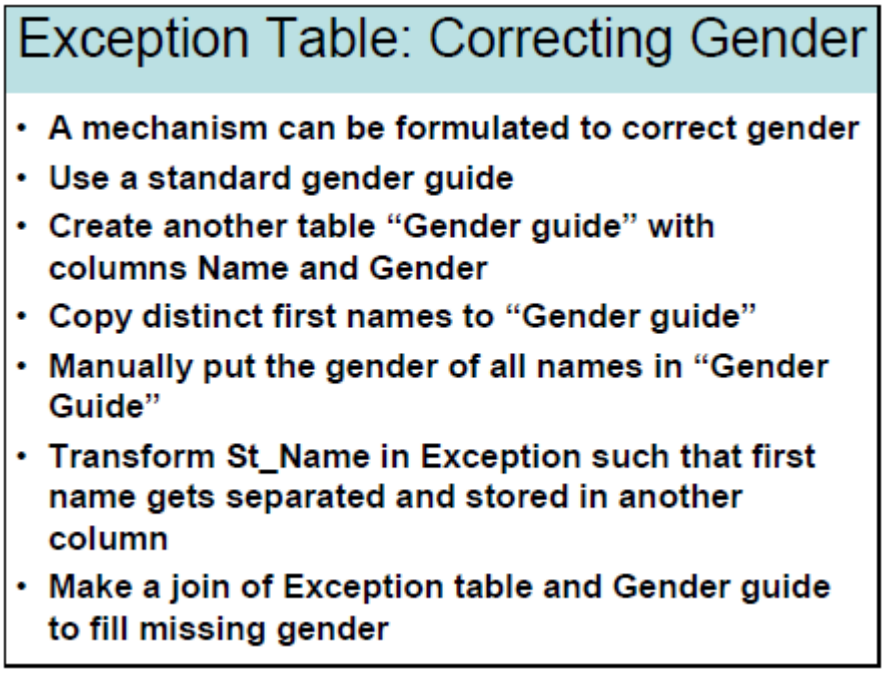
After completion of data profiling for Lahore campus data, we are required to correct the records in the error table. After corrections the corresponding records in the actual table are required to be updated. The methodology to correct the exception table is dependent on the following factors:

• **Number of records corrupted**

If the number of corrupted records are very large in number then we need to update it through SQL Queries or some other automated way, otherwise, if they are less in number then they can be updated through inspection or manual checking.

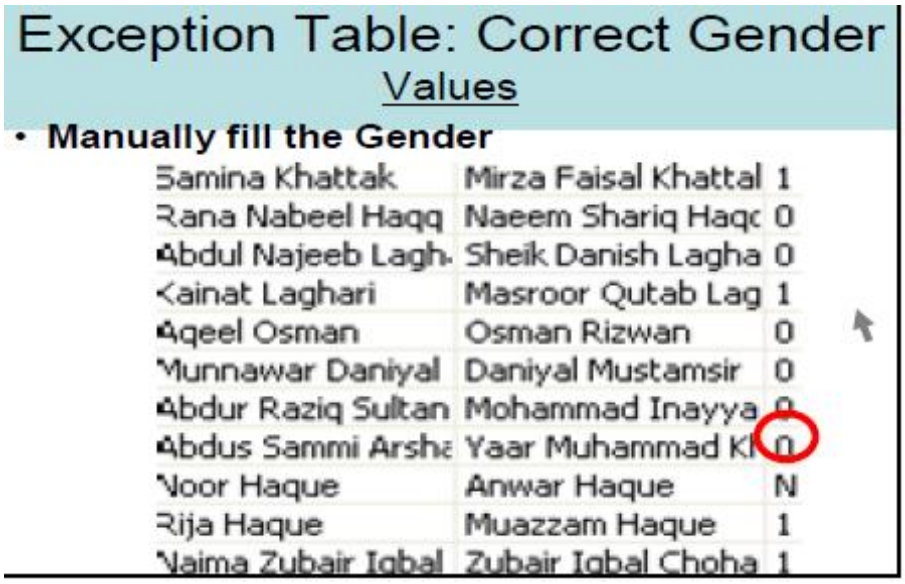
• **Type of corruption or error**

If the dates are missing we must need to consult golden copy. If gender is missing we are not required to consult golden copy. Name can help us in identifying the gender of the person.

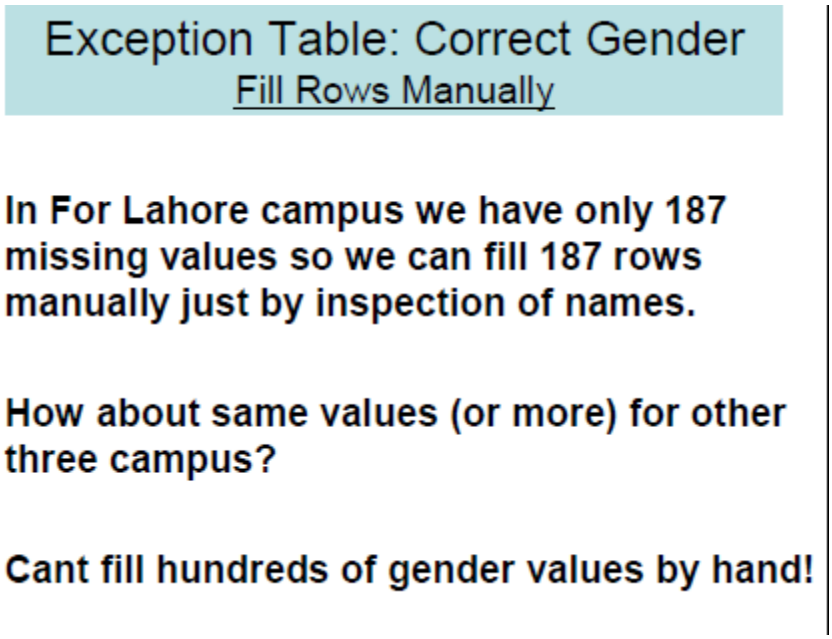


If for very large number of records gender is missing, it would become impossible for us to manually check each and every individual's name and identify the gender. In such cases we can formulate a mechanism to correct gender. We can either use a standard gender guide or create a new table Gender_guide. Gender_guide contains only two columns name and gender. Populate Gender_guide table by a query for selecting all distinct first names from student table. Then manually placing their gender.

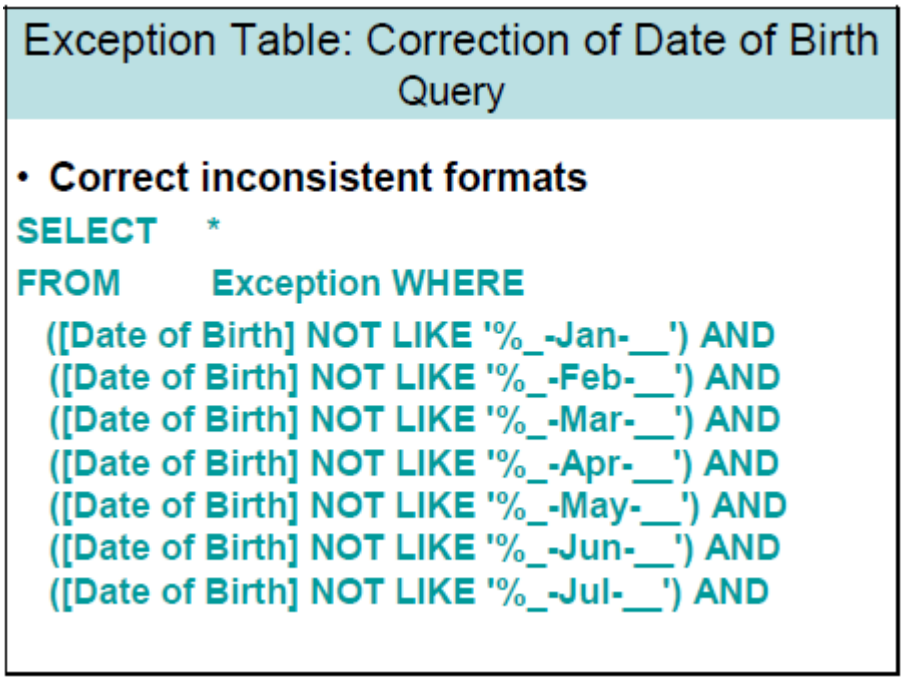
This table can serve us as guide by telling what can be the gender against this particular name. For example if we have hundred students in our database with first name equal to
'Muhammed'. Then in our Gender_guide table we will have just one entry 'Muhammed' and we will manually set the gender as 'Male' against 'Muhammed'. Now to fill missing genders in exception table we will just do a inner join on Error table and Gender_guide table. We will get the gender against matching names.



The slide shows an interesting case. We can update gender easily against names like Sara that clearly identifies that the student is female. But there may be certain names that are common for males and females like Shamim, Khursheed, etc. In the slide we can see a name Noor Haque, doesn't conveying the gender. It may be male and female as well. So, for such case at this stage we can use 'N', but these cases can only be resolved through consulting golden copy. This conversion i.e. N needs to be reflected in Meta data.



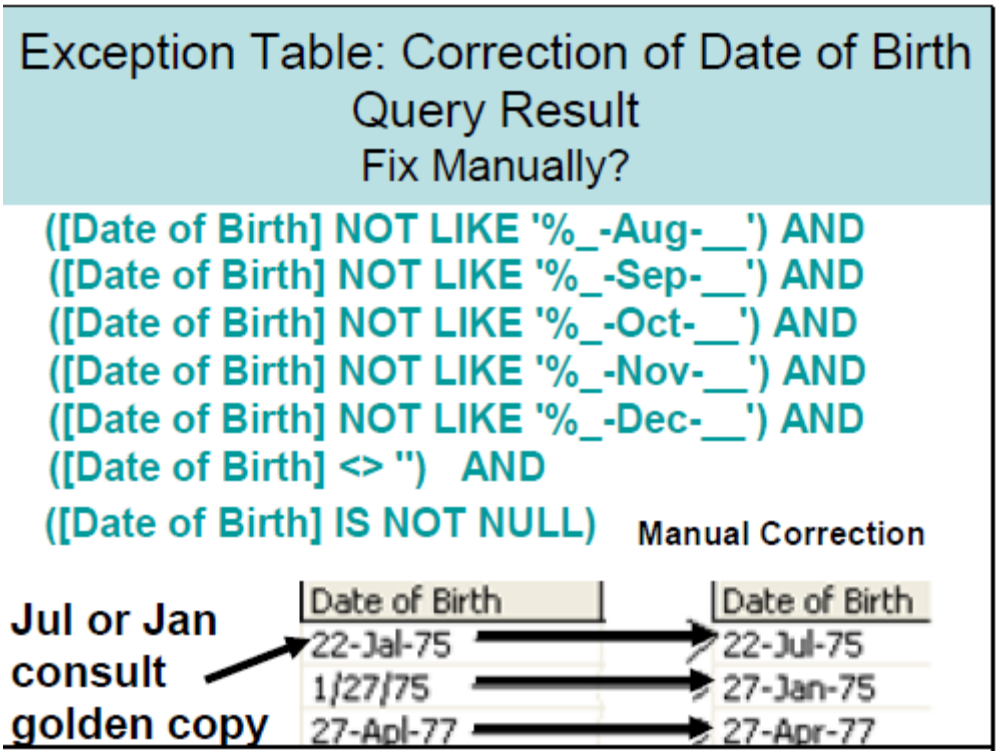
In this particular case of Lahore campus we have only 187 missing values so we can fill 187 rows manually just by inspection of names.



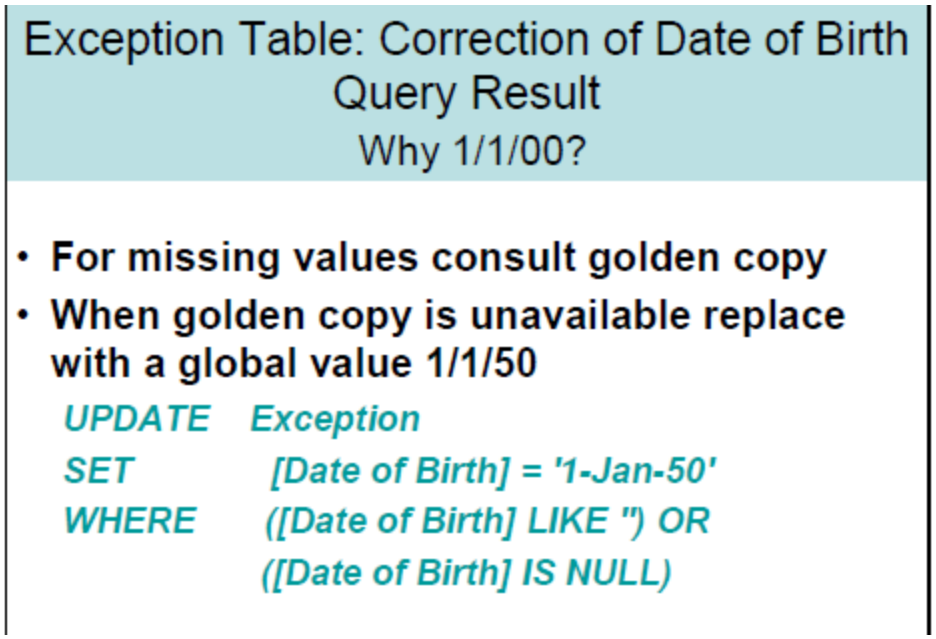
Now we need to correct the inconsistencies in date formats. First select all dates with inconsistent formats. The following query can do this for us

SELECT *
FROM Exception WHERE

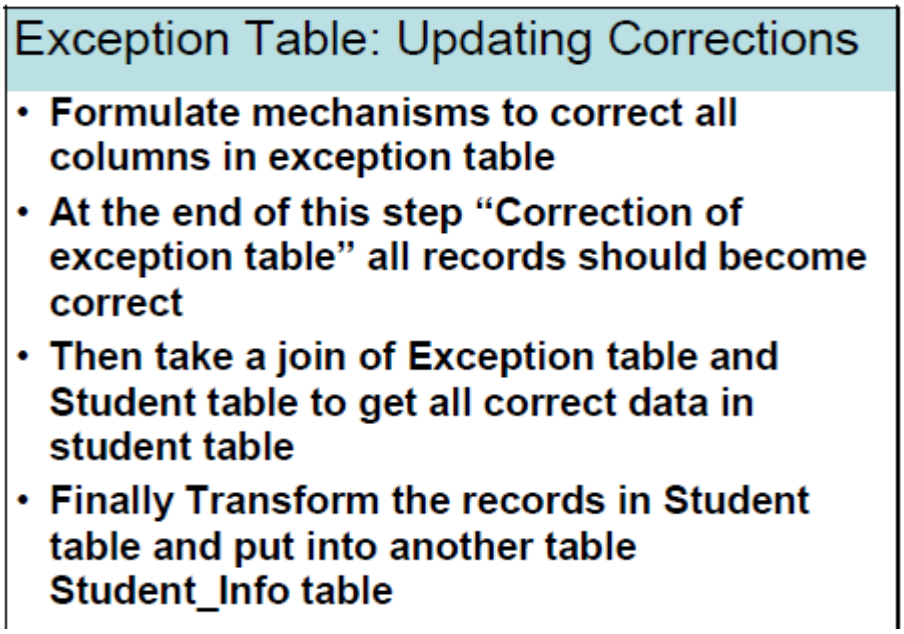
([Date of Birth] NOT LIKE '%_-Jan-__') AND
([Date of Birth] NOT LIKE '%_-Feb-__') AND
([Date of Birth] NOT LIKE '%_-Mar-__') AND
([Date of Birth] NOT LIKE '%_-Apr-__') AND
([Date of Birth] NOT LIKE '%_-May-__') AND
([Date of Birth] NOT LIKE '%_-Jun-__') AND
([Date of Birth] NOT LIKE '%_-Jul-__') AND
([Date of Birth] NOT LIKE '%_-Aug-__') AND
([Date of Birth] NOT LIKE '%_-Sep-__') AND
([Date of Birth] NOT LIKE '%_-Oct-__') AND
([Date of Birth] NOT LIKE '%_-Nov-__') AND
([Date of Birth] NOT LIKE '%_-Dec-__') AND
([Date of Birth] <> '') AND
([Date of Birth] IS NOT NULL)



The slide shows the results of the above query. We can see that first incorrect date is 22- Jal-75. This error cannot be corrected until and unless we do not consult golden copy. Remaining two errors can be corrected without consulting golden copy.

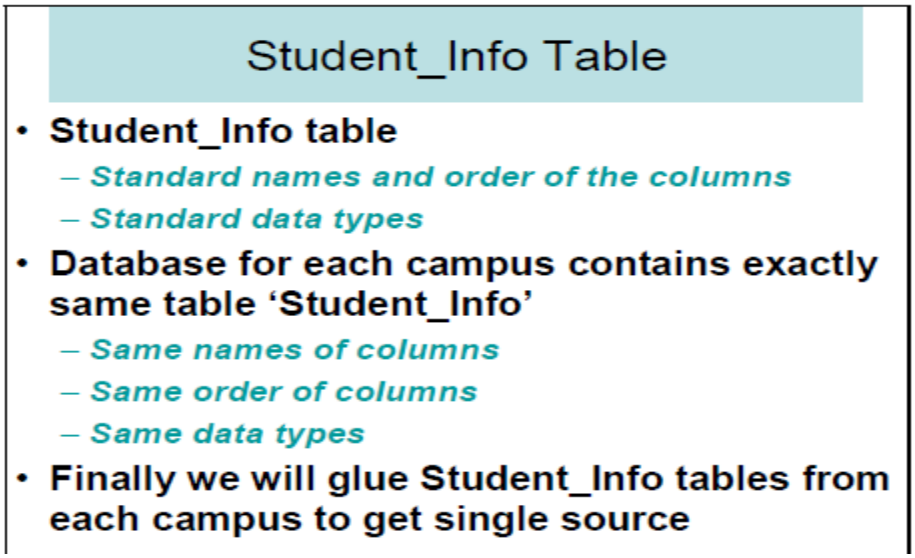


There are some nulls and empty strings in Date of Birth. We cannot put in values without consulting golden copy. So for the time being we can use a standard value like 1/1/1900 or 1/1/1950 etc., again a Meta data entry. Later on these values can be replaced with original values after consulting golden copy.



Its up to the designer what methodology he/she designs to correct the exception table. The goal is to correct all records in exception table up to a certain level. Hundred percent correction is not possible to achieve because of errors in golden copy. After correction of exception table we need to take a join of exception table and student table so that all records of student table can be updated with the values in exception table. After this exercise student table should be clean up to a variable level.

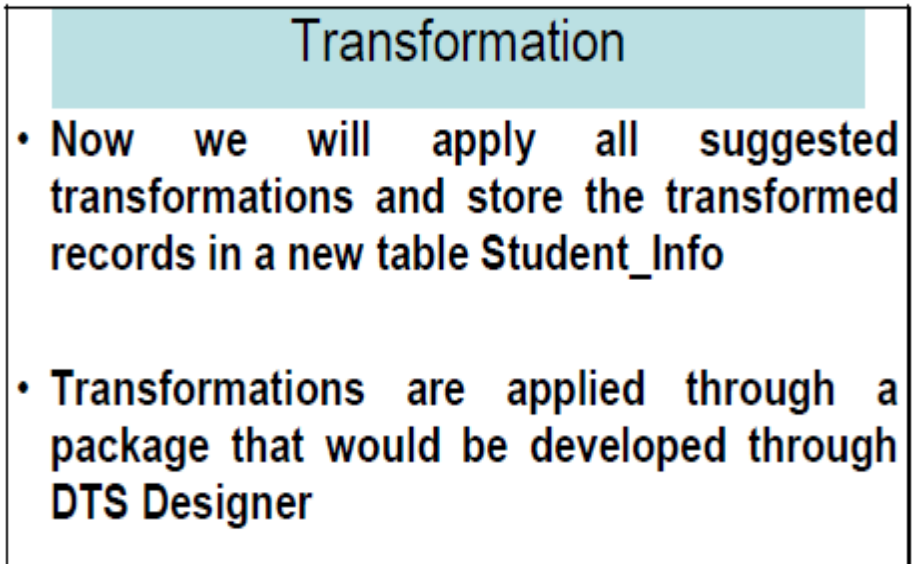
After the cleansing of all records it is the time to transform records and put them into another table named as 'Student_Info'.



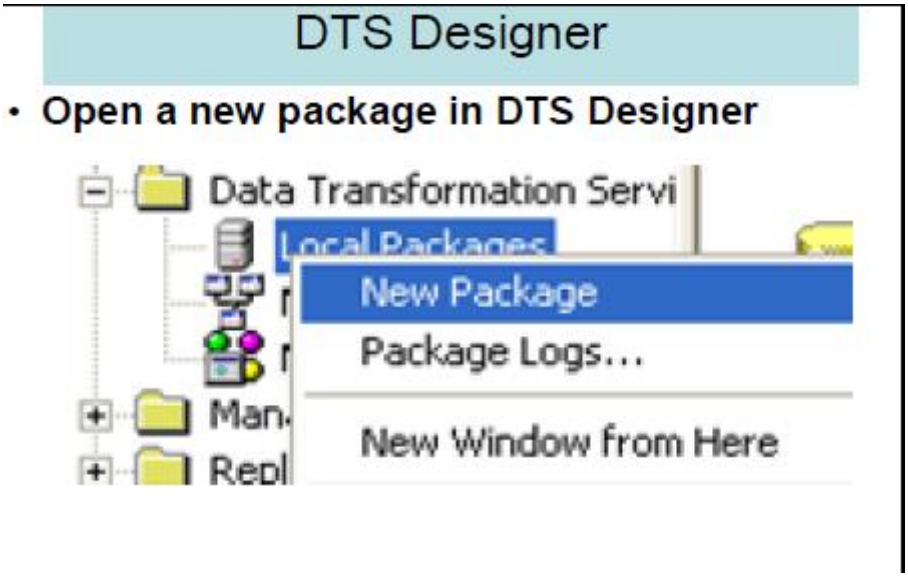
After correcting student table we need another table to store all records after applying transformations. To serve this purpose we create another table Student_Info. The most important thing about this table is that it contains the names and data types of columns that are suggested in data profiling. For example, data type for dates columns (birth date and registration date) is DateTime, therefore, in Student_Info table we set data types for these columns dateTime at the time of creation. Same is the case with all other columns.

Student_Info table does not contain any column against row_id as this column was just added for cleansing purposes. Another important point is that Student_Info table may contain more or less columns as compared to Student table. Like in Peshawar campus we do not have the column Gender but Student_Info table for Peshawar campus does contain it. This column can be filled by joining with the table Gender_guide we created earlier to find the missing genders.

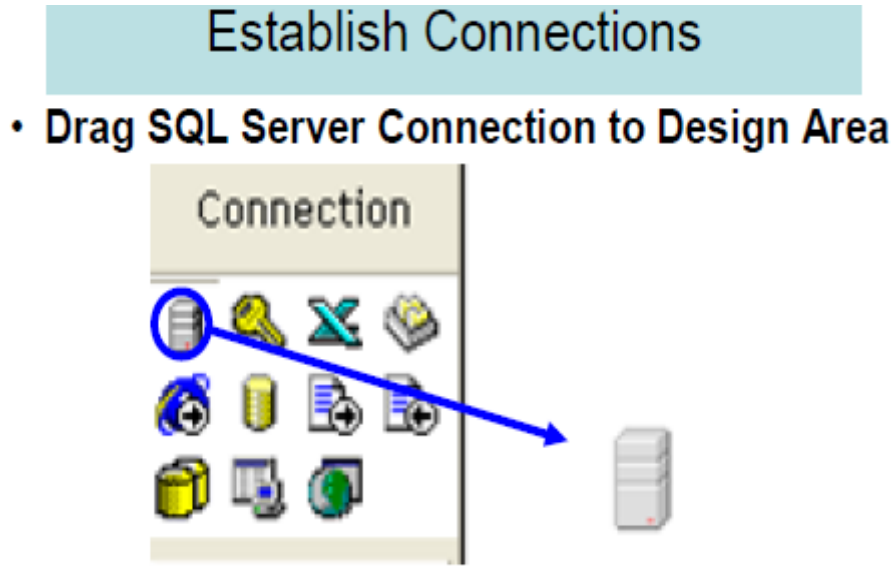
Same query of create table is used to create table Student_Info in the databases of each campus, so that exactly the same table can be created in all databases, with same names and same order of columns. After transformations and completion of Student_Info table we will just glue four tables (student_Info table, one from each campus) to get a single standardized table. As we know that in each campus order of columns in student table is different whereas order of columns in Student_Info table across each campus is same. Its mean with in a database, like Islamabad_Campus, order of columns in both tables Student and Student_Info differ. We need to apply 'copy column transformation' so that this different order of columns would not create any problem at the time of loading Student_Info table.



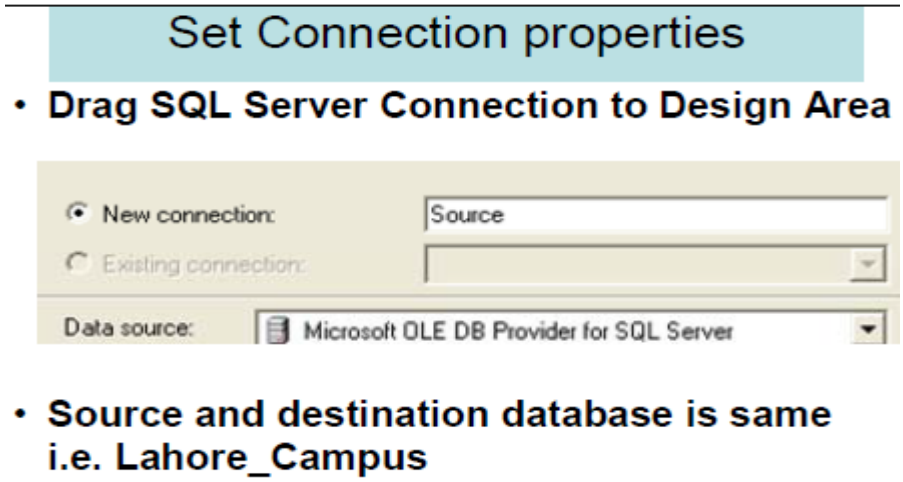
While data profiling we have suggested certain transformations on each column. Now it is the time to apply all transformations on Student table and finally put all transformed records to student_info table. To apply transformations we need to develop a package though DTS designer because wizard cannot provide us enough functionality to design a package with complex transformations.



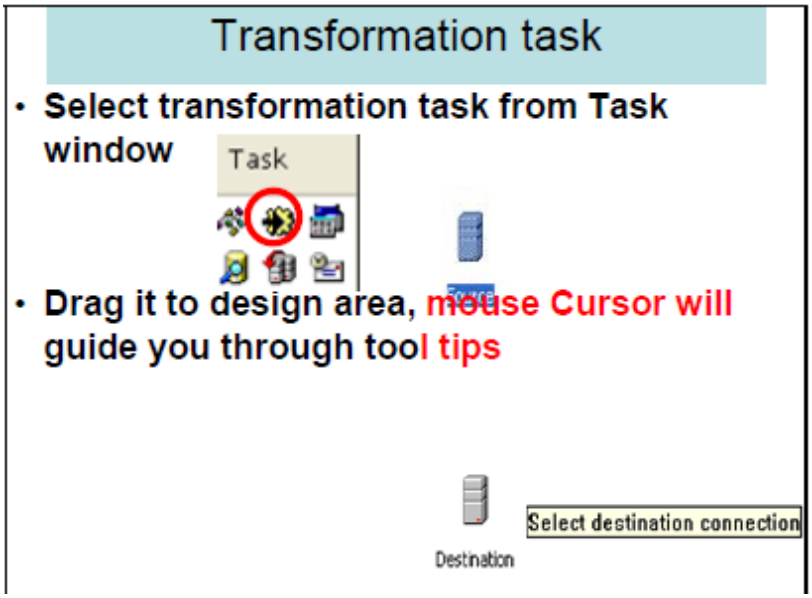
To open a new package in DTS Designer. Right click the local packages and select 'New Package'. As a result DTS Designer interface would open.



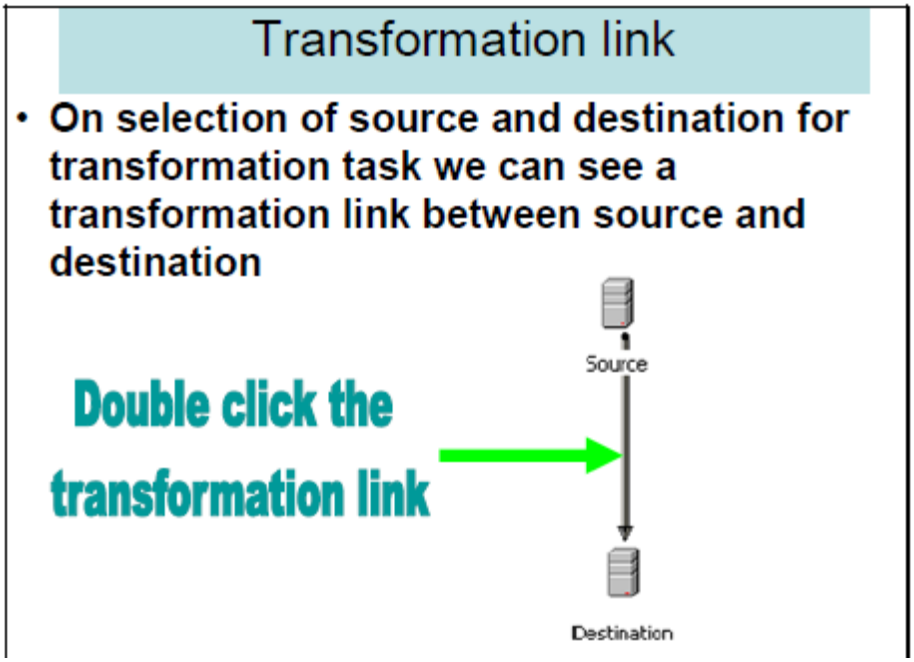
On the left side a small pane window shows all the available connections that can be established through SQL Server. At this stage our source is Student table in Lahore_Campus SQL database. Therefore, for source connection we click at the SQL Server icon and drag it to the design area.



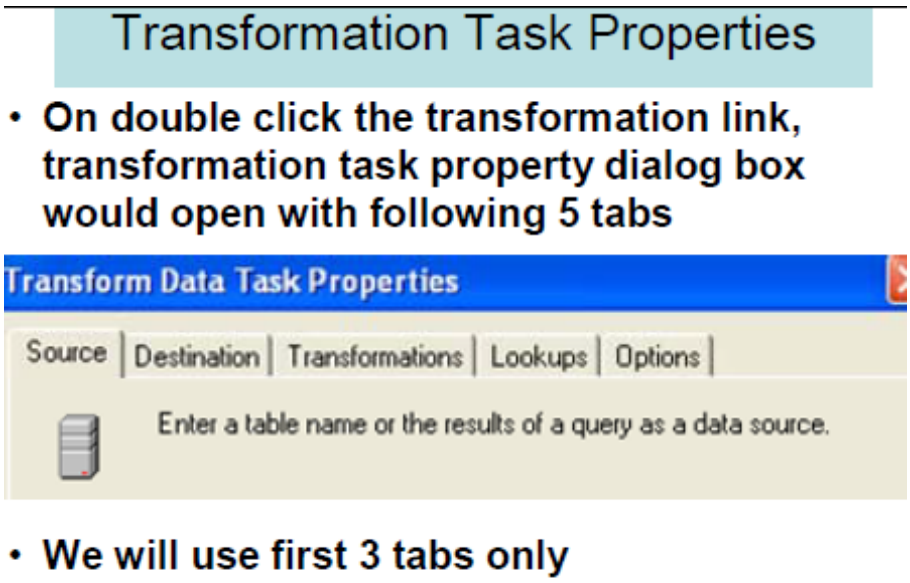
As soon as we place the connection icon to the design area another dialog window opens and provides us the way to set the properties of the connection. First of all it asks us whether we want to create a new connection or we want to use any other connection that was created before in the same package. If we want to use any existing connection we can select existing connection but here we want to create a new connection. Name the new connection as Source. Similar to source we need to create another SQL Server connection for destination table that is 'Student_Info'. In properties of connection we are required to specify the name of destination database as well.



After creating both source and destination connections we need to select the task to be performed between source and destination. All available task can be found on the left pane window 'Task' in form of icons. The task is transformation and is represented by the icon we can see highlighted in the slide. Click the transformation task icon and drag it to the design area. First click in the source connection and then click on the destination connection.



As soon as destination connection is clicked a task link appears between both connections. To set the properties and details of transformations (detail means what transformation needs to be applied) we need to double click the transformation link.



Double clicking the transformation link opens a dialog box with five tabs.

Source

Source tab let us define the source table.

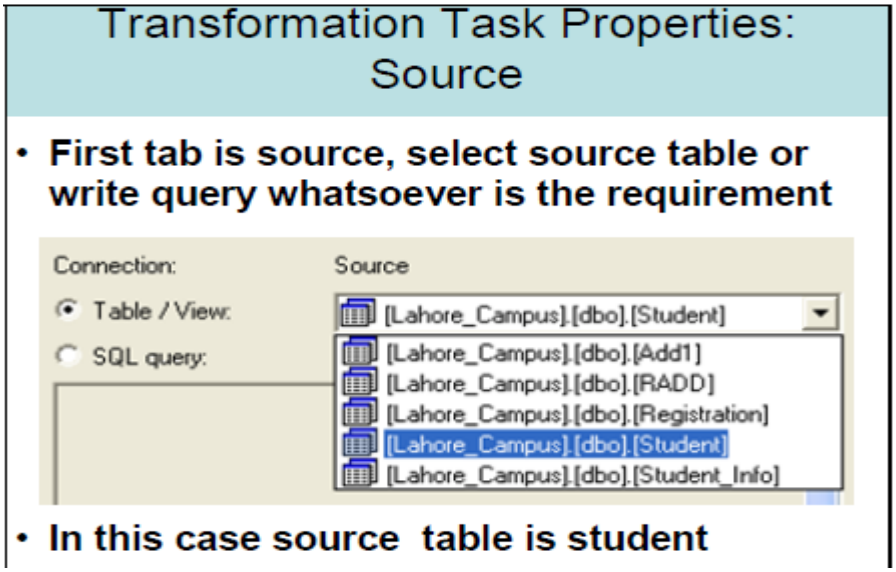
Destination

Destination tab let us define the destination table. In this case both will be the same.

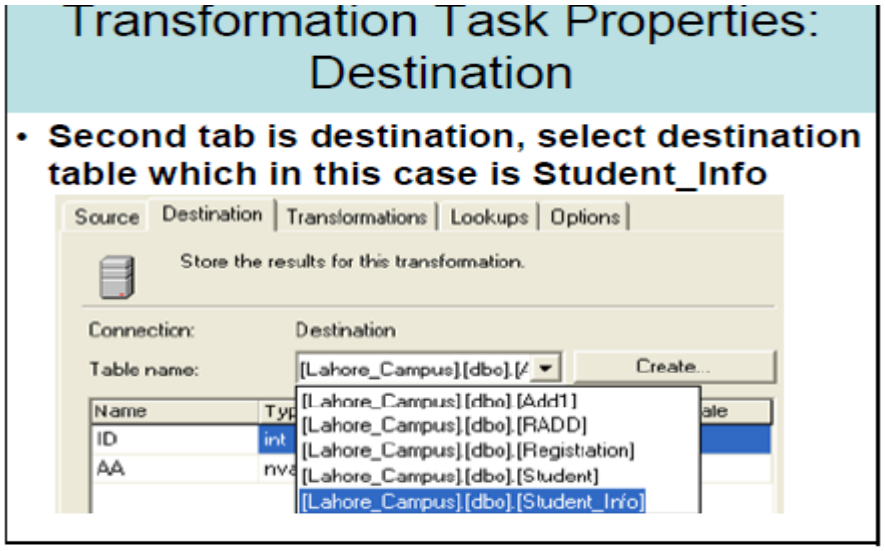
Transformation

Transformation tab let us define the type of transformations to be applied between source and destination.

Rest of the two tabs **Lookup** and **Options** will not be used by us. As all the task required to be done can be completed by these three tabs.



The slide shows the dialog under source tab. If we want to extract data from source database through some query then we may specify query by selecting the radio option 'SQL Query'. On the other hand, if we want to copy all columns of a table or a view then we may specify it through drop down menu after selecting the first radio option Table/View. The drop down menu shows all tables and views available in the database specified while setting the properties of the connection. In this case source table is student.



Second tab is for specifying the destination table if it exists. Otherwise, if destination table does not exist then we may create it through **Create** button on the right of drop down menu. In our case destination table Student_Info has been created earlier, therefore, at this stage we just need to select the Student_Info table from the drop down menu.

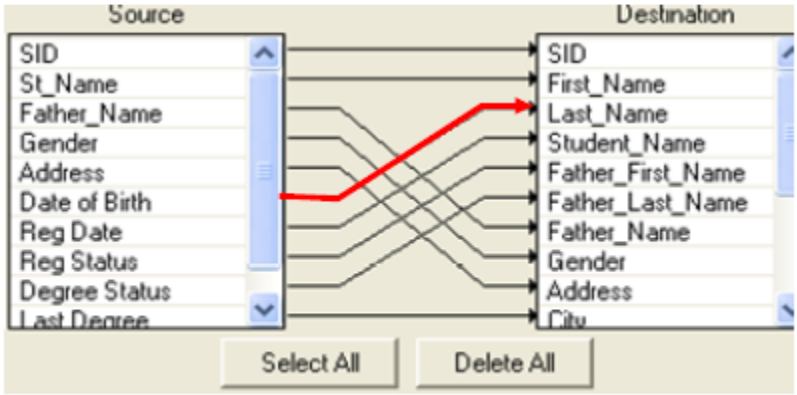
Transformation Task Properties: Transformations

- Press third tab transformation
- Designer tries to map source destination column automatically
- Drop all mappings if Designer asks you through a dialog to delete all mappings

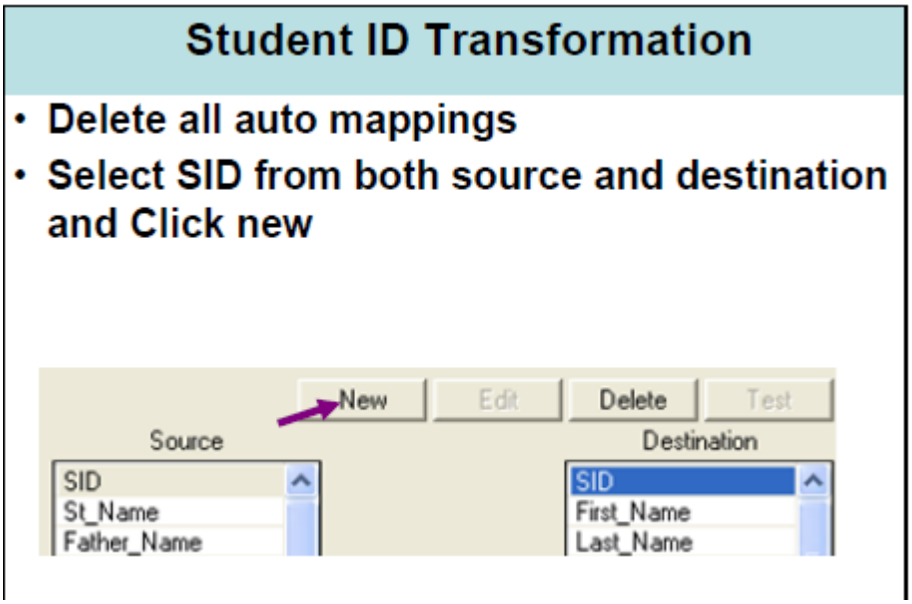
Now comes the most important tab **Transformation**. As soon as transformation tab is clicked designer tries to map the source and destination table columns automatically on the base of resemblances in the names. For example, if we have a column SID in source and a column SID in destination then designer tries to map these two columns. A dialog box appears and ask weather we want optimizer mappings remained there or we want to drop all mappings and create these mappings manually. We prefer to go for later option and drop all automatic mappings because optimizer does not transform genders (from 0/1 to M/F) or names etc.

Transformation Properties: Transformation Mappings

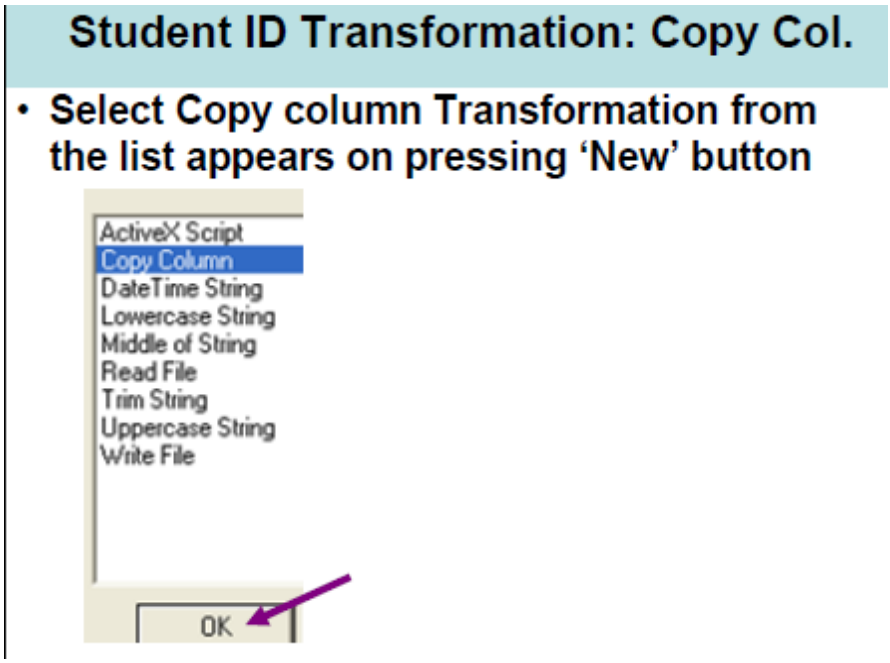
- Delete all mappings



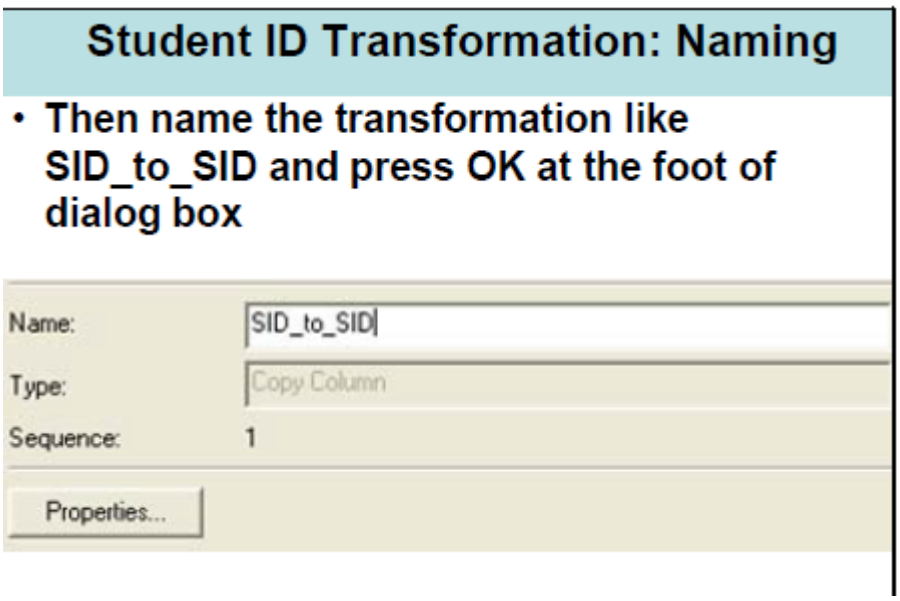
The slides show the mappings done by designer itself, delete all of these mappings.



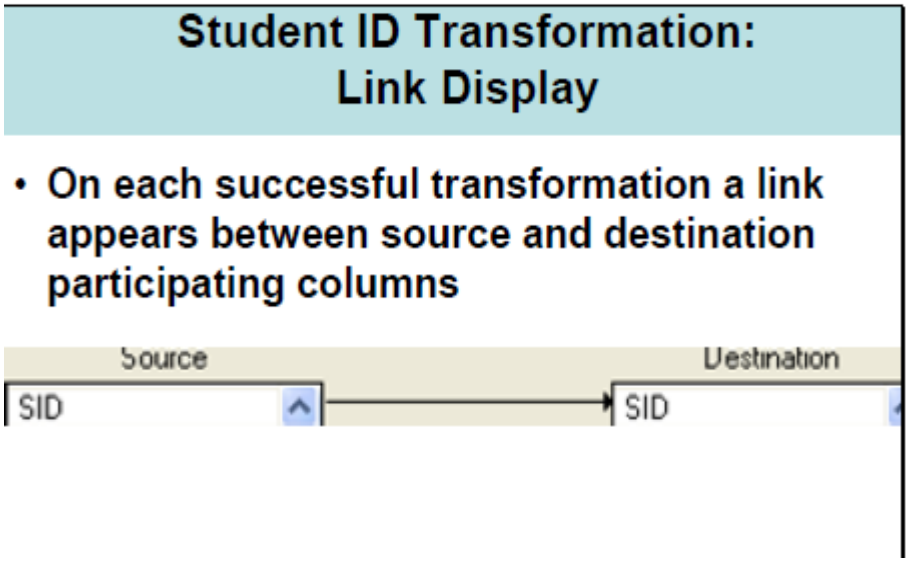
To create new mappings first of all we need to select the source column from source list box and then destination column from destination list box. When both gets highlighted as shown in the slide then press the **New** button. The slide shows highlighted SID column from both Source and destination list box. In both tables the first column is SID.



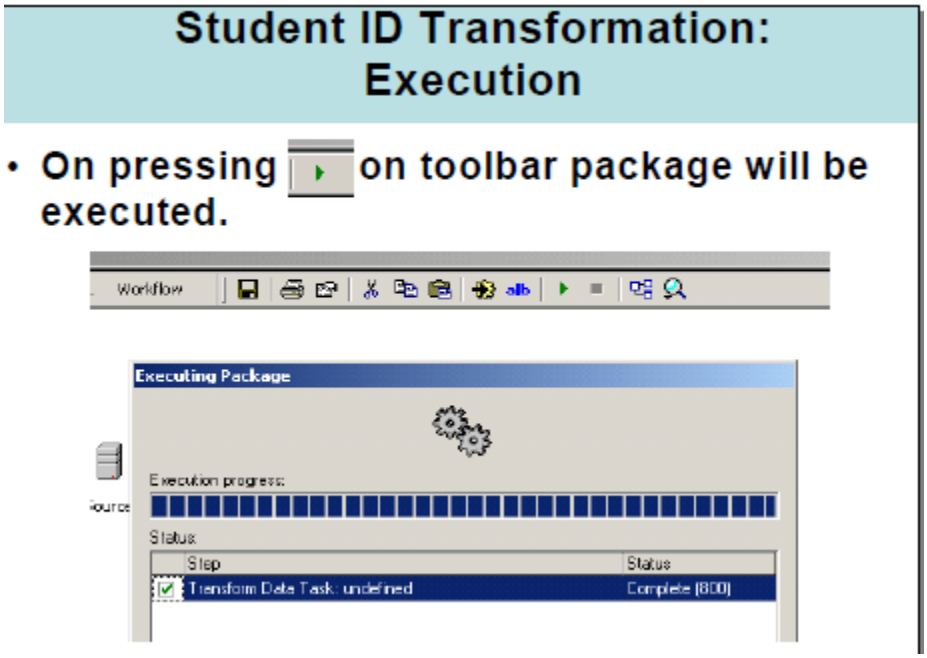
On pressing the new button, following list box appears showing all available forms of transformations. Select **Copy Column** transformation and press OK.



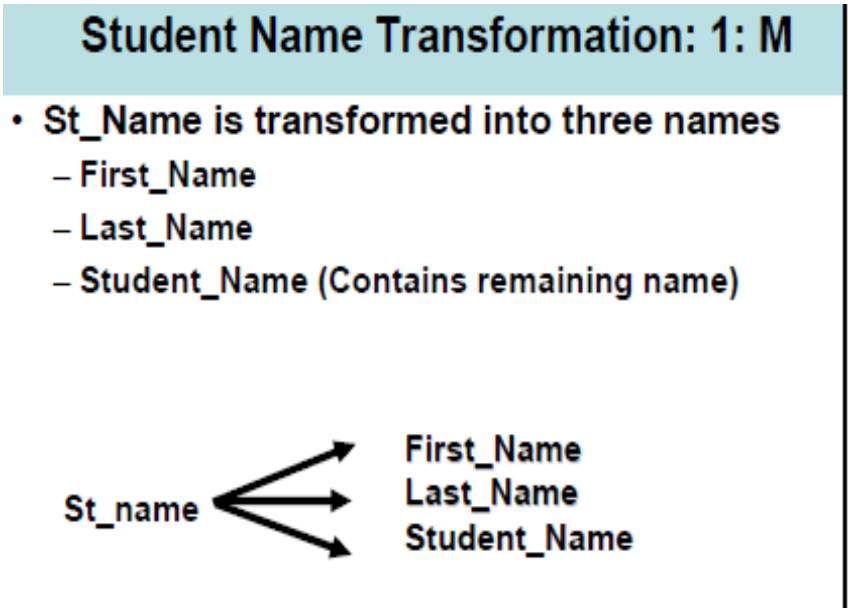
After selecting the type of transformation needs to be applied we are required to assign name to this transformation like SID_to_SID. For this transformation we are not required to set the properties as it is the simplest form of transformation just copy from source to destination. The only difference is the size of variable. At source it was varchar[250] whereas at destination it is varchar[10]. Such transformations are done automatically by SQL Server, does not need to specify in properties.



On each successful transformation a link appears between source and destination participating columns. To modify the properties of transformation we are just need to double click the link and the same properties dialog box would appear.

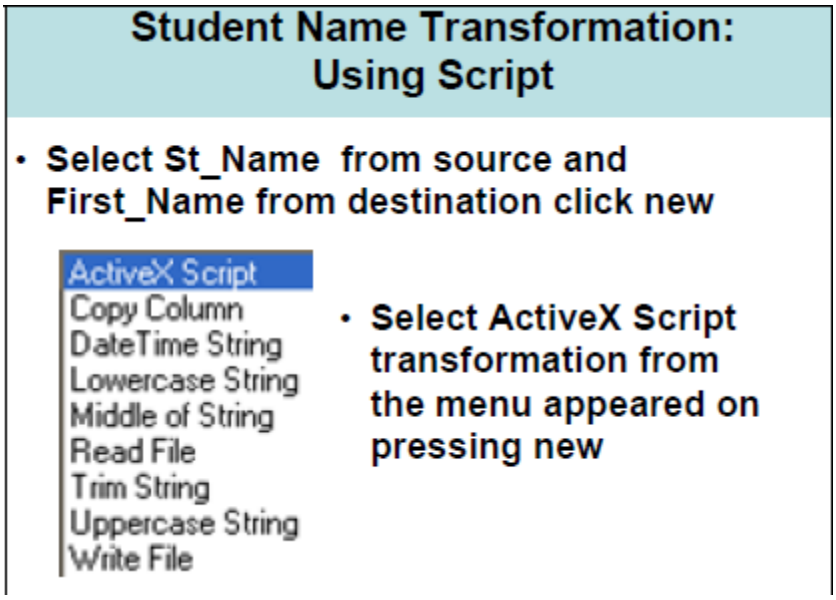


On each successful transformation a link appears between source and destination participating columns. To modify the properties of transformation we are just need to double click the link and the same properties dialog box would appear.

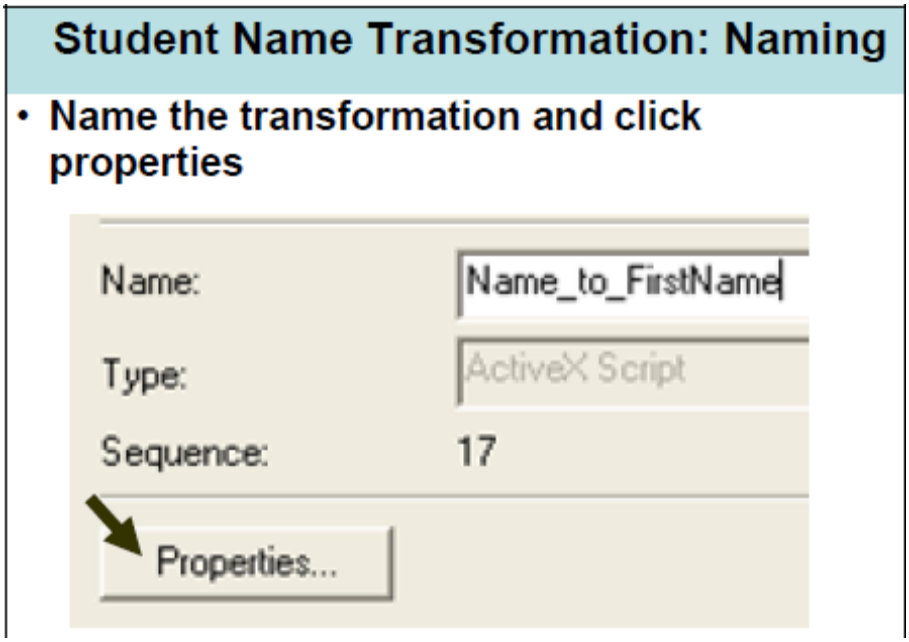


After SID we are required to transform St_Name. It is one to many transformation that is required to be specified through script. In this case is source column is just one
'St_Name' and the destination columns are three 'First_Name', 'Last_Name', &
'Student_Name'. We are required to create three new transformations,

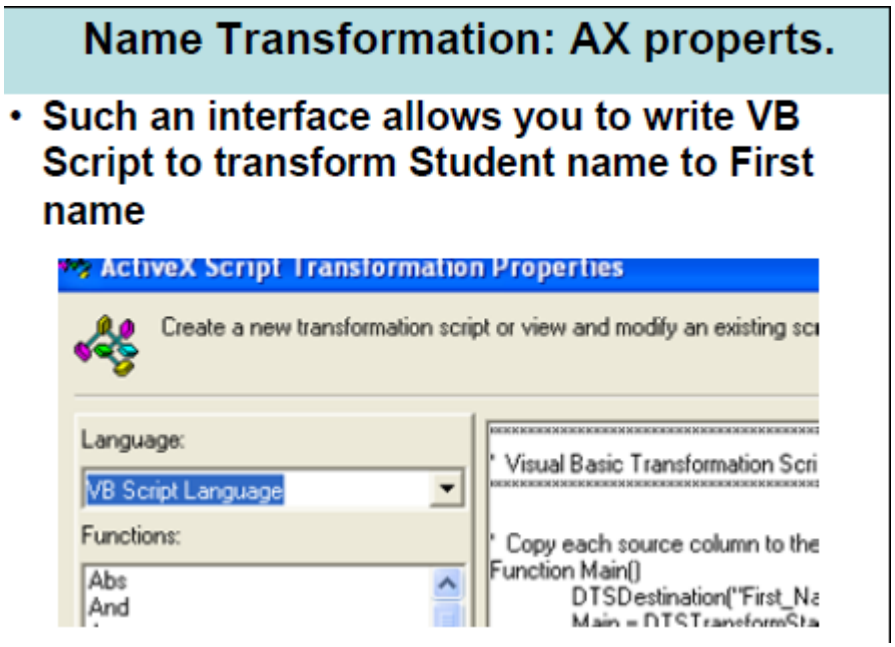
St_Name to First_Name
St_Name to Last_Name
St_Name to Student_Name
First of all select St_Name from source list box and First_Name from destination list box. When both of the columns get highlighted press the new button.



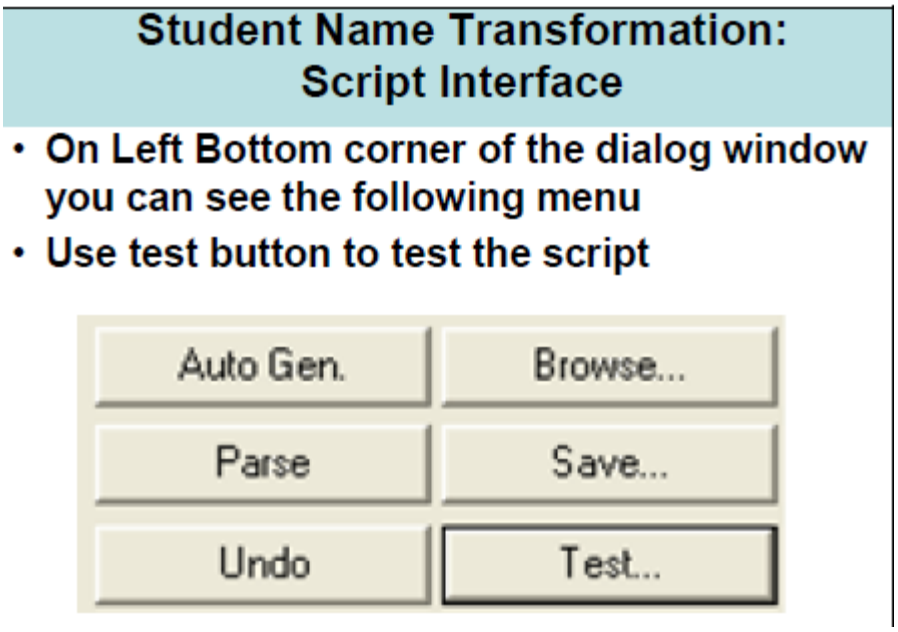
As we are required to write script to transform St_Name into First_Name therefore select Active-X Script.



Name the transformation and press **'Properties'** to write script.



Slide shows the interface within the designer to write and test scripts that are written for transformation. Language dropdown menu on the extreme left corner allow us to select options available for scripting the transformation. As we are using VB scripts therefore make sure that VB Script should be selected in the dropdown menu.



On the extreme left corner of the scripting interface dialog we can see six options as shown in the slide. We can save our scripts, we can browse them later on, we can parse them for syntax errors, we can undo last move, we can auto generate simple copy column script and we can test our scripts on actual column of database table. To verify the correctness of the script we can use **Test** option. Test option runs the script on the actual column and shows the first hundred values as an output.

Student Name Transformation: Transformation Script

- **Following Script allows you to transform Student name to first name**

```
' Visual Basic Transformation Script
************************************************************************

' Copy each source column to the destination column
Function Main()
        DTSDestination("First_Name") = FName(DTSSource("St_Name"))
        Main = DTSTransformStat_OK
End Function

Function FName(name)
        FName  =Trim( Left(Trim(name), InStr(name," ")))

End Function
```

The slide shows the script that has been written to transform St_Name from source table to First_Name of destination. If you do not know VB Script you can also use same script with correct column names.

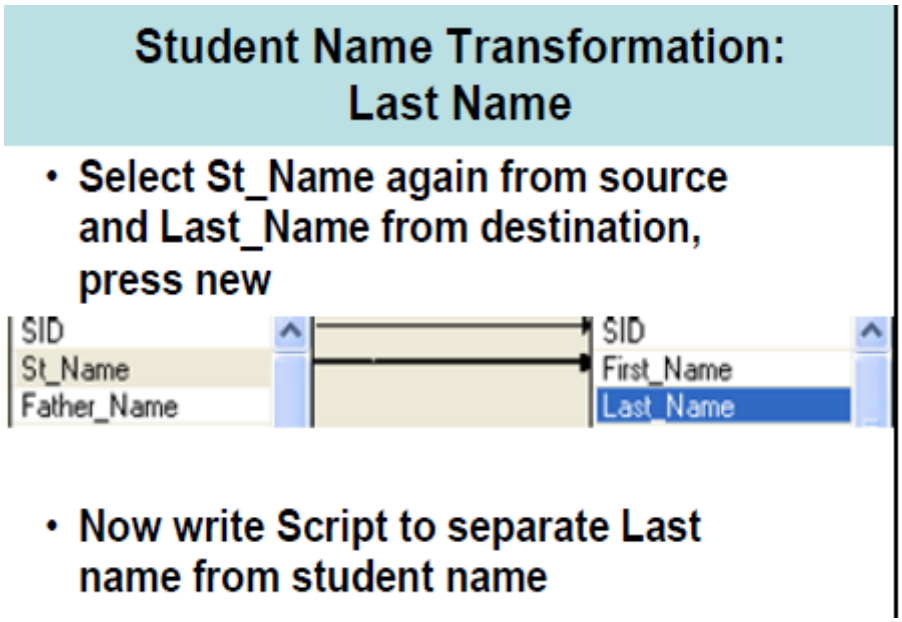
Student Name Transformation: Test

Source: D:\DOCUME~1\lmr

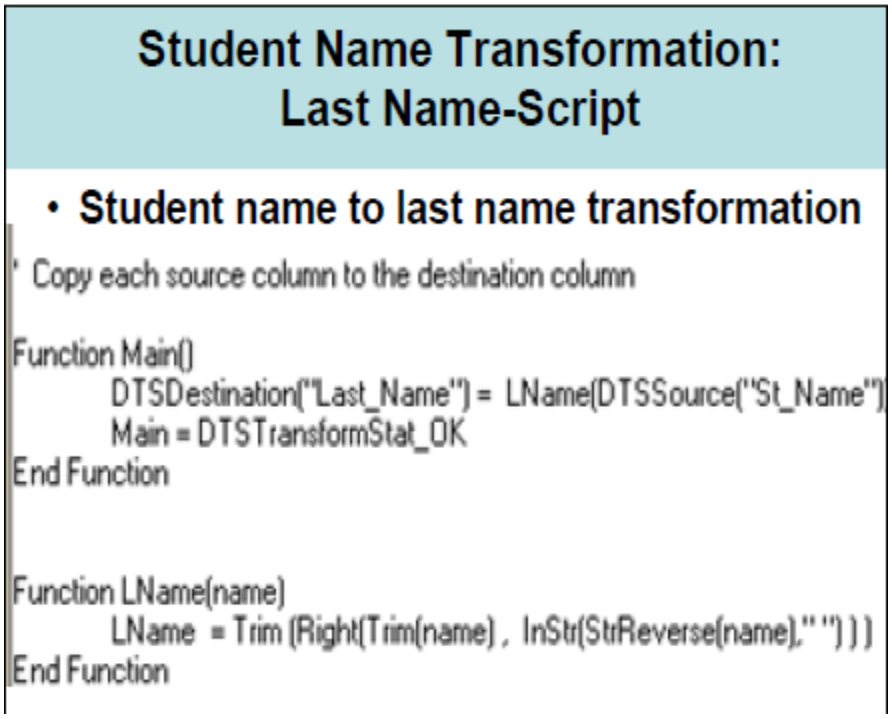
First_Name
Hafiz
Muazzam
Babar
Muhammad
Hadiqa
Masood
Momd.
Abida
Abdur

- **Output of the test run**
- **Working fine**

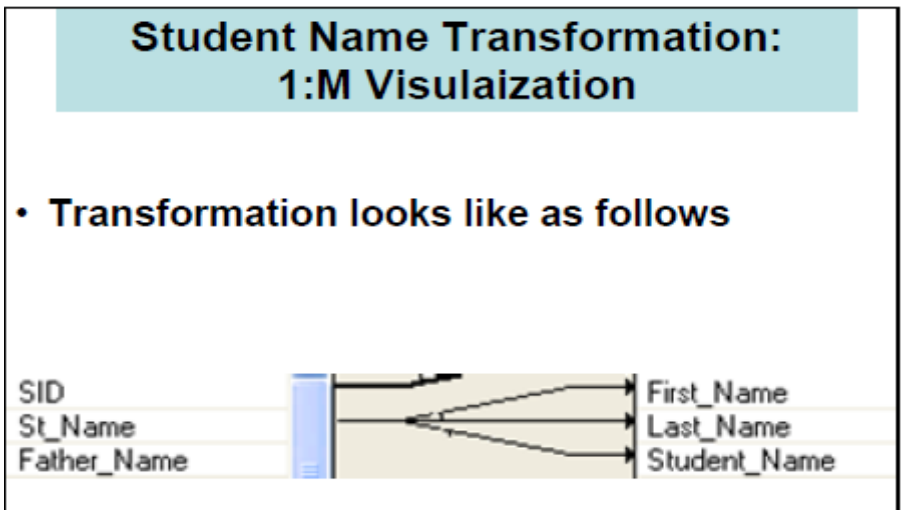
The slide shows the output of the test run of given script on Lahore_Campus database.
We can see all first names are separated from the actual full names. This is the desired transformation for First_Name.



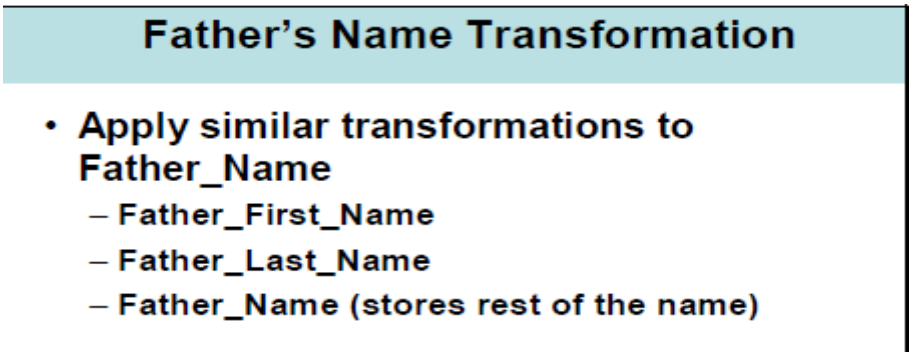
Now we will separate the last name from the names of the students. To create this type of transformation we are required to select **St_Name** from source list box and **Last_Name** from the destination list box and press **New button**.



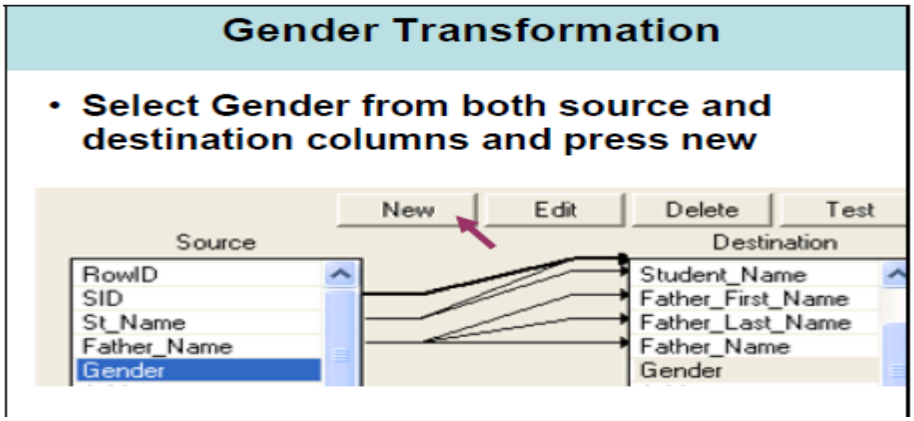
The slide shows VB Script for separating last name from full name of the student.



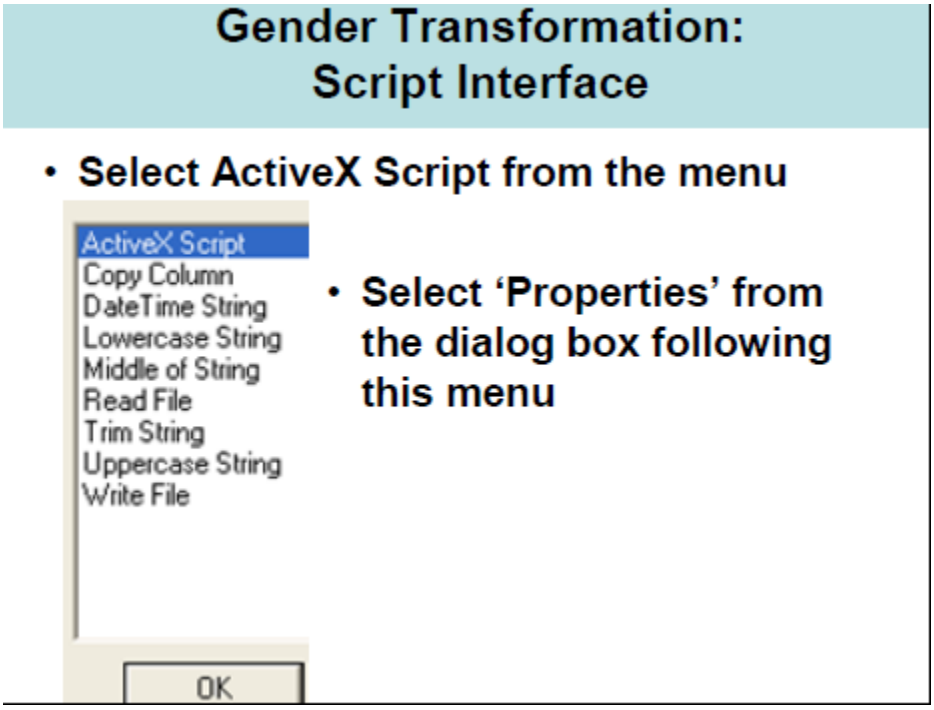
After creating all of the transformations
St_Name to First_Name
St_Name to Last_Name
St_Name to Student_Name
Link for one to many transformations looks like the one shown in the slide.



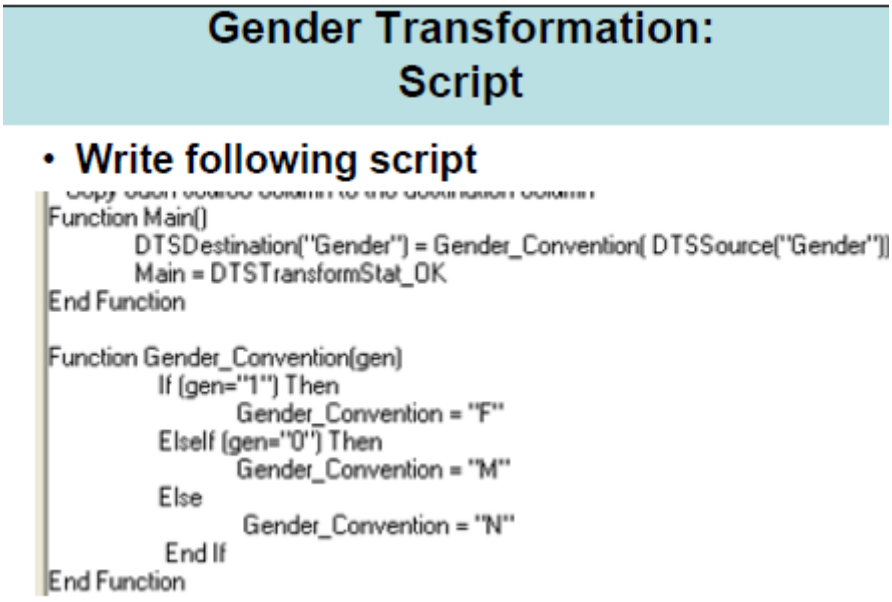
We need to apply similar one to many transformation to father name column.



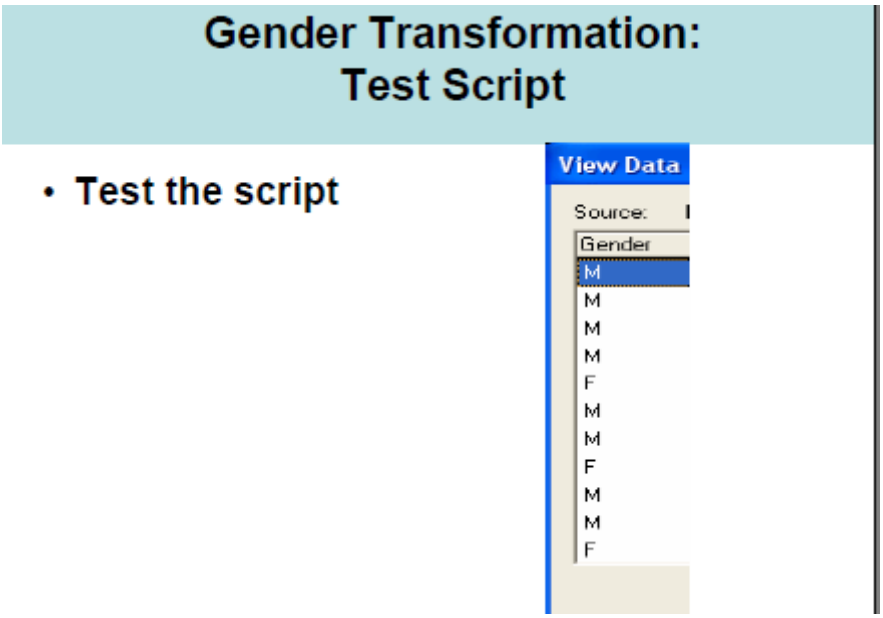
Now we need to create another transformation for standardizing the conventions to store gender that is **M** for male and **F** for female. Select Gender from both source and destination and press **'New'.**



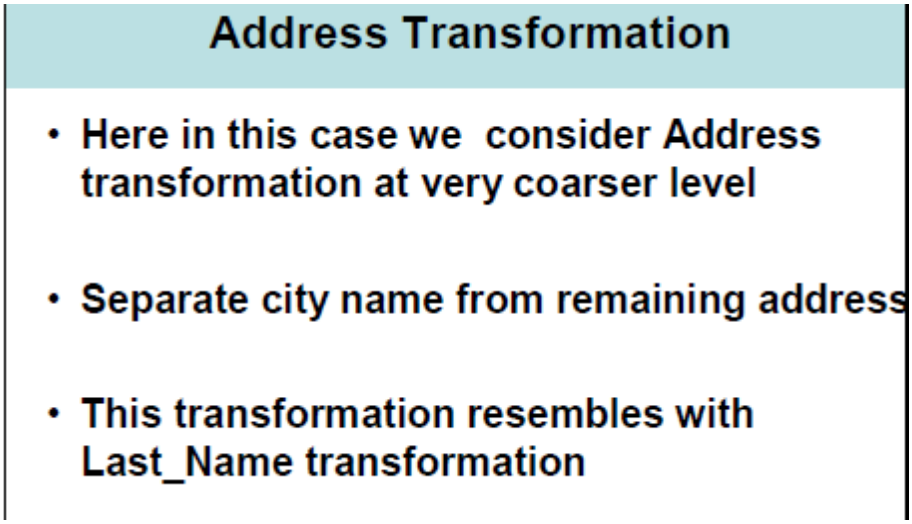
Again select ActiveX script.



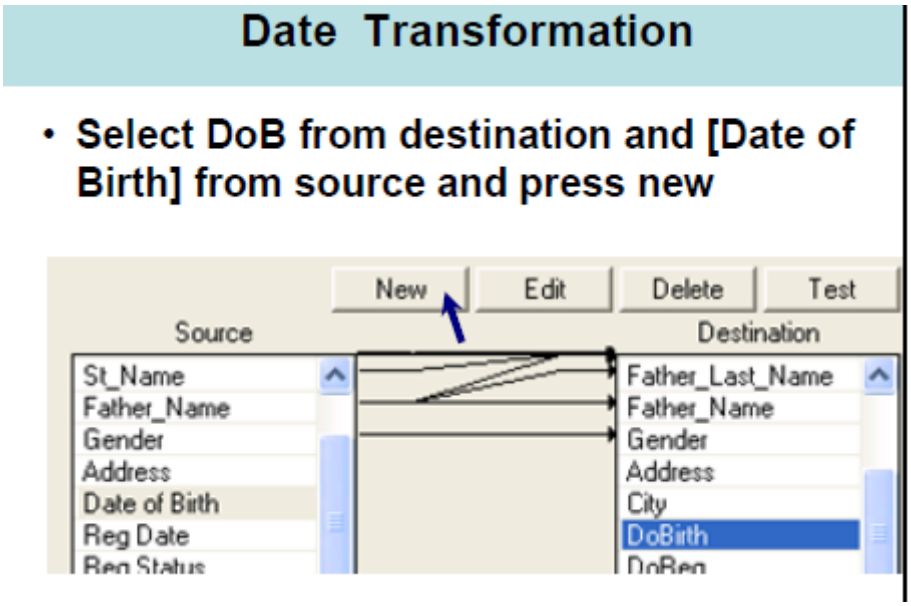
The slide shows the activeX script to standardize the gender convention.



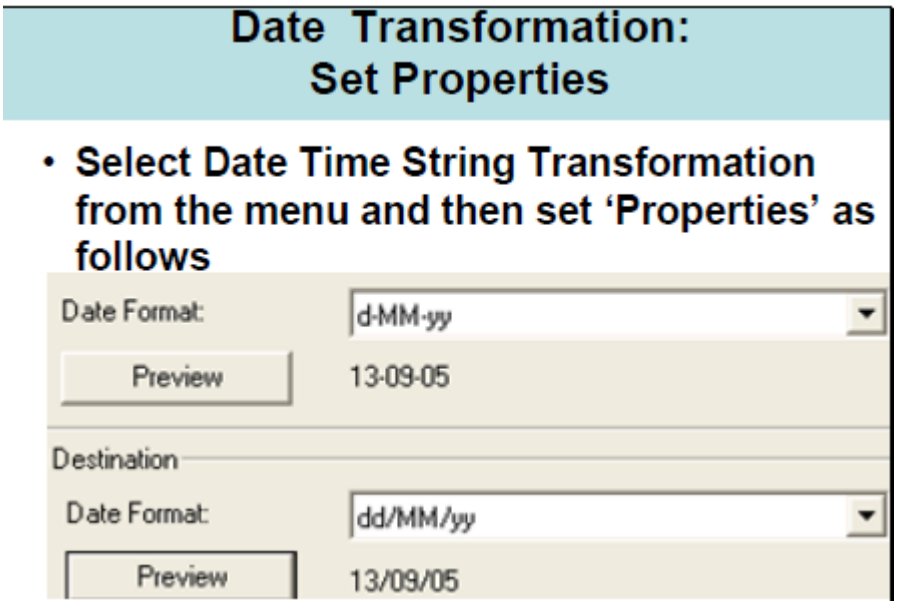
Test run shows that the script is working fine.



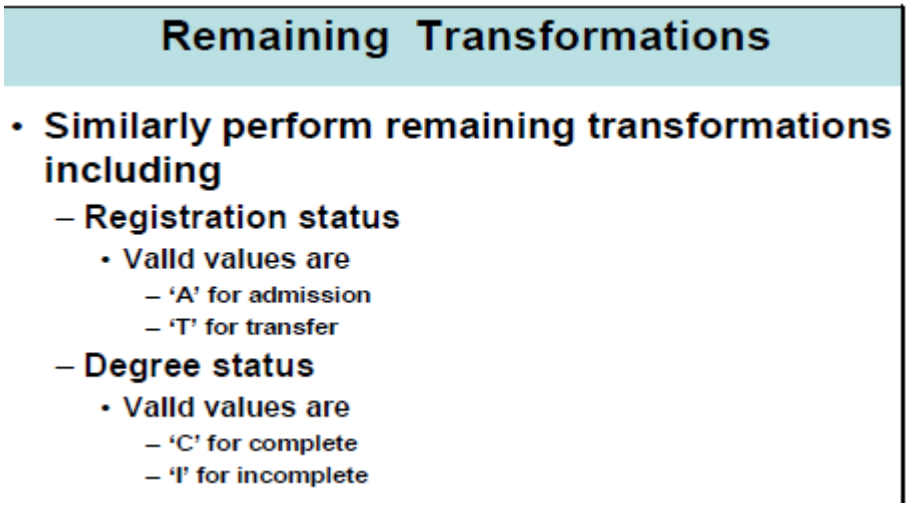
Now comes to another common example of one to many transformation that is Address transformations. For real life purposes we usuallly break address into many columns like house #, Street #, City, State, Region etc. It depends on the requirements of the users and customers. For the given example we will break the address into two Column City and remaining address. This example resembles to the separation of last name from student full name because in all records city is the last part of the address. So we can use the same script with certain modification of column names.



Now we need to transform the date of birth column. Select **DoB** from destination and **[Date of Birth]** from source and press **New**. Select **Date Time String** transformation from the list of available transformations.



In the properties of Date Time String transformation we can see the dialog box as shown in the slide. In this dialog box we can see two dropdown combo boxes with title **Date Format**. In first combo box select or manually specify (through typing) the format of the date in the source column. In the second box select the format for destination and click **ok**.



Similarly perform remaining transformations including **Registration Status** and **Degree Status**. You can generate scripts for them easily by performing little modifications in gender transformation script.

By this time all transformations are completed. Now we are required to standardize data. Like we can see in the name column that there are a lot of inconsistencies in names due to variations in spellings. We can find many variations in spellings of same name like Mohd., Mohammed, Muhammed, Mohamad, etc.. For meaning full analysis we must have standardized data in our columns because computer can not recognize that Mohd. Khalid and Mohammad Khalid is the same name. Same is the case with the names of cities. Some people use abbreviations in addresses while some others like to write full names e.g. Lahore or Lhr. In this example we will perform standardization for Names, Addresses and Last degree.

Name Standardization

- Name is transformed into following three fields
  - First name (First_Name)
  - Second name (Student_Name)
  - Last name (Last_Name)
- We will devise a simple strategy to standardized name that can later be followed by any other column (city / last degree)

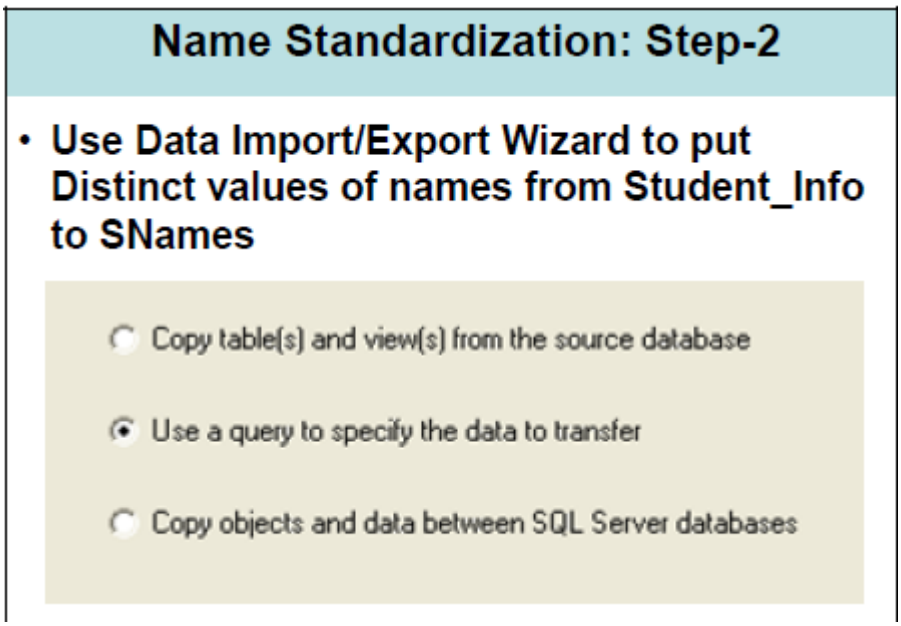
There are no fixed strategies to standardize the columns. Again it depends on the project designer what methodology he/she devises. We can devise a simple methodology that can later be used for other columns as well.

Name Standardization: Step-1

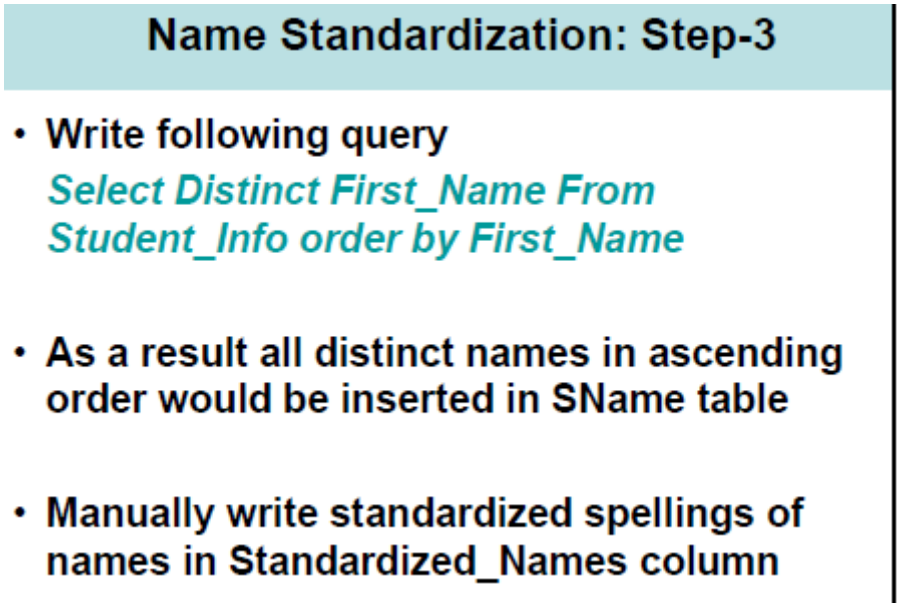
- Create a new table 'SNames' with two columns

Column Name	Data Type	Length	Allow Nulls
Name	varchar	15	✓
Standardized_Names	varchar	15	✓

This methodology some what looks like the one we used to fill the missing gender information of students? Create a new table with two columns 'Name' and 'Standardized_Names'.



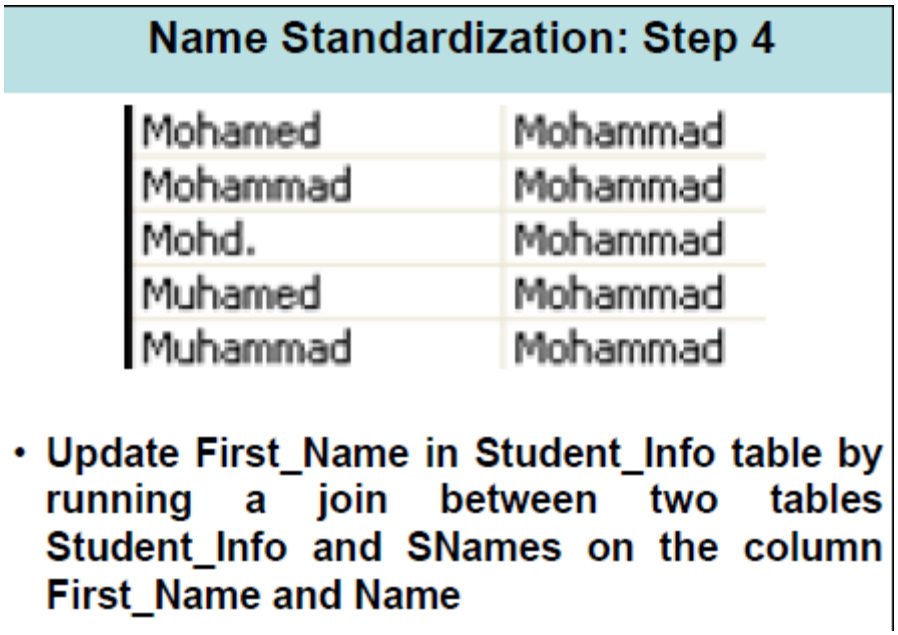
Create a new package through wizard to put the distinct names of the students from Student_Info table. Select second radio option that is use a query to specify the data to transfer. Write a query to select distinct names from Student_Info table.



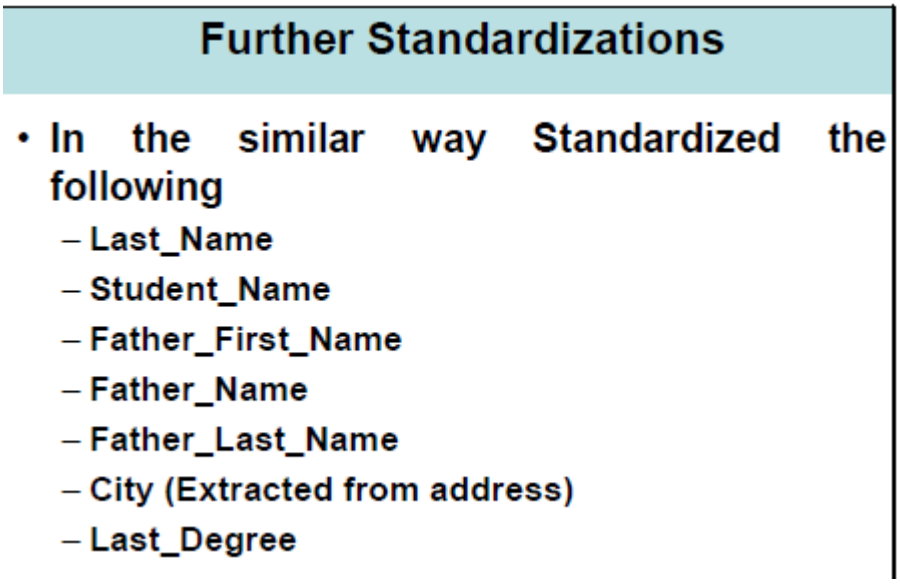
Following query can serve the purpose of selecting distinct names from Student_Info table and loading them to SName table in ascending order.

Select Distinct First_Name From Student_Info order by First_Name;

Load all names to Name column of the SName table and against each name write its standardized spellings in Standardized_Names column manually.

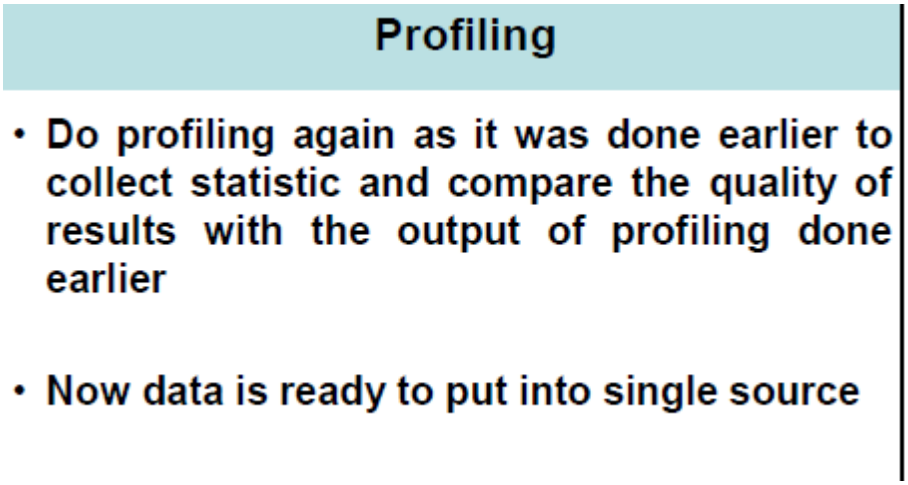


The slide shows a screen shot of SNames table. We can update all first names in Student_Info table just by joining it with SName on the column First_Name (Student_Info) and Name (SNames).



In the similar way standardize the rest of the columns. We can standardize all of the following columns with the same methodology:

Last_Name
Student_Name
Father_First_Name
Father_Name
Father_Last_Name
City (Extracted from address)
Last_Degree



To test the quality of all cleansing and standardization process we are required to profile all columns again. This time there should be no missing and invalid values in the data. As we cannot attain hundred percent cleanliness levels therefore for remaining errors and invalidities in the data we should have the proper reasoning. After all this exercise now we are ready to put all data into single source.