Testing Masters
TECHNOLOGIES

ETL Testing Material

Testing Masters

This document contains material for ETL Testing

Address:
#104,Nandini
Residency,
Addagutta Society,
Pragathi Nagar Road
Jntu X-Roads,HYD

Index

6. Transformations in ETL
    - Sample Load from Source to Target
    - Joiner Transformation
    - Derived columnTransformation
    - Lookup Transformation
    - Union Transformation
    - Sorter Transformation
    - Conditional Split Transformation
    - Script component
7. Introduction to ETL Testing
8. ETL Testing Concepts
    - Requirements Gathering
    - High Level Design
    - Low level Design
    - Development
    - Unit Testing
    - Integration Testing
    - Test cases preparation
9. Types of ETL Testing
10. ETL Testing Defects
11. Defects Reporting

DATA WAREHOUSING

Chapter 1 – Data warehousing Basics:

1.1 Need for Data Warehousing

Every Organization is in need of maintaining the data. The means of maintaining data can be different means such as Books, tapes, excels. As time goes on changing, the affectivity of the data maintenance and handling the data also started changing. Let's think of handling data when there are more than 1000 customers or employees in any organization and their data needs to be preserved to be handled in later times. All this has not proved successful until datawarehousing came in to place.

In challenging times good decision-making becomes critical. The best decisions are made when all the relevant data available is taken into consideration. The best possible source for that data is a well-designed data warehouse.

Some Organizations are really small which cannot afford for an data warehouse. They can actually use data marts which resembles the data warehouse where their transactional data is very low. Data warehouse has become more efficient when handling data for larger organizations.

1.2 What is Data Warehousing?

Data warehousing combines data from multiple, variable sources into one database which can be easily manipulated. This can be used for Analysis, Transformation and Reporting. Usually Larger companies use this data warehousing for analyzing trends over a period of time for viewing transactional data and plan accordingly.

Data warehouse is defined as subject oriented,Integrated,Time Variant and Non-Volatile collection of data for the management for decision making processing.

Subject Oriented: Thisis used to analyze particular subject area.

Integrated: This shows that integrates data from different sources.

Time variant: Historical data is usually maintained in a Data warehouse, i.e. retrieval can be for any period. This usually contrasts with the transactional system, in which only the most recent data is maintained. But in the Data warehouse,the transactional data is moved periodically where the recent and the previous data is also maintained.

Non-Volatile: Once the data is placed in the data warehouse,it cannot be altered,which means we will never be able to alter the historical data.

1.3 OLTP and OLAP:

OLTP Online Transaction Processing System

OLTP is nothing but a database which actually stores the daily transactions which is called as the current data. Usually OLTP is used for more of the online applications where the application needs to update very frequently in order to maintain consistency in the data.

OLTP deals with the large number of data.

OLAP Online Analytical Processing System

OLAP deals with analyzing the data for Decision making and planning.This actually deals with the aggregations and the data in OLAP will be from different data sources

Compared to OLTP, OLAP deals with relatively small amount of data.

1.4 Difference between Database and Data warehouse

Database: This can be treated as an OLTP system where it stores the data from the applications for use.Small applications can run with a database.

Data warehouse: This is accumulation of the all the data related to the application, It can be from the Database where the Transaction data resides, Flat files that are used,Legacy or Mainframe sources.Large organization will need a data warehouse where there is a need for analytics for decision making.

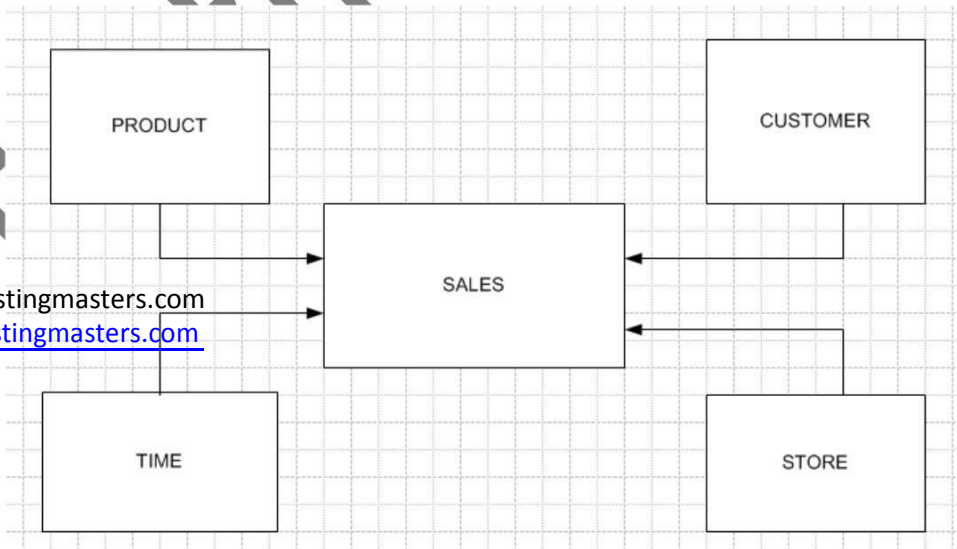
Database	Data Warehouse
Database has the current data which has a chance of Updating day by day.	Data warehouse Stores the Historical data where the accuracy is maintained over period of time.
Contains the Day to day operations data.	Contains the Long term Operations data.
Database professionals, agents access this particular data.	Managers, Analysts access the data warehouse data
Data in the Database will always be in the form of Input.	Data in the Data warehouse will always be the output data where it is used for Analyzing trends.
Database is used for Transactions.	Data warehouse is used for Analytics.
Database has both Read/Write access.	Data warehouse has only the Read access.
Database contains only few number of records compared to a Data warehouse.	Data warehouse contains millions of records as the DWH gets refreshed with historic data.
Database is always Normalized.	Data warehouse is Denormalised
The data view in the database will always be Relational.	The data view in the Data warehouse id always Multidimensional
Database contains the detailed data.	Data warehouse contains the consolidated data.

2 Data warehousing Concepts:

2.1 Data Model: Data model tells how the logical structure of a database is modeled. Data Models are fundamental entities to introduce abstraction in DBMS. Data models define how data is connected to each other and how it will be processed and stored inside the system.

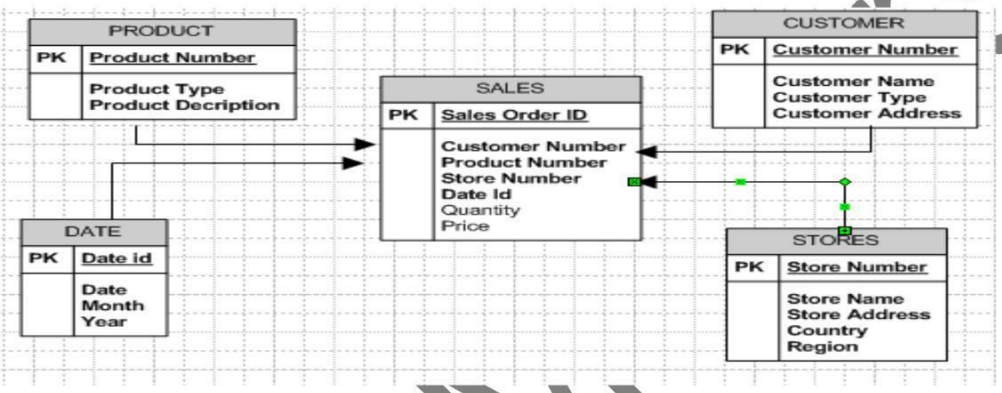
2.1.1 Conceptual DataModel: This usually pictures the highest level of relationshipbetweenthe entities.

a. Displays the important entities and the relationships among them.
b. No attribute is specified.
c. No primary key is specified.



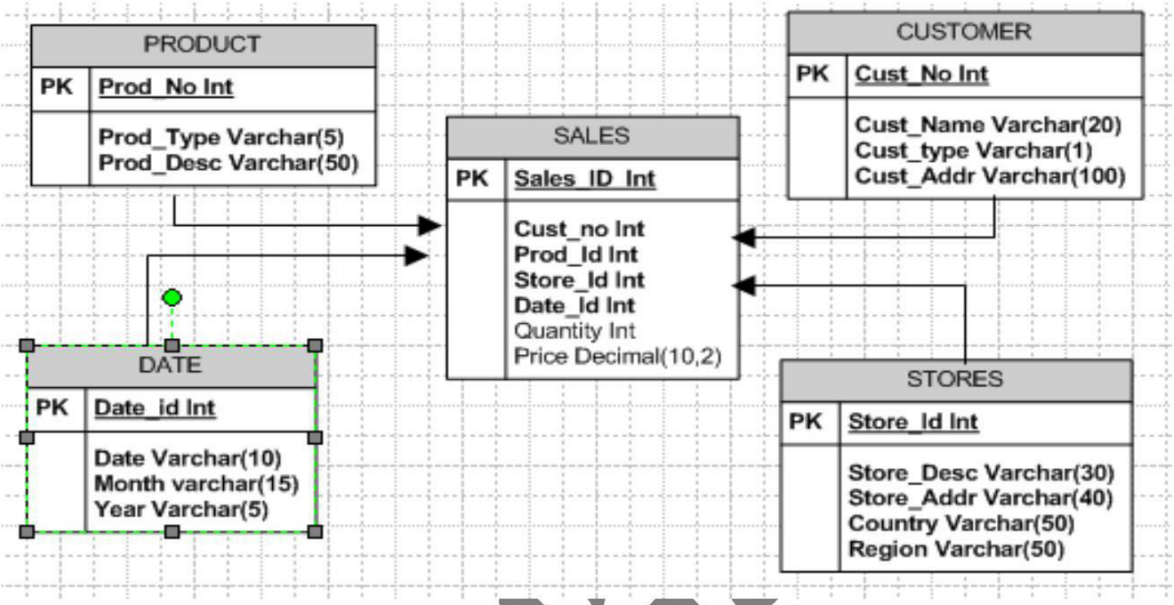
2.1.2 Logical Data Model: This defines the data as much as possible, to show how they can be physically implemented in the database.

a. Displays all the entities and the attributes and the relationships between them.
b. Primary key for each entity is specified.
c. Foreign keys for each entity if exists is specified.
d. Normalization is performed.



2.1.3 Physical Data Model : This defineshow the model is physically existing in the system

a. Displays all the tables and columns.
b. Displays foreign keys.
c. Displays lookup tables.
d. Change the relationships into foreign keys.
e. Entity now becomes table.
f. Attribute now becomes column
g. Datatypes are also shown in this model.



Difference between Conceptual, Logical, and Physical Data model

Conceptualis just a high level representation of the Entities that were used as a part of DB design.

Logical is the next level representation for a Conceptual design in which the entities and attributes comes into picture but with understandable English names. The Relationships are also defined.

Physical is the more granular level of representing the DB design, in which the entities are now represented as Tables and the attributes are represented as columns along with the primary and foreign key relationships. By looking at the physical design, a person can develop the Database,which in other means can be called as the physical representation of the database.

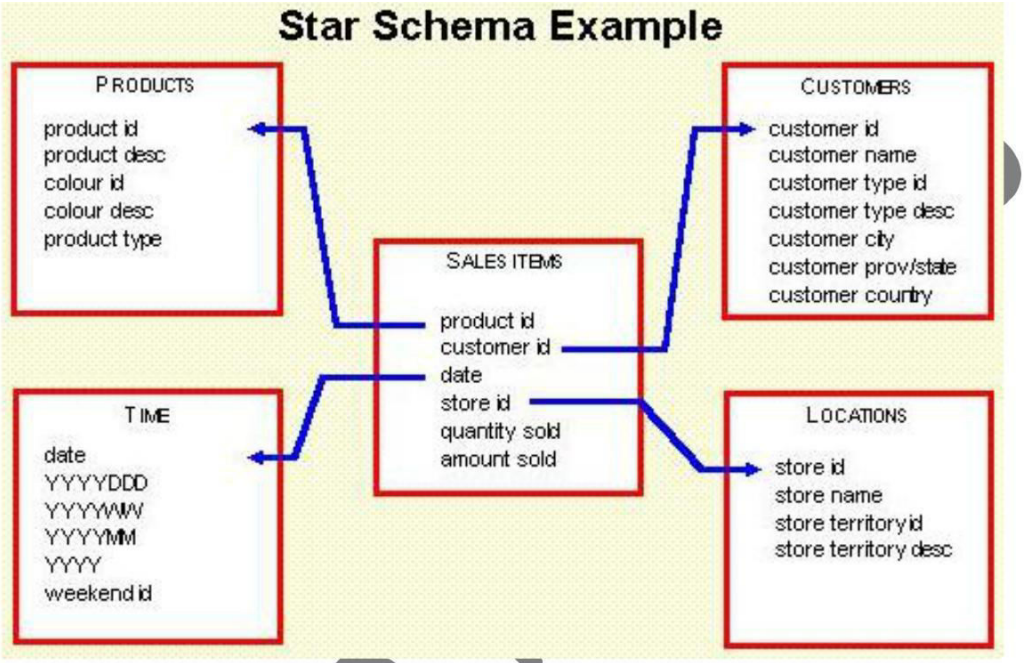
2.1.4 Dimensional Data Model:

This defines the Data modeling technique by defining the schema (Star and Snowflake schema).

2.2 Schema: A database schema defines its entities and the relationship among them. Database schema is a descriptive detail of the database, which can be depicted by means of schema diagrams. All these activities are done by database designer to help programmers in order to give some ease of understanding all aspect of database.

2.2.1 Star Schema: This consists of the fact table in the middle and the dimensional table around it.

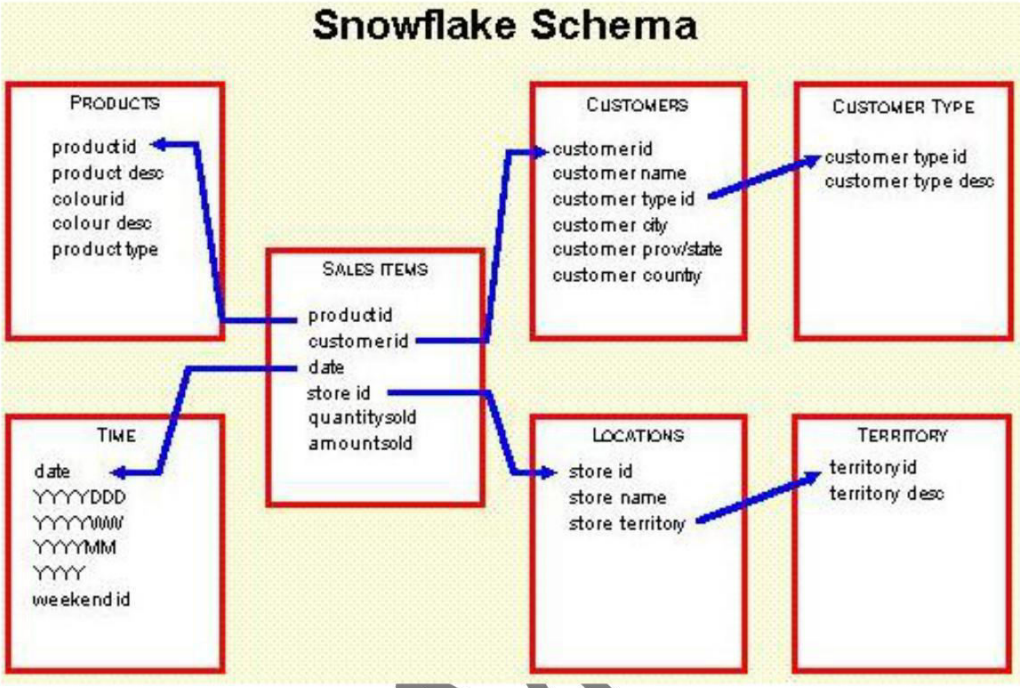
Fact table is usually a sum of all the dimensions.Dimension table is a single entity.A primary key in a dimension table is represented as a foreign key in a fact table.



2.2.2 Snowflake Schema: This is an extension of the Star Schema, where each point of a start is divided into more granular level.

Each Dimension table is still normalized into multiple lookup tables.The main advantage of the snowflake schema is to improve the query performance due to minimized disk storage and joining can also happen on smaller lookup tables.

Disadvantage is maintenance efforts to manage the number of lookup tables.



2.3 SCD Types:

Slowly changing dimensions (SCD) determine how the historical changes in the dimension tables are handled. Implementing the SCD mechanism enables users to know to which category an item belonged to in any given date.

SCD is more particular to Data warehousing,where the record varies over time.Slowly Changing Dimensions are often categorized into three types namely Type1, Type2 and Type3.

Ex: Consider a customer with name Rahul who is living in London from 2003.This can be depicted in a table as below:

Customer Number	Customer Name	Year	Location
1001	Rahul	2003	London

Type 1 SCD: Replaces the old entry with the new value

The customer Rahul has moved from London to Paris in the year 2005

In this Type 1 SCD, the table will be changed as below:

Customer Number	Customer Name	Year	Location
1001	Rahul	2005	Paris

In this case, the previous entry which is treated as history is lost.

Type 2 SCD: Creates an New entry in the table

The New record is inserted into the same table, In this Type 2 SCD, the table will be changed as below::

Customer Number	Customer Name	Year	Location
1001	Rahul	2003	London
1001	Rahul	2005	Paris

In this case, each record will be treated differently, which makes the table to grow fast and complicates the ETL process. This is mainly for tracking the historical changes.

Type 3 SCD: Creating New Fields in the table

New fields are added to the table to main the history

In this Type 3 SCD, the table will be changed as below:

Customer Number	Customer Name	Year	Location	Old Year	Old Location
1001	Rahul	2005	Paris	2003	London

The previous record itself is modified such that neither the history is lost and even the new record is also displayed. But this can accommodate only one change.This Type3 SCD is rarely used, when the changes are prone to change only once.

2.4 Normalization:

Normalization is a process of eliminating the redundant data and storing the related information in a table.

The key points of Normalization isas below:

- Eliminating Redundant Data
- Faster Update
- Improve Performance
- Performance in Indexes

Below are the different Normal Forms

2.4.1 First Normal Form: If the table is said to be in the 1st Normal Form it should follow the below rules

- Each cell must have one value.
- Eliminating Duplicate Columns
- Create separate table for the group of the related data and each row should be identified by Primary Key.

Let us take an example here:

Name	Department	Phone Number	Salary	TAX
Name 1	Computers	12354588,6887890,123456	4000	40.00
Name 2	Electronics	12345678,2345666,789654333	5000	50.00
Name 3	Civil	4567890	3000	30.00

In the above table we see that there are different phone numbers for the single Name and we have to remove these duplicates by uniquely identifying each of them and giving a unique identification by giving them the Primary Key.

Now the below table is redesigned such that it is in the First Normal Form.

ID	Name	Department	Phone Number	Salary	TAX
1	Name 1	Computers	12354588	4000	40.00
2	Name 1	Computers	6887890	4000	40.00
3	Name 1	Computers	123456	4000	40.00
4	Name 2	Electronics	12345678	4000	40.00
5	Name 2	Electronics	2345666	4000	40.00
6	Name 2	Electronics	789654333	4000	40.00
7	Name 3	Civil	4567890	3000	30.00

2.4.2 Second Normal Form: If the table is said to be in the 2st Normal Form it should follow the below rules

- It should satisfy the 1st Normal Form.
- Separate the particular Columns, values are duplicated in each row should be placed in the separate Table.
- Create the relationship between the tables.

Here in the above table we see that the Name and the department columns are duplicated and in order to handle this we need to maintain the duplicates in the different table as below:

EMPID	Name	Department	Salary	TAX
1	Name 1	Computers	4000	40.00
2	Name 2	Electronics	4000	50.00
3	Name 3	Civil	3000	30.00

ID	EMPID	Phone Number
1	1	12354588
2	1	6887890
3	1	123456
4	2	12345678
5	2	2345666
6	2	789654333
7	3	4567890

Here in these tables above EMPID is treated as the primary Key for the First Table and the Foreign Key for the Second Table.

2.4.3 Third Normal Form: If the table is said to be in the 3rd Normal Form it should follow the below rules

- It should satisfy the 2nd Normal Form.
- Separate the particular Columns that are not dependent on the primary key of the table.

EMPID	Name	Department	Salary
1	Name 1	Computers	4000
2	Name 2	Electronics	4000
3	Name 3	Civil	3000

ID	EMPID	Phone Number
1	1	12354588
2	1	6887890
3	1	123456
4	2	12345678
5	2	2345666
6	2	789654333
7	3	4567890

Salary	TAX
4000	40.00
3000	30.00

2.4.4 Fourth Normal Form: If the table is said to be in the 4th Normal Form it should follow the below rules

- It should satisfy the 3rdNormal Form.
- The Non Key columns should be dependent on fully primary key instead of partial key, fits not separate the tables.

Here in the above as we have already put a primary key in the table during the First Normal Form the above tables are in Fourth Normal Form also.

Chapter 3 - Business Intelligence:

Business intelligence is the set of techniques and tools for the transformation of raw data into meaningful and useful information for business analysis purposes. BI technologies are capable of handling large amounts of unstructured data to help identify, develop and otherwise create new strategic business opportunities. The goal of BI is to allow for the easy interpretation of these large volumes of data.

Difference between Data warehouse and Business Intelligence

Data warehouse is a way of storing data and creating information through leveraging data marts. Data marts are segments or categories of information and/or data that are grouped together to provide 'information' into that segment or category. Data warehouse does not require Business Intelligence to work. Reporting tools can generate reports from the DW.

Business Intelligence is the leveraging of DW to help make business decisions and recommendations. Information and data rules engines are leveraged here to help make these decisions along with statistical analysis tools and data mining tools.

The Business Intelligence tool that we are going to learn is SSIS which uses SQL as the Backend, Sonow let us learn the SQL Basics

SQL

Chapter 4 - SQL (Structured Query Language):

This is a standard language for accessing the database which is used for retrieval and management of data.It includes database creation, deletion, fetching rows and modifying rows etc.

SQL is the standard language for Relation Database System.

DBMS and RDBMS:

DBMS: Database Management System is where the data is stored in the form of Flat files and having a Parent Child relationship. It's not that in DBMS,the data cannot be stored in tables,but it is that even though the data is stored in tables,it will not have any relation between them

RDBMS: Relational Database Management System is where the data is always stored in the form of tables.The table is a collection of related data entries and it consists of columns and rows.

DDL Commands:

Command	Description
CREATE	Creates a new table, a view of a table, or other object in database
ALTER	Modifies an existing database object, such as a table.
DROP	Deletes an entire table, a view of a table or other object in the database.

DML Commands:

Command	Description

INSERT	Creates a record
UPDATE	Modifies records
DELETE	Deletes records

DQL Commands:

Command	Description
SELECT	Selects a dataset based on the conditions

4.1 DDL Commands: Data Definition Language (DDL) is a part of SQL that is used to create, modify, and delete database objects such as table, view, and index. Below are the most common DDL commands:

4.1.1 CREATE TABLE: Every data is stored in a table in the database. The way we do it is by CREATE TABLE statement.A table is made up of rows and columns. Each row represents one piece of data, and each column can be thought of as representing a component of that piece of data.

The Syntax for creating the table is as below:

```
CREATE TABLE <TABLE_NAME>
(<COLUMN 1><DATATYPE>CONSTRAINTS,
(<COLUMN 1><DATATYPE>CONSTRAINTS,
... )
```

Below is the Example:

```
CREATE TABLE Customer
(Customer_Idint,
First Name char(50),
Last Name char(50),
Address char(50),
City char(50),
Country char(25,
Birth Datedatetime);
```

Six types of constraints can be placed when creating a table:

- **NOT NULL Constraint**: **Ensures** that a column cannot have NULL value.
- **DEFAULT Constraint:** Provides a default value for a column when none is specified.
- **UNIQUE Constraint**: `Ensures` that all values in a column are different.
- **CHECK Constraint:** Makes sure that all values in a column satisfy certain criteria.
- **Primary Key Constraint:** Used to uniquely identify a row in the table.
- **Foreign Key Constraint:** Used to ensure referential integrity of the data.

4.1.2 CREATE VIEW: View is a virtual table.A view consists of rows and columns just like a table. The difference between a view and a table is that views are built on top of other tables and do not hold data themselves. If data is changing in the underlying table, the same changes are reflected in the view. A view can be built on top of a single table or multiple tables.

The Syntax for Creating a View is as below:

CREATE VIEW <VIEW NAME>
AS SELECT * FROM <TABLENAME>

Below is the Example:

```
CREATE VIEW Customer_VW
As
Select * from Customer
```

4.1.3 CREATE INDEX: The INDEX is used to create and retrieve data from the database very quickly. Index can be created by using single or group of columns in a table. When index is created, it is assigned a ROWID for each row before it sorts out the data. Proper indexes are good for performance in large databases, but you need to be careful while creating index. Selection of fields depends on what you are using in your SQL queries.

The Syntax for Creating and Index is as below:

CREATE INDEX<VIEW NAME>
ON <TABLE_NAME>(COLUMN_NAME1,COLUMN_NAME2,…)

Below is the Example:

CREATE INDEX idx1 on Customer(birth_date)

4.1.4 ALTER TABLE: The ALTER TABLE command is used to add, delete or modify columns in an existing table. You would also use ALTER TABLE command to add and drop various constraints on an existing table.

The constraints that are explained above can also be used for creating as well for altering.

The Syntax for Alter Table is as below:

ALTER TABLE <TABLE_NAME> [ADD|DROP|MODIFY] COLUMN_NAME [DATATYPE]

Below is the Example:

ALTER TABLE Customer ADD Age int
ALTER TABLE Customer DROP COLUMN Age
ALTER TABLE Customer MODIFY COLUMN Age varchar(2)
The same ADD/DROP/MODIFY commandsare used to modify the constraints that are required

4.1.5 ALTER VIEW: The ALTER VIEW command is used to alter the query that was used to create a view

The Syntax for Alter View is as below:

ALTER VIEW <VIEW_NAME>As
SELECT QUERY ON TABLES

Below is the Example:

ALTER VIEW as Select Customer_id,Age from Customer

4.1.6 ALTER INDEX: You cannot alter the index by using any command.All you have to do is to drop the index first and then recreate the index based on the requirement

4.1.7 DROP TABLE: Sometimes we need to clear off the table from the database where it is not necessary.This is when we need to get rid of the table from the database which helps DBA in maintenance activities.

The Syntax for DROP TABLE is as below:

DROP TABLE<TABLENAME>

Below is the Example:

DROP TABLE Customer

DROP TABLE Customer

4.1.8 DROP VIEW: When you have a view in the database, you need a way to drop the view if it is no longer needed.

The syntax is as given below:

DROP VIEW <VIEW_NAME>

Below is the Example:

DROP VIEW Customer_VW

4.1.9 DROP INDEX: An index can be dropped if you don't require the index on a particular column or if you want to alter the index. Care should be taken when dropping an index because performance may be slowed or improved.

The syntax is as given below:

DROP INDEX <INDEX_NAME>

Below is the Example:

```
ALTER TABLE Customer
DROP INDEX idx1
```

4.2 DML Commands: Data Manipulation Language (DML) is used for managing data in the database. These commands can be rolled back. But in SQL they are committed unless you write them in a transaction.

4.2.1 INSERT: This command is used to insert the data into the table.

The syntax for the INSERT command is as below:

INSERT INTO <TABLENAME>VALUES (Value1,Value2…etc)

Below is the Example:

INSERT INTO CUSTOMER values('Rahul','A','ADDRESS','HYDERABAD','INDIA',1985-01-01')

4.2.2 UPDATE:

This command updates the already existing rows in the table.

The Syntax for the Update command is as below:

UPDATE <TABLENAME> set <COLUMN>=Value

Below is the Example:

UPDATE CUSTOMER SET CUSTOMER_NAME='RAJU' WHERE CUSTOMER_NO=1001

4.2.3 DELETE: This command deletes the rows from the table.

Syntax for this command is as below:

```
DELETE FROM <TABLENAME>
```

Below is the Example:

DELETE FROM CUSTOMER

4.2.4 TRUNCATE TABLE: Sometimes we need to delete all the data in the table when we don't need the particular data.

Syntax for TRUNCATE TABLE is as below:

TRUNCATE TABLE<TABLENAME>

Below is the Example:

TRUNCATE TABLE Customer

Difference between DELETE and TRUNCATE:

Both of these statements deletes the rows in the table but Delete can also use with the condition where it can delete some rows that satisfies the condition.

But there is one more difference in these two statements where the DELETE requires more system resources, and hence takes longer to complete, because the RDBMS has to record all changes one row at a time in the transaction log, while a TRUNCATE TABLE operation does not record the change one row at a time, so it can be completed quicker.

Difference between DROP and TRUNCATE:

DROP TABLE will delete the table from the database which means the table will not be physically existing in the database whereas TRUNCATE will delete all the rows in the table and table will physically exist in the database.

4.2.5 Except: This is used tofind the difference between the dataset in between two tables.

The Syntax is as below:

Select <Columns>from table_name
EXCEPT
Select <Columns>from table_name

Below is the Example:

Select * from Customer

Except
Select * from Employee

4.3 DQL Commands: Data Query Language (DQL) is used for selecting data from the database. This command is used to select the data from the tables that are present on the Database even by putting the conditions.

4.3.1 SELECT:

This command is used to select the data from one or more tables.

The syntax for the SELECT command is as below:

SELECT <COLUMNS_NAME1>,….<COLUMN_NAMEn>
FROM
<TABLE_NAME>

Below is the Example:

SELECT * from Customer
SELECT Customer_Id,FirstName,Last Name from Customer

The Select Statement can have many clauses that can be used to fetch the correct data for the Requirement.

4.3.2 DISTINCT Clause: This Clause as the name itself gives the distinctvalues from the Table in the Select statement.

The syntax is as below:

SELECT DISTINCT column1, column2....columnNFROMtable_name

Below is the Example:

SELECT DISTINCT First Name,LastName from Customer

4.3.3 WHERE Clause: This Clause is used for fetching the data based on a certain condition.

The Syntax is as below:

SELECT column1, column2....columnNFROMtable_name WHERE CONDITION

Below is the Example:

SELECT Customer_Id,First Name,LastName from Customer where Age>18

There can also be n number of conditions that can be Specified in the where clause where the data set that comes out as an output satisfies all the conditions OR a certain number of Conditionswhen we use a AND /OR clause in between the conditions

4.3.4 IN Clause: This Clause is used to fetch the dataset from the Where clause where the data set contains the data present in the IN Clause.

The Syntax is as below:

SELECT column1, column2....columnNFROMtable_name
WHERE column_name IN (val-1, val-2,...val-N)

Below is the Example:

SELECT Customer_Id,First Name,LastName from Customer where Age IN (18,20,21)

4.3.5 BETWEEN Clause: This Clause is used to fetch the dataset from the Where clause where the Data output is in between the given Values inclusive of the Start and the End value.

The Syntax is as below:

SELECT column1, column2....columnNFROMtable_name
WHERE column_name BETWEEN val-1 AND val-2

Below is the Example:

SELECT Customer_Id,First Name,LastName from Customer where Age Between 18 and 23

4.3.6 LIKEClause: This Clause is used to fetch the dataset from the Where clause where the Data output likely matches with the given format.This need not be the exact match.

The Syntax is as below:
SELECT column1, column2....columnNFROMtable_name
WHERE column_name LIKE { PATTERN }

Below is the Example:

SELECT Customer_Id,First Name,LastName from Customer where First Name like '%Ram%'

4.3.7 ORDER BY Clause: This Clause is used to sort the dataset either in the Ascending order or on the Descending order..If No order is specified,then its Ascending, If DESC is specified ,its Descending order which means by default its Ascending.

The Syntax is as below:

```
SELECT column1, column2....columnNFROMtable_name
WHERE CONDITION
ORDER BY column_name {ASC|DESC}
```

Below is the Example:

```
SELECT Customer_Id,First Name,LastName from Customer Order by Customer_ID
SELECT Customer_Id,First Name,LastName from Customer Order by Customer_IDdesc
```

4.3.8 GROUP BY Clause: This Clause is used to group a particular dataset based on the requirement to find the statistics.

The Syntax is as below:

SELECT SUM(column_name) FROM table_nameWHERE CONDITION GROUP BY column_name;

Below is the Example:

```
SELECT SUM(marks) from Student group by student_name
SELECT SUM(marks) from Student  where roll_no between 1 and 5 group by student_name
```

4.3.9 COUNT Clause: This Clause is used to give the count of the dataset that satisfies the condition.

The Syntax is as below:

```
SELECT COUNT(column_name) FROM table_name WHERE CONDITION
```

Below is the Example:

```
SELECT COUNT(*) from Customer
```

4.3.10 HAVING Clause: This Clause is used to satisfy the condition after grouping the data where the condition is dependent on the grouping.

The Syntax is as below:

```
SELECT SUM(column_name) FROM table_name WHERE CONDITION
GROUP BY column_nameHAVING (arithematic function condition)
```

Below is the Example:

```
SELECT SUM(marks) from Student where roll_no between 1 and 5 group by student_name
having SUM(Marks)>150
```

Main Concepts of the Group By and Having Clause:

- To use Group By Clause, we need to use at least one aggregate function.
- All columns that are not used by aggregate function(s) must be in the Group By list.
- We can use Group By Clause with or without Where Clause.
- To use Having Clause, we have to use Group By Clause since it filters data that we get from Group By Clause.

4.4 Joins:

SQL Joins are used to relate information in different tables. A Join condition is a part of the SQL query that retrieves rows from two or more tables. A SQL Join condition is used in the SQL WHERE Clause of select, update, delete statements.

A SQL join is used to combine rows from two or more tables, based on a common field between them and it can actually perform this for more tables as well.

The Syntax for Joins is as below:

```
SELECT col1, col2, col3...
FROM table_name1, table_name2
WHERE table_name1.col2 = table_name2.col1
```

The Syntax can also be in the below format as well:

```
SELECT col1, col2, col3...
FROM table_name1 join table_name2
on ( table_name1.col2 = table_name2.col1)
```

Types of Joins that are used in SQL:

- Inner Join
- Left Join
- Outer Join
- Full Outer Join

The join will result in a Cartesian product if the joining does not be on the Columns.Let us now take an example and then see how the Joins represent the data.

Customer Table:

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000
2	Khilan	25	Delhi	1500
3	kaushik	23	Kota	2000
4	Chaitali	25	Mumbai	6500
5	Hardik	27	Bhopal	8500
6	Komal	22	MP	4500
7	Muffy	24	Indore	10000

Orders Table:

OID	DATE	CUSTOMER_ID	AMOUNT
102	2014-10-08	3	3000
100	2014-10-08	3	1500
101	2014-11-20	2	1560
103	2014-05-20	4	2060

4.4.1 Inner Join: Returns all rows when there is at least one match in both tables.

```
SELECT C.ID, O.OID, C.NAME, C.ADDRESS, O.AMOUNT
FROM CUSTOMER C INNER JOIN ORDERS O ON (C.
ID=O.CUSTOMER_ID)
```

This query will return the rows where the Id matched with the customer id in the order table.

The output of the Inner join will be as below:

ID	OID	NAME	Address	AMOUNT
3	102	kaushik	Kota	3000
3	100	kaushik	Kota	1500
2	101	Khilan	Delhi	1560
4	103	Chaitali	Mumbai	2060

4.4.2 Left Outer Join: Returns all rows from the left table, and the matched rows from the right table.

```
SELECT C.ID, O.OID, C.NAME, C.ADDRESS, O.AMOUNT
FROM CUSTOMER C LEFT OUTER JOIN ORDERS O ON
(C. ID=O.CUSTOMER_ID)
```

This Query returns the all the rows with the matched data of the two tables then remaining rows of the Left table and NULL for the Right tables column.

The Output of the Left Outer join is as below:

ID	OID	NAME	Address	AMOUNT
1	NULL	Ramesh	Ahmedabad	NULL
2	101	Khilan	Delhi	1560
3	102	kaushik	Kota	3000
3	100	kaushik	Kota	1500
4	103	Chaitali	Mumbai	2060
5	NULL	Hardik	Bhopal	NULL

6	NULL	Komal	MP	NULL
7	NULL	Muffy	Indore	NULL

4.4.3 Right Outer Join: Returns all rows from the Right table, and the matched rows from the left table.

```
SELECT C.ID, O.OID, C.NAME, C.ADDRESS, O.AMOUNT
FROM CUSTOMER C RIGHT OUTER JOIN ORDERS O
ON (C. ID=O.CUSTOMER_ID)
```

This Query returns the all the rows with the matched data of the two tables then remaining rows of the Right table and NULL for the Left tables column.

The Output of the Right Outer join is as below:

ID	OID	NAME	Address	AMOUNT
3	102	kaushik	Kota	3000
3	100	kaushik	Kota	1500
2	101	Khilan	Delhi	1560
4	103	Chaitali	Mumbai	2060

4.4.4 Full Outer Join: Returns rows when there is a match in one of the tables.

```
SELECT C.ID, O.OID, C.NAME, C.ADDRESS, O.AMOUNT
FROM CUSTOMER C FULL OUTER JOIN ORDERS O ON
(C. ID=O.CUSTOMER_ID)
```

This Query returns the all the rows with the matched data of the two tables then remaining rows of the Left table and Right table .

The Output of the Full Outer join is as below:

ID	OID	NAME	Address	AMOUNT
1	NULL	Ramesh	Ahmedabad	NULL
2	101	Khilan	Delhi	1560
3	102	kaushik	Kota	3000
3	100	kaushik	Kota	1500

4	103	Chaitali	Mumbai	2060
5	NULL	Hardik	Bhopal	NULL
6	NULL	Komal	MP	NULL
7	NULL	Muffy	Indore	NULL

4.5 Functions:

SQL provides many in-built functions to perform operations on data. These in-built functions are defined into 2 subcategories:

4.5.1 Aggregate Functions: These are purely mathematical functions which return a single value from a group of values.

Ex:Avg,count,Min,Maxetc

4.5.2 Scalar Valued Functions: These functions return a single value from an Input value.

E.g :Ucase,Lcase,Roundetc

4.5.3 User Defined Functions: These functions allow you to define our own functions using the below syntax.

```
CREATE FUNCTION<Function name>
(Input value1, Input value2…)
Returns (Output value1, output value2….)
As
Begin
Statements
End
```

Advantages of functions are as below:

- They allow modular programming which means you can create them once, stored in Database, call it number of times, modify independently of program source code.
- Allows faster Execution which means functions reduce the compilation cost by caching the plans and reusing them for repeated executions. Functions need not be reparsed and re-optimized which gives faster execution times.

- Reduces the Network Traffic which means that if an operation that filters data based in some complex constraint that cannot be expressed as a function. The Function can then invoke in where clause to reduce the no of rows sent to the client.

Ex:

```
CREATE FUNCTION Total Salary
  (@Commission decimal (5,2), @Salary decimal(5,2))
RETURNS decimal (12,3)
AS
BEGIN
  RETURN (@Commission*@Salary)
END
```

This function can then be used anywhere an integer expression is allowed, such as in a computed column for a table:

```
CREATE TABLE Employee
  (
Employeeidint PRIMARY KEY,
Name     varchar(20),
DesignationVarchar(50),
Commission decimal(4,1),
Salarydecimal(4,1),
Total Salary AS
          (
dbo. Total Salary (Commission,Salary )
          )
  )
```

The above example gives an output that the total salary is calculated based on the Commission and the salary an employee is getting.

This is a very simple example of how to use the functions. Usually Functions should be used when there is a reuse of code which actually improves the Performance of the module by reusing the code and also helps in decreasing the number of efforts that are put on.

4.5.6 Cursors: Usually SQL queries handles large amount of data and there will be cases where the data should be handled for each row and this can happen only by the use of Cursor.

The Cursor usually fetches every row and then does the operation for that row and then goes for fetching the next row.

The Syntax for this Cursor is as below:

```
DECLARE <Cursor name > CURSOR
FOR <Select statement>--decides the rows where the cursor needs the handle data
OPEN <Cursor name> ---open the cursor for each row
FETCH NEXT FROM <Cursor name>; ---to fetch each row from the cursor
```

Below is the Example:

```
DECLARE DB_SUCCESS CURSOR FOR

select ACCT_NO,CUSTOMER_NO,DB_DATA from [dbo].[DB_INBOUND_SINGLETABLE_TEMP] nolock
where [DATAIDENTIFIER]='2QCB'

OPEN DB_SUCCESS

FETCH next FROM DB_SUCCESS INTO @KY_BA,@CUST_NO,@DB_DATA

WHILE @@fetch_status = 0

        BEGIN

                        IF EXISTS(SELECT * FROM CRA_BANKRUPTCY nolock WHERE ACCT_NO=@KY_BA
AND [NO_OF_BANKRUPTCIES_LAST_6YRS]=SUBSTRING(@DB_DATA,13,2) AND CURRENT_FLAG='Y' )
                        BEGIN

                        UPDATE CRA_BANKRUPTCY SET

                        END_DT=GETDATE(),MODIFIED_DT=GETDATE() WHERE ACCT_NO=@KY_BA AND
CURRENT_FLAG='Y'
                        END
                        ELSE
                        BEGIN
                        UPDATE CRA_BANKRUPTCY SET CURRENT_FLAG='N' WHERE ACCT_NO=@KY_BA
AND CURRENT_FLAG='Y'
                        INSERT INTO
CRA_BANKRUPTCY(ACCT_NO,CUSTOMER_NO,NO_OF_BANKRUPTCIES_LAST_6YRS,START_DT,END_DT,C
URRENT_FLAG,INSERTED_DT)

        VALUES(@KY_BA,@CUST_NO,SUBSTRING(@DB_DATA,13,2),GETDATE(),GETDATE(),'Y',GETDATE(
))

                        END


        FETCH next FROM DB_SUCCESS INTO @KY_BA,@CUST_NO,@DB_DATA


END
CLOSE DB_SUCCESS
 DEALLOCATE DB_SUCCESS
```

4.6 Procedures: This is a collection of DML statements,functions and cursors which can perform all the actions at one time. It can also accept the Parameters and provide the result set if necessary.

The advantages of Procedures are as follows:

- Security due to Encryption.
- Performance gain due to compilation.
- Being able to hold the code in the central repository which means Altering code in SQL server without altering in several places and will be able to keep the statistics on the code to be optimized.
- Reduction in the amount of data passed over a network by keeping code in the server.
- Hiding the raw data by allowing only stored procedures to gain access to the data.

There are 2 types of stored procedures.

User defined: These are defined by the developer based on the requirement.

System-Stored: These are defined by the SQL server which is readily used if needed.

The Syntax for this Procedure is as below:

```
CREATE PROCEDURE <Procedure Name> (
Input parameters, Output Parameters (If required))
AS
BEGIN
<Set of SQL statements>
END
```

Below is the Example:

We will now see the above cursor used in this procedure

```
CREATE PROCEDURE [dbo].[DB_BANKRUPTCY]
AS
BEGIN
SET NOCOUNT ON;

DECLARE @KY_BA bigint
DECLARE @CUST_NO BIGINT
DECLARE @DB_DATA VARCHAR(MAX)

DECLARE DB_SUCCESS CURSOR FOR

select ACCT_NO,CUSTOMER_NO,DB_DATA from [dbo].[DB_INBOUND_SINGLETABLE_TEMP] nolock
where [DATAIDENTIFIER]='2QCB' and acct_no<148868400

OPEN DB_SUCCESS

FETCH next FROM DB_SUCCESS INTO @KY_BA,@CUST_NO,@DB_DATA

WHILE @@fetch_status = 0

        BEGIN

                        IF EXISTS(SELECT * FROM CRA_BANKRUPTCY nolock  WHERE ACCT_NO=@KY_BA
AND [NO_OF_BANKRUPTCIES_LAST_6YRS]=SUBSTRING(@DB_DATA,13,2) AND CURRENT_FLAG='Y' )
                        BEGIN

                        UPDATE CRA_BANKRUPTCY SET

                        END_DT=GETDATE(),MODIFIED_DT=GETDATE() WHERE ACCT_NO=@KY_BA AND
CURRENT_FLAG='Y'
                        END
                        ELSE
                        BEGIN
                        UPDATE CRA_BANKRUPTCY SET CURRENT_FLAG='N' WHERE ACCT_NO=@KY_BA
AND CURRENT_FLAG='Y'
                        INSERT INTO
CRA_BANKRUPTCY(ACCT_NO,CUSTOMER_NO,NO_OF_BANKRUPTCIES_LAST_6YRS,START_DT,END_DT,C
URRENT_FLAG,INSERTED_DT)

        VALUES(@KY_BA,@CUST_NO,SUBSTRING(@DB_DATA,13,2),GETDATE(),GETDATE(),'Y',GETDATE(
))
                        END

        FETCH next FROM DB_SUCCESS INTO @KY_BA,@CUST_NO,@DB_DATA
END
CLOSE DB_SUCCESS
 DEALLOCATE DB_SUCCESS
 END
```

Difference between Procedure and a Function:

- Procedures can return zero or n values whereas functions should at least return one value which is mandatory.
- Procedures can have input or Output parameters where functions have only input parameters.
- Procedure allows select as well as DML's where function allows only select statement.
- Functions can be called in a procedure where procedure cannot be called in a function.
- Exception can be handled by Try-Catch bock in procedure where Try-Catch block cannot be used in a function.
- We can go for Traction management in procedure where this cannot go in for Function.
- Procedures cannot be used in select statement where functions can be embedded din select statement.
- User defined functions can be used in SQLstatements anywhere in Where/Select/Having where Procedures cannot.
- User Defined Functions that return tables can be treated as another row set. This can be used in joins.

SSIS

Chapter 5 - OVERVIEW of ETL:

ETL the word explains as Extraction, Transformation and Loading as the abbreviation explains.

Extraction defines the extraction of data from the sources (The sources can be either from a legacy system, a Database or through Flat files).

Transformation defines the data that is transformed as part of cleansing, aggregation and whatever the data changes might be, all the data changes are done in this Transformation step.

Loading defines the load of data from the Transformed data in to the Target Systems which are called as Destinations (The Destinations can again be either Legacy system, Database or a flat file).

The ETL tool that we are now going to concentrate is **SSIS.**

SSIS is abbreviated as SQL Server Integration Services which is a BI tool.

SSIS –SSMS (SQL Server integration Services – SQL Server Management Studio):

Let us now consider that we need to load data from a Source table from one server to a Target table in another server.

The Structure of the table is as below:

Source: Employee1

DB: EMPLOYEE

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int

5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

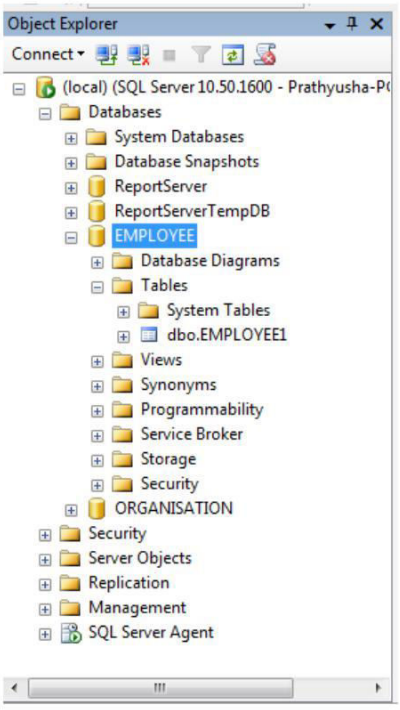
Target: Employee
DB : ORGANISATION

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

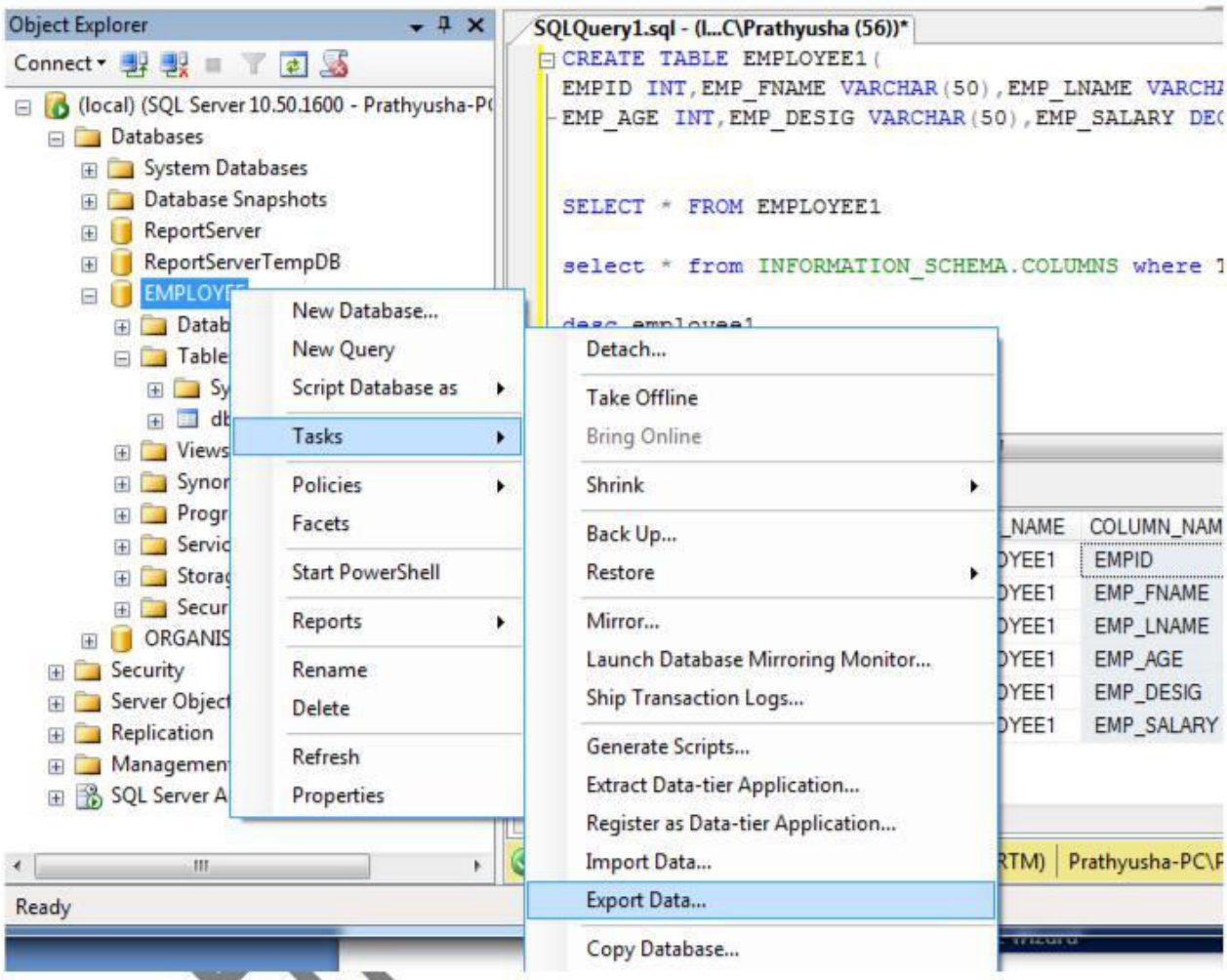
We usually do this when we need the data from one table from a source to another table in a target database.

This can be achieved in 2 ways:

Step1: SQL Server Import and Export Wizard: Locate the EMPLOYEE database in the SSMS Object Explorer.

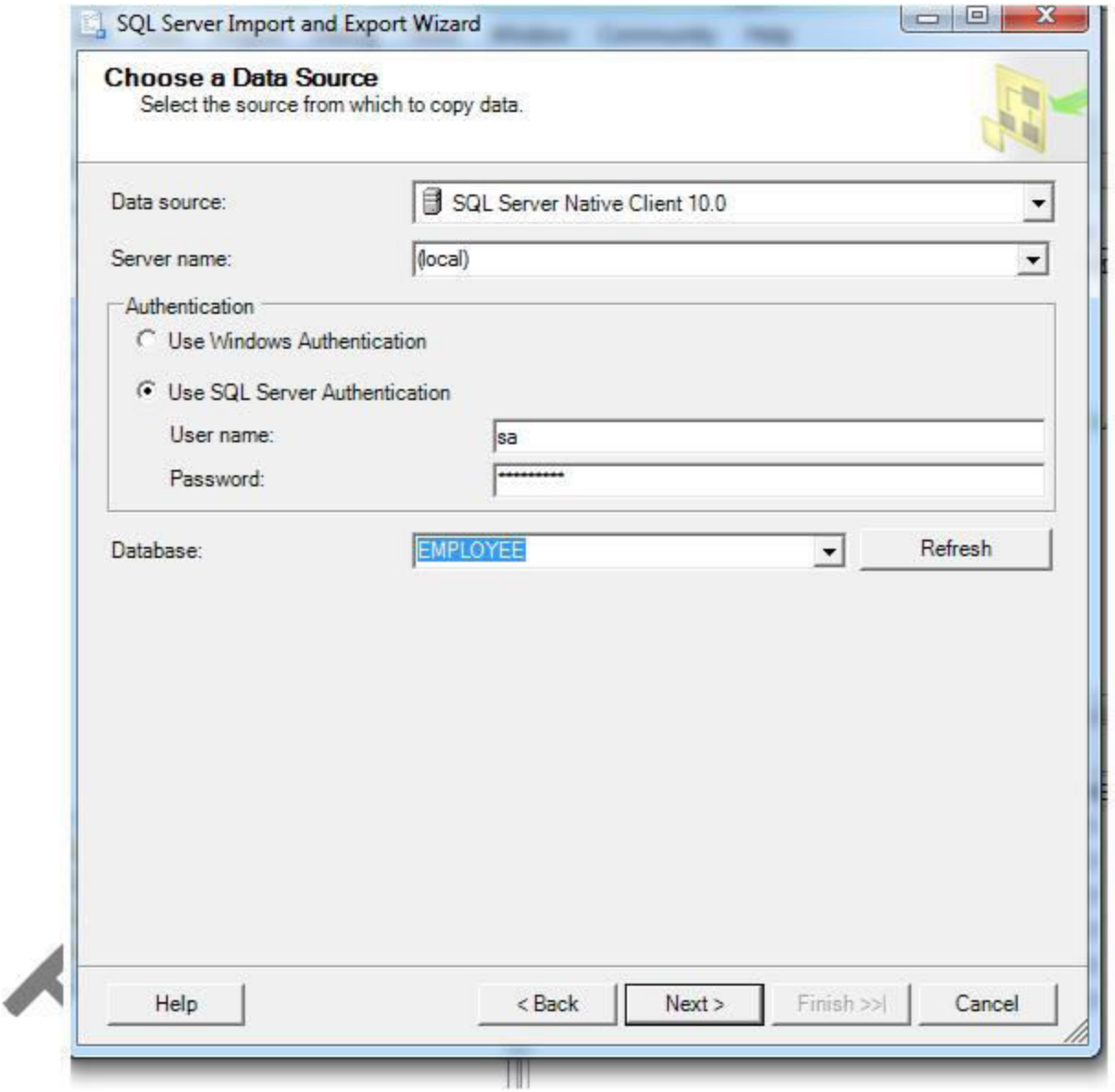


Step2: Right click on the EMPLOYEE database in the Object Explorer, select Tasks, and then Export Data from the context menu to launch the Export Wizard.



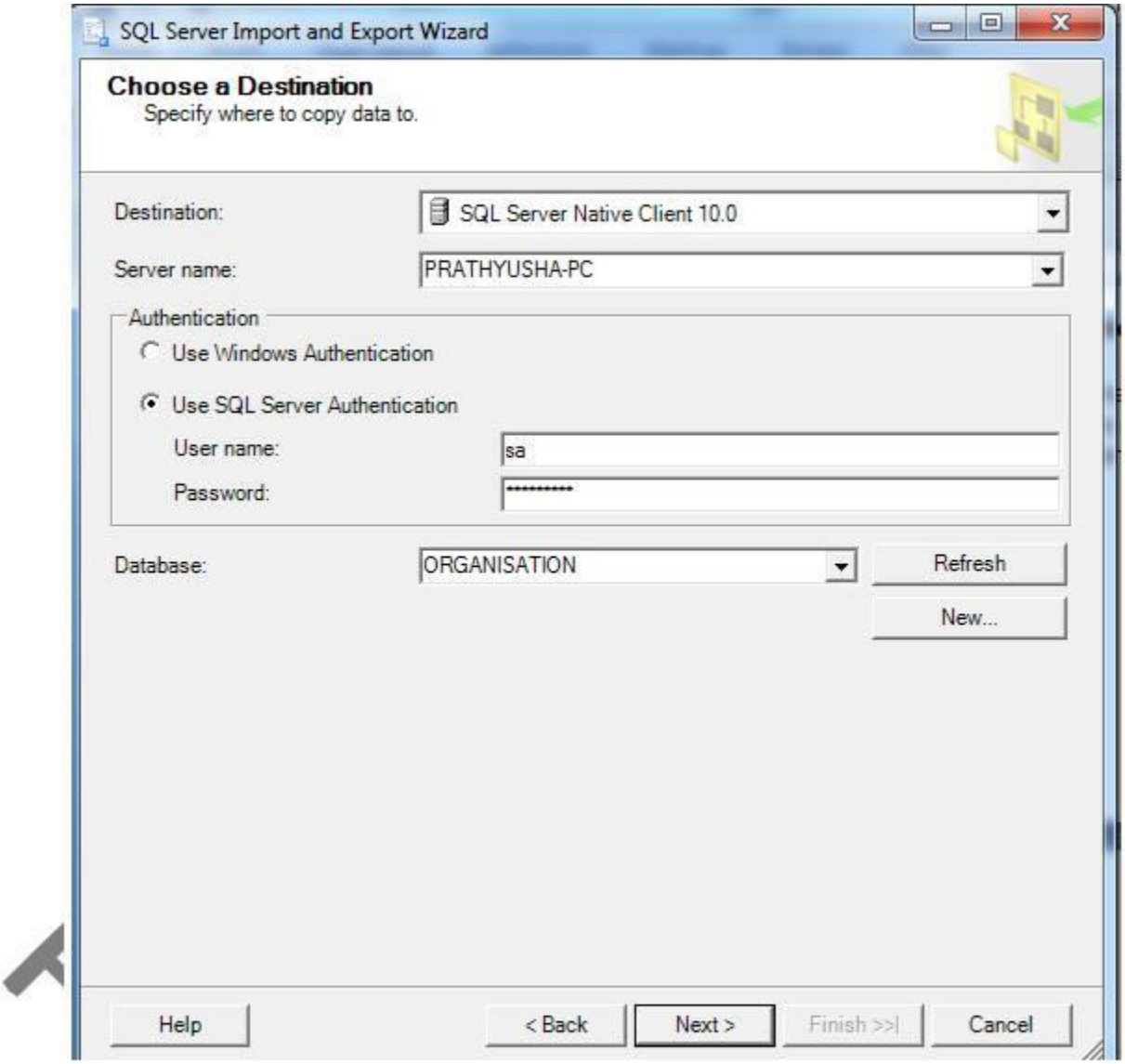
Step3: Choose a Data Source:

The Choose a Data Source dialog allows you to specify the source of your data. Here in our example, it's the EMPLOYEE DB and the server where the Source exists.



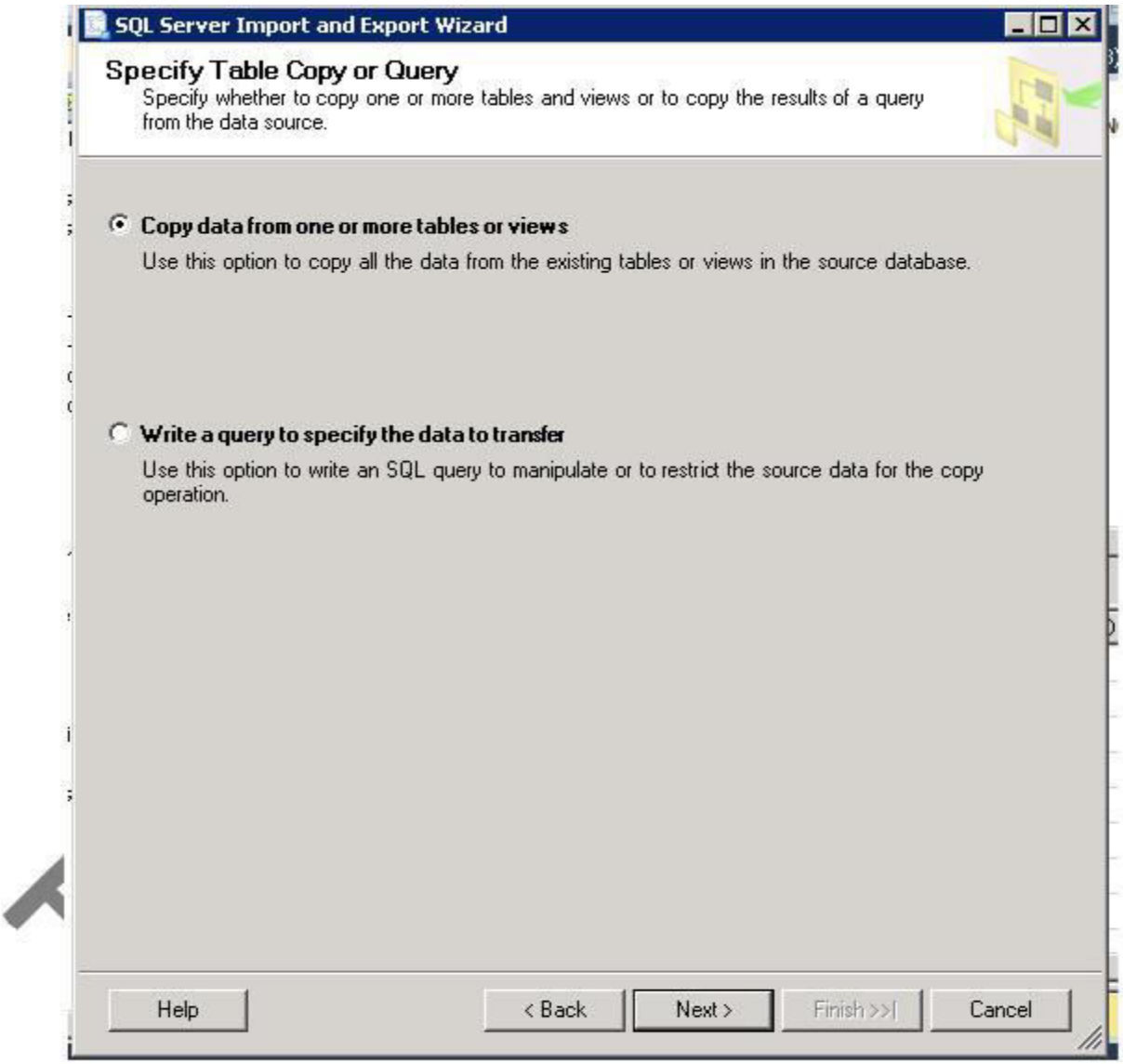
Step4: Choose a Destination:

The Choose a Destination dialog allows you to specify the destination data source for the data you are exporting. Here in our example, we will provide the Target database and the server details where the data should move on.



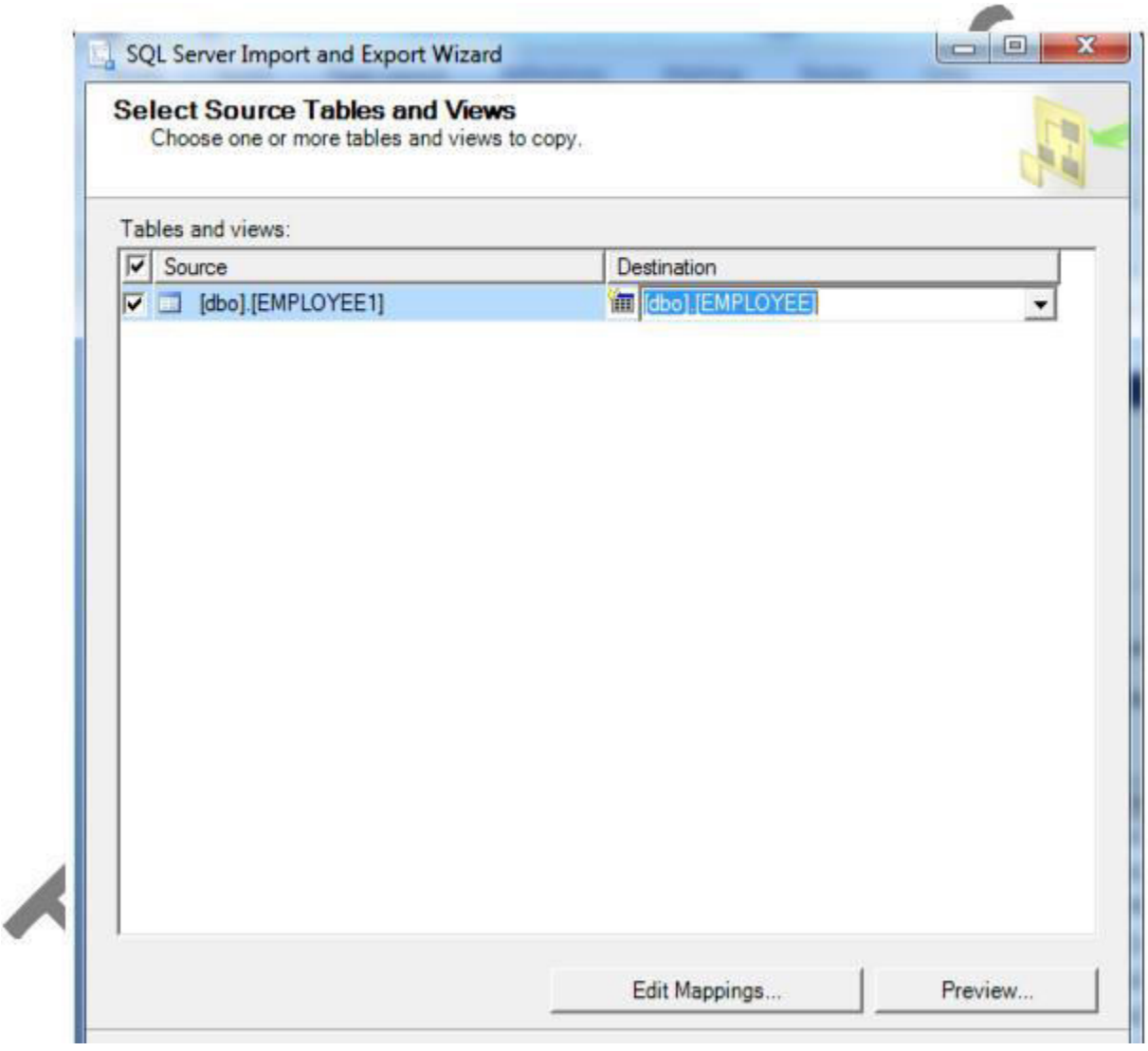
Step5: Specify Table Copy or Query:

The Specify Table Copy or Query dialog allows you to choose whether to export data by selecting tables and/or views from the data source or specifying a query to extract data. Here in our example, we will just load the data from the table to another directly.

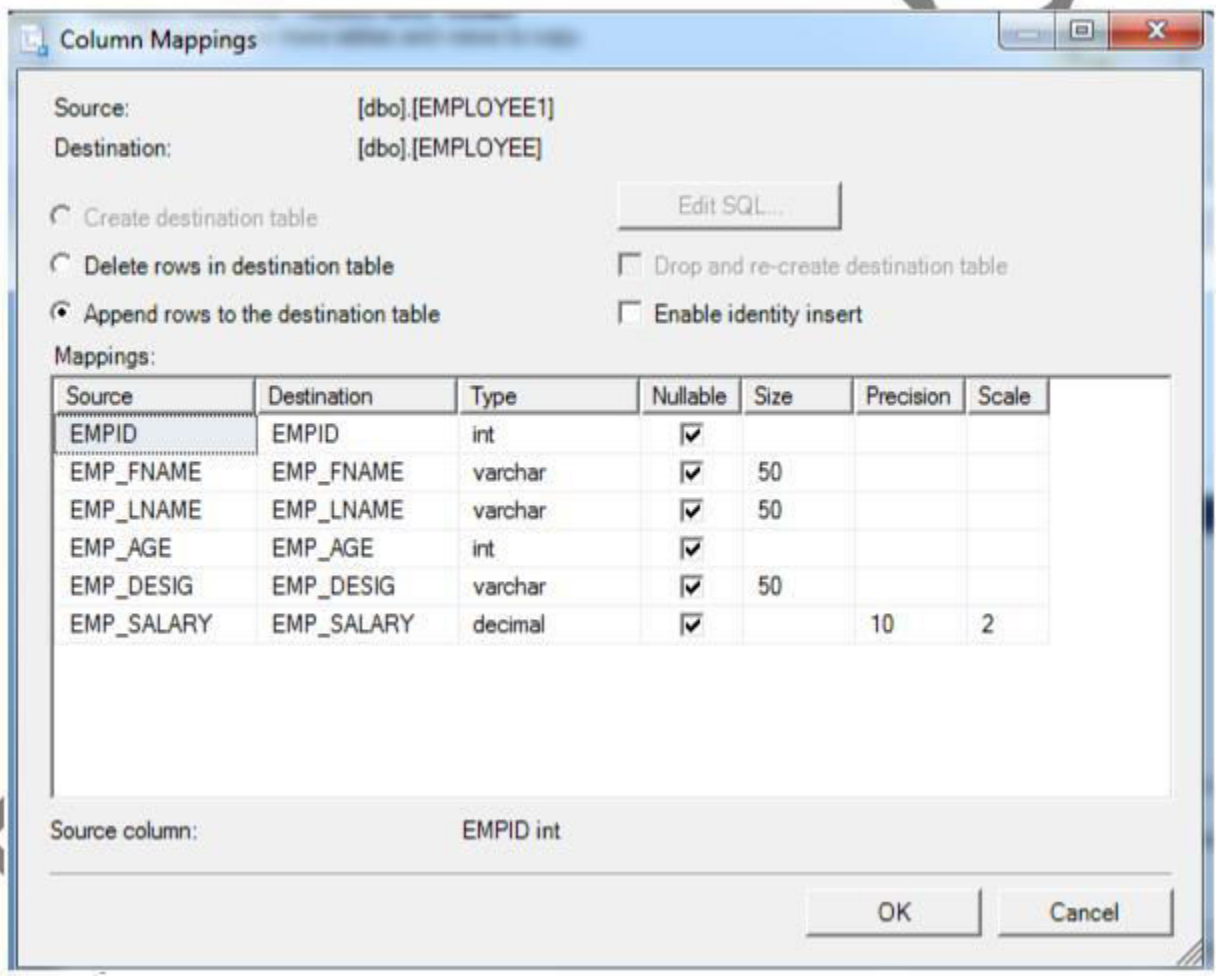


Step6: Select Source Tables and Views:

The Select Source Tables and Views dialog allows you to select the tables and views that you want to export. Here in our Example, we should select the Employee table on both the sides.

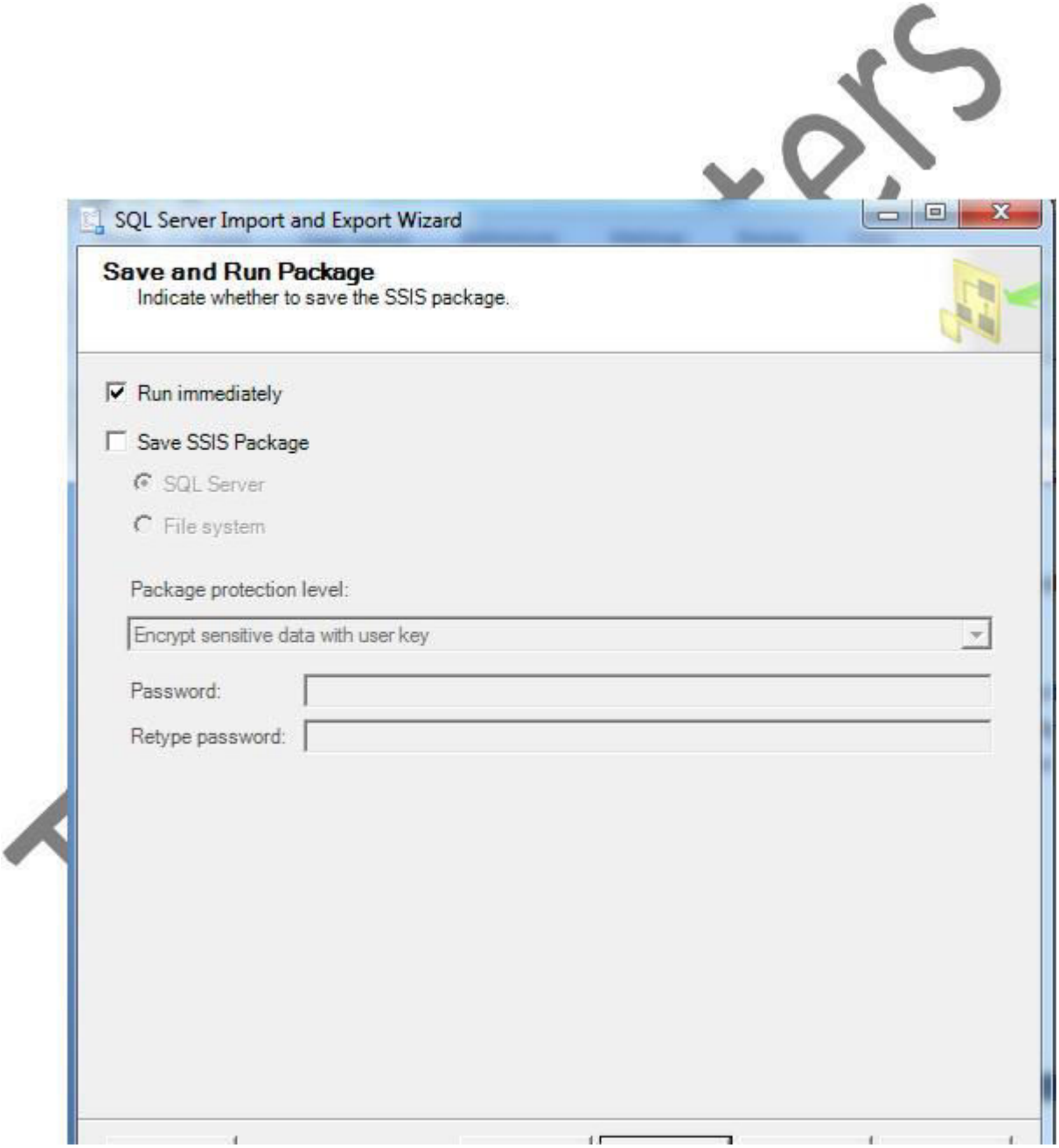


Step7: You can click the Edit Mappings button to review the column mappings from the data source to the data destination as shown below. In our example, we can actually change the mappings fields from Source to Target in the Edit Mappings dialog box.

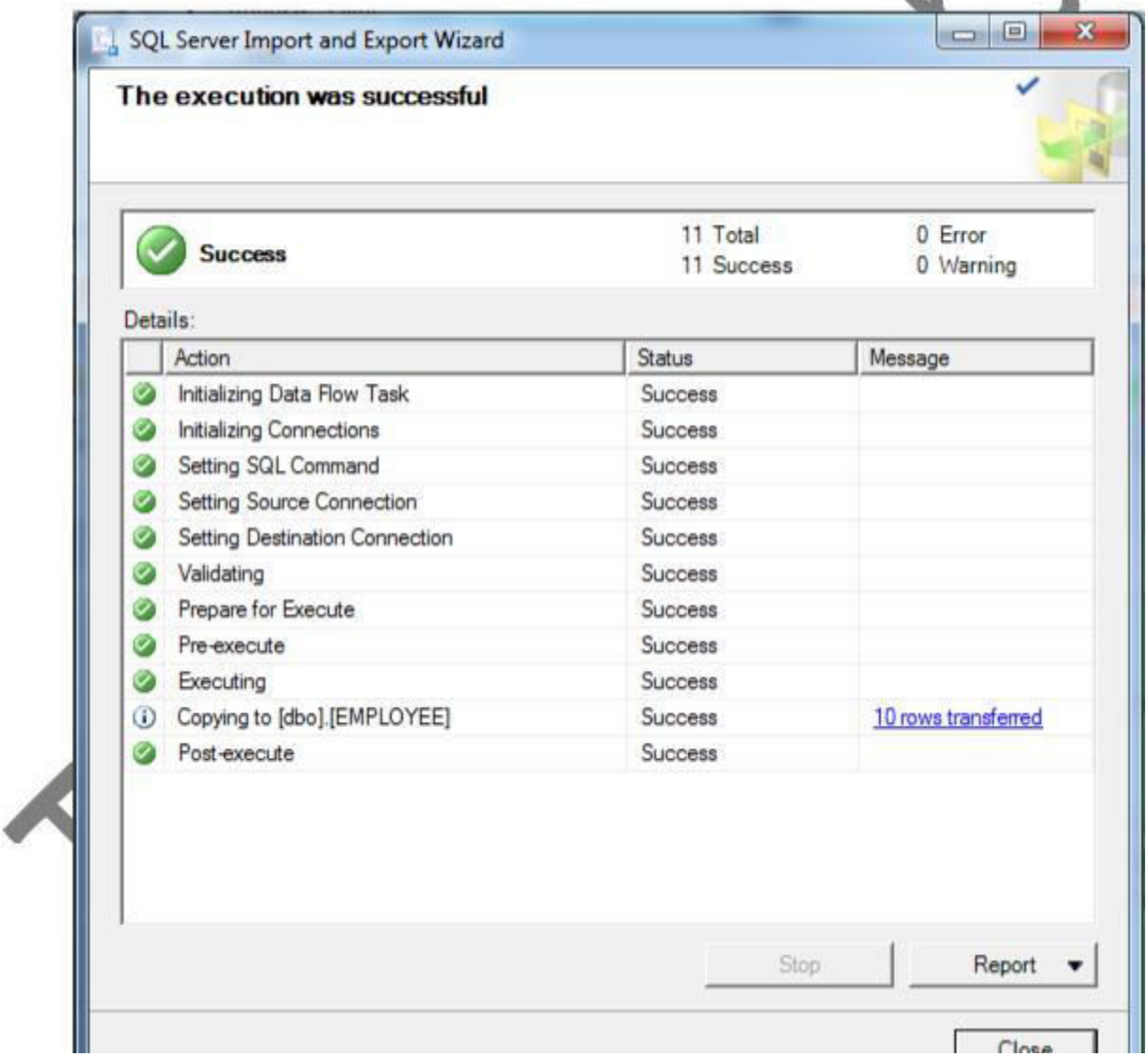


Step8: Save and Execute Package:

This allows you to save and execute the SSIS package which is created internally for loading the Data from Source to the Target table.



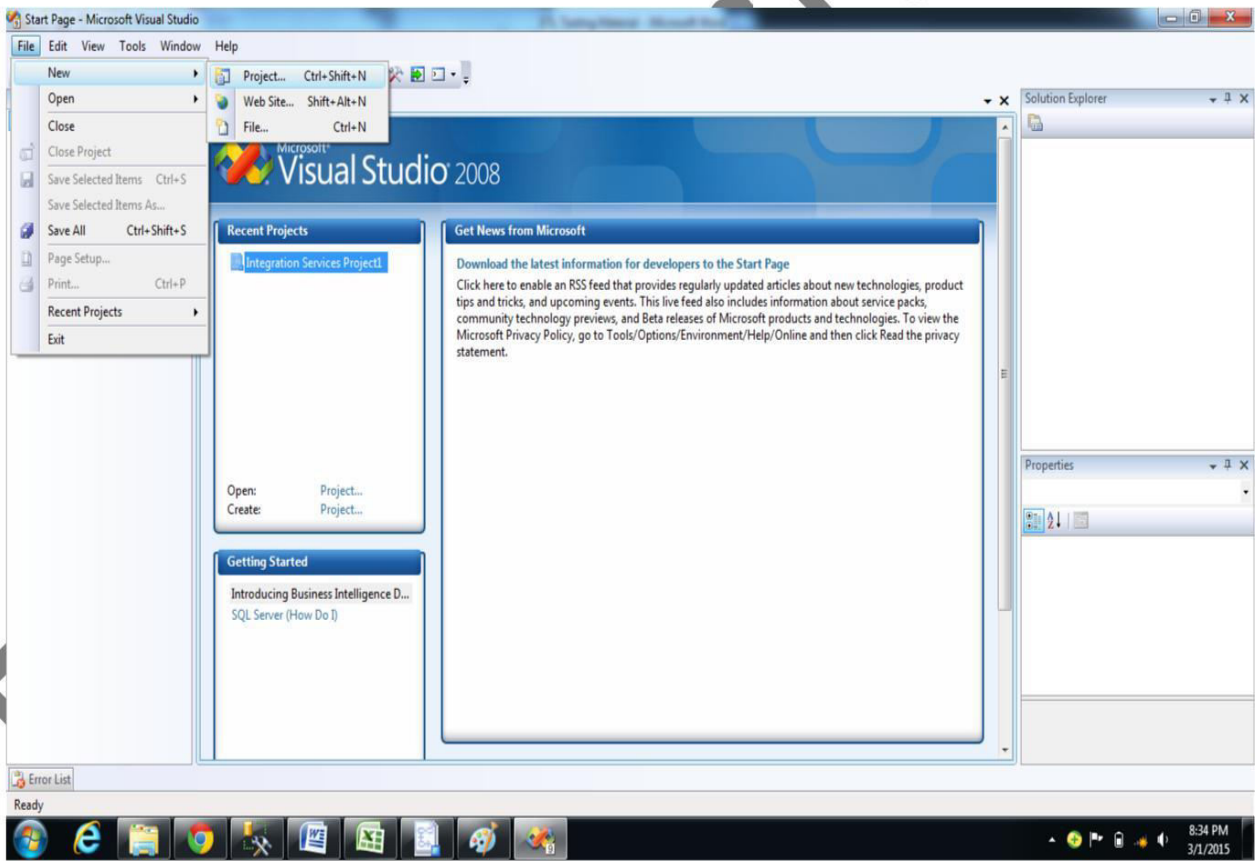
Step9: After pressing the Finish Button, the job will execute and loads the data from one table to another.



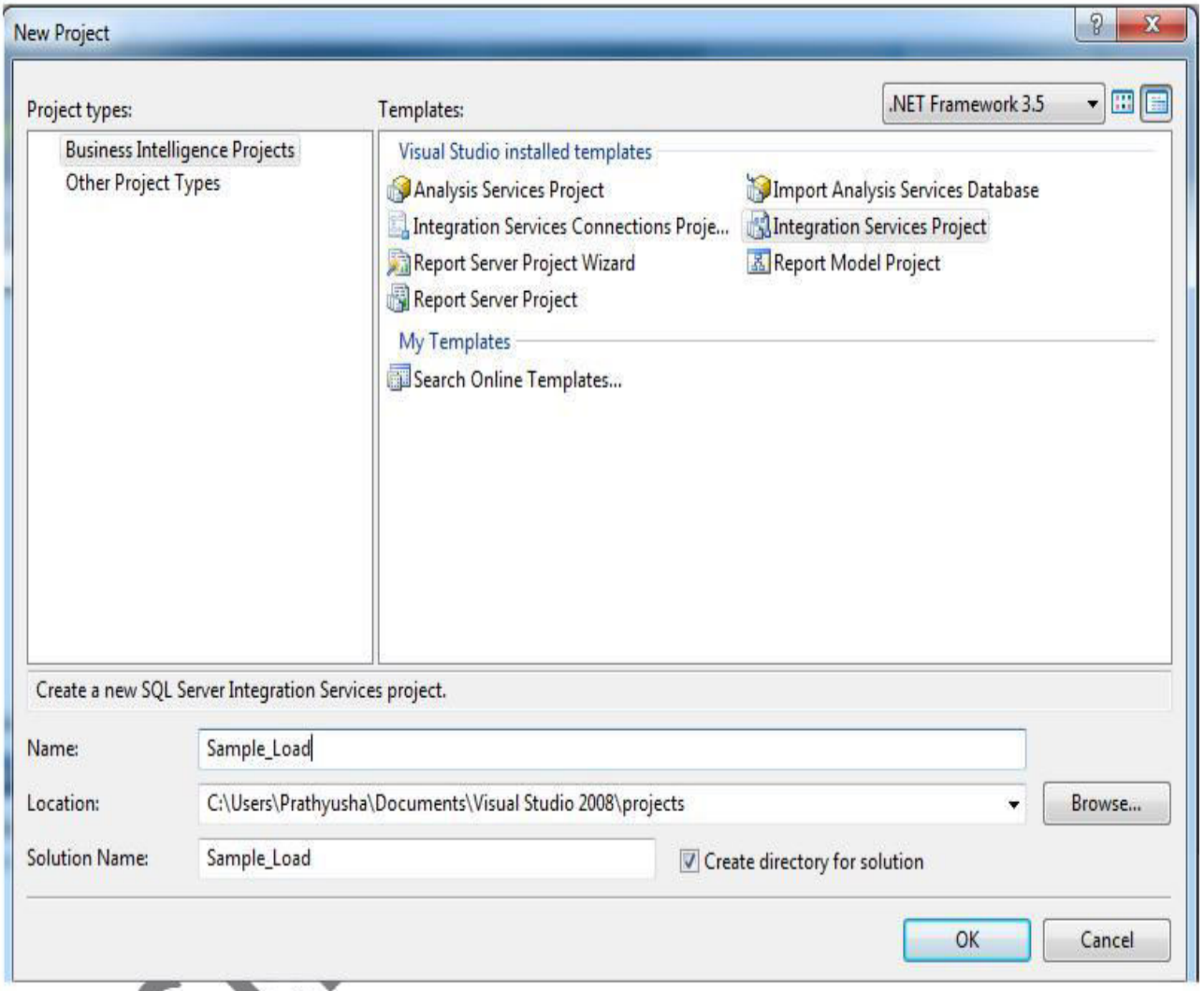
The same can be done through an SSIS package which is an ETL tool that we are going to learn now. This actually proves to be more useful when the same task needs to be repeated for a number of times.

Creating a SSIS Package of loading the data from Source to Target:

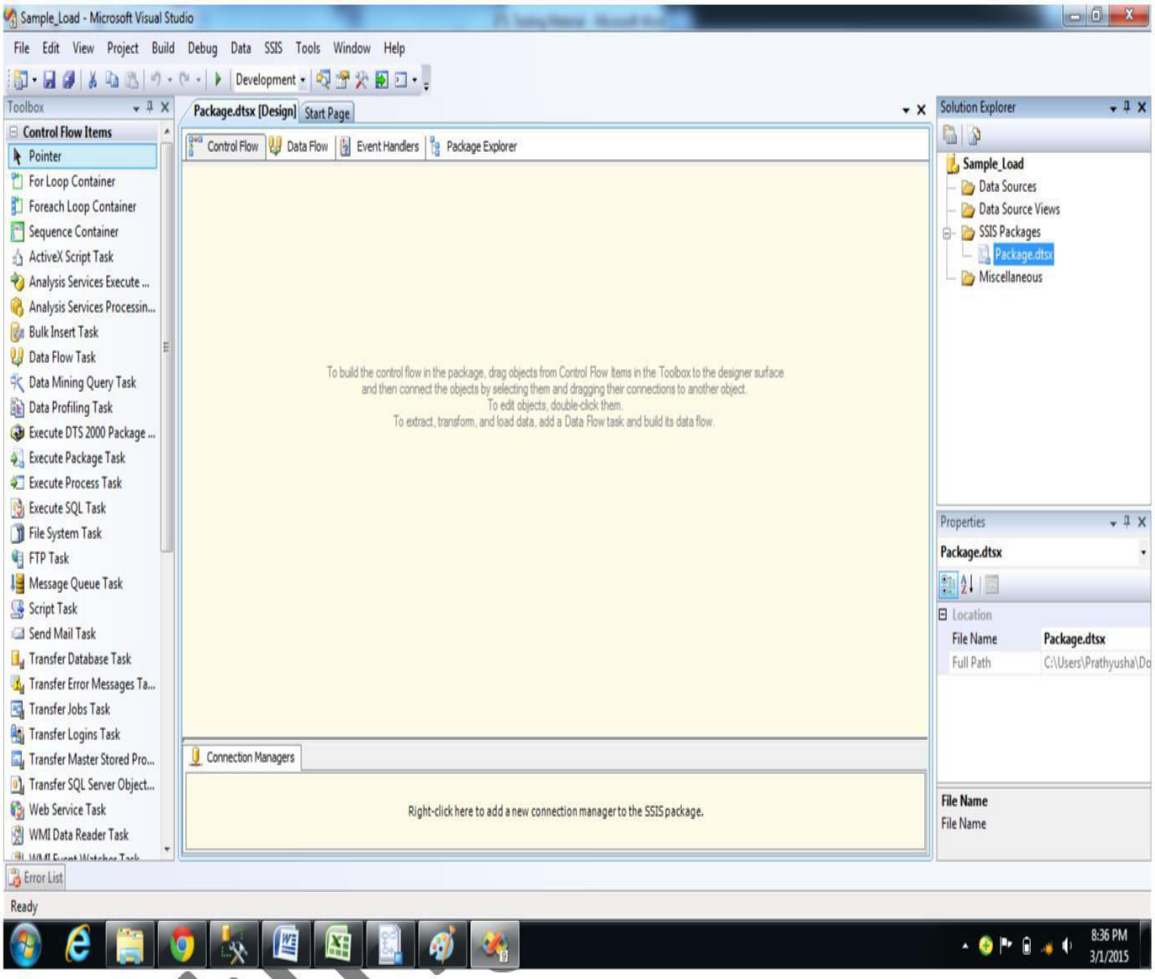
Step1: Create a new Project in the Business Intelligence Development Studio.



Step2: Select the Integration services Project and provide the project name which you want to name the project as. You can also select the path where the project wants to be saved.



Step3: Below is the screen that is opened when the project is created and we see a Tool Box at the left hand side and a Server Explorer window in the Right Hand side.

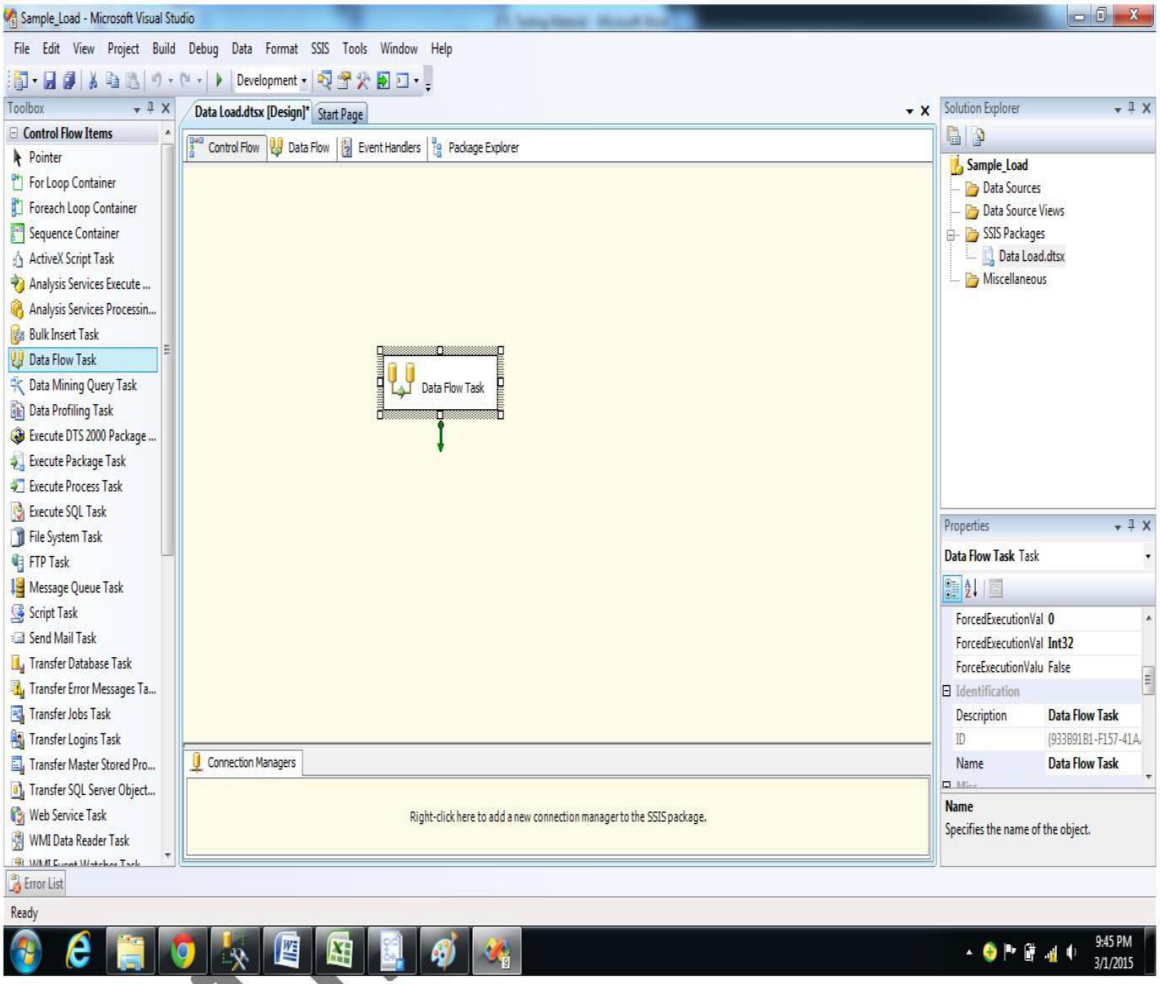


You can now see the tabs Control Flow and the Data Flow which plays a vital role in designing the package.

Control flow is where you define operations and the order of executing them.

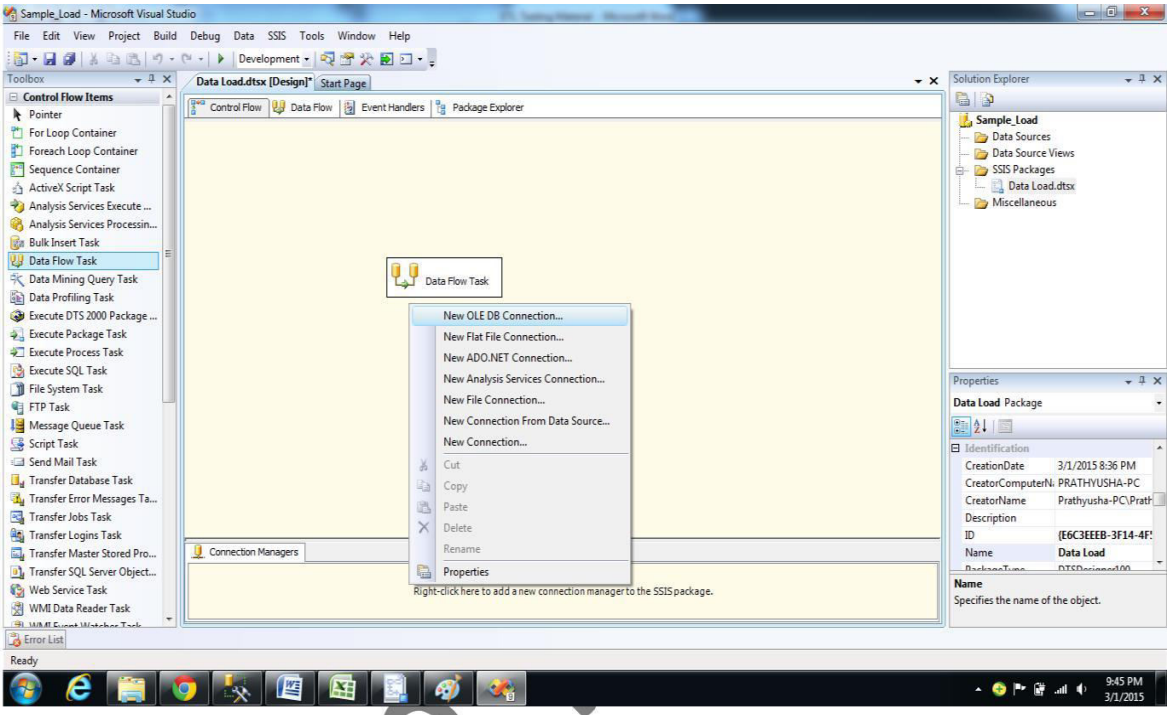
Data Flow is where you define the data stream where data comes from the data sources transforms and then loaded in the destination.

Step4: In our example, we have to use a Data Flow Task in the Control flow.



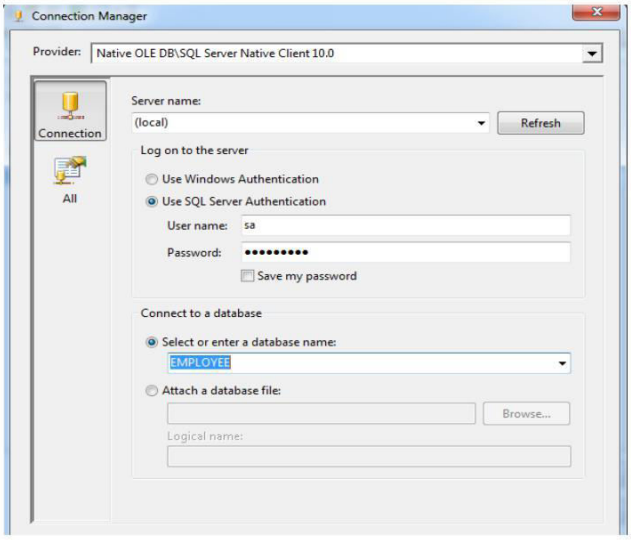
As we are dealing with the Table already existing in the database, we have to create and OLEDB connection.

Step5: In order to first access the Database we have to create a connection manager by Right clicking on the Connection Managers tab that is present below.

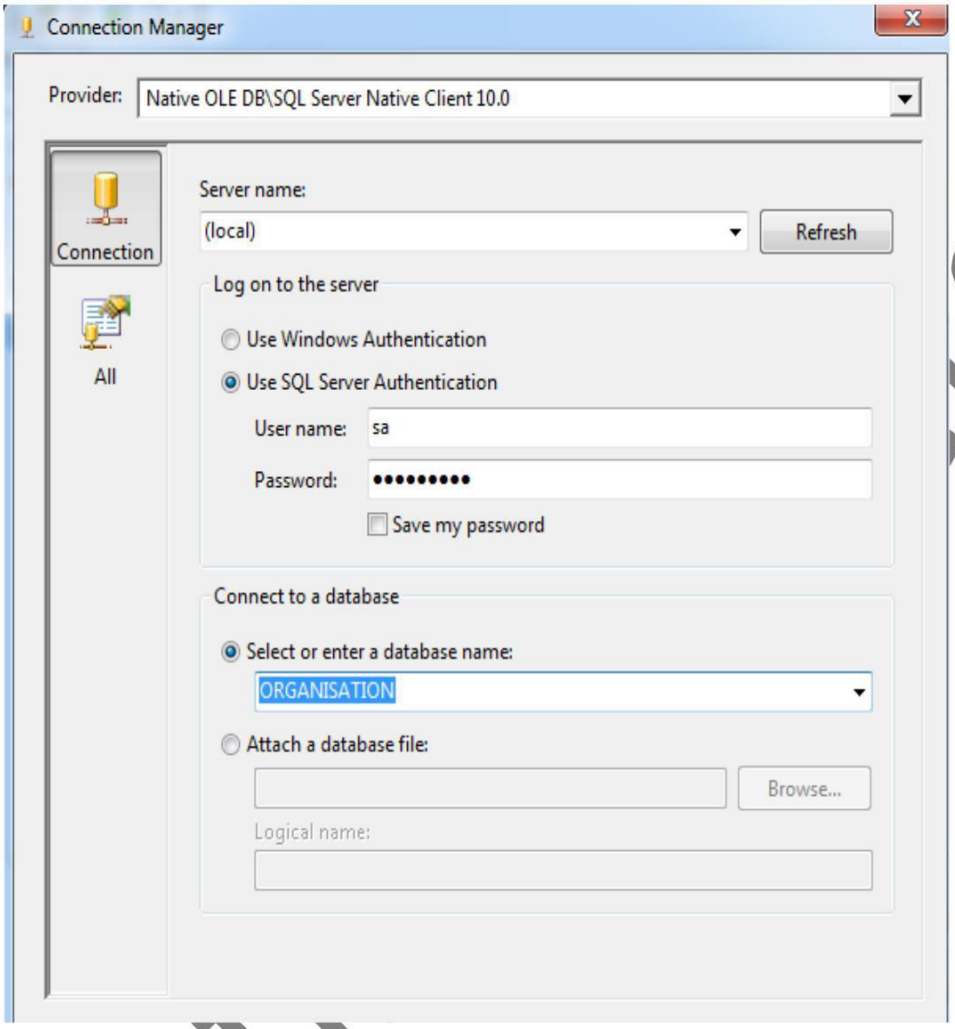


Step6: We should now create 2 Connection managers,one for the Source and one for the Destination.

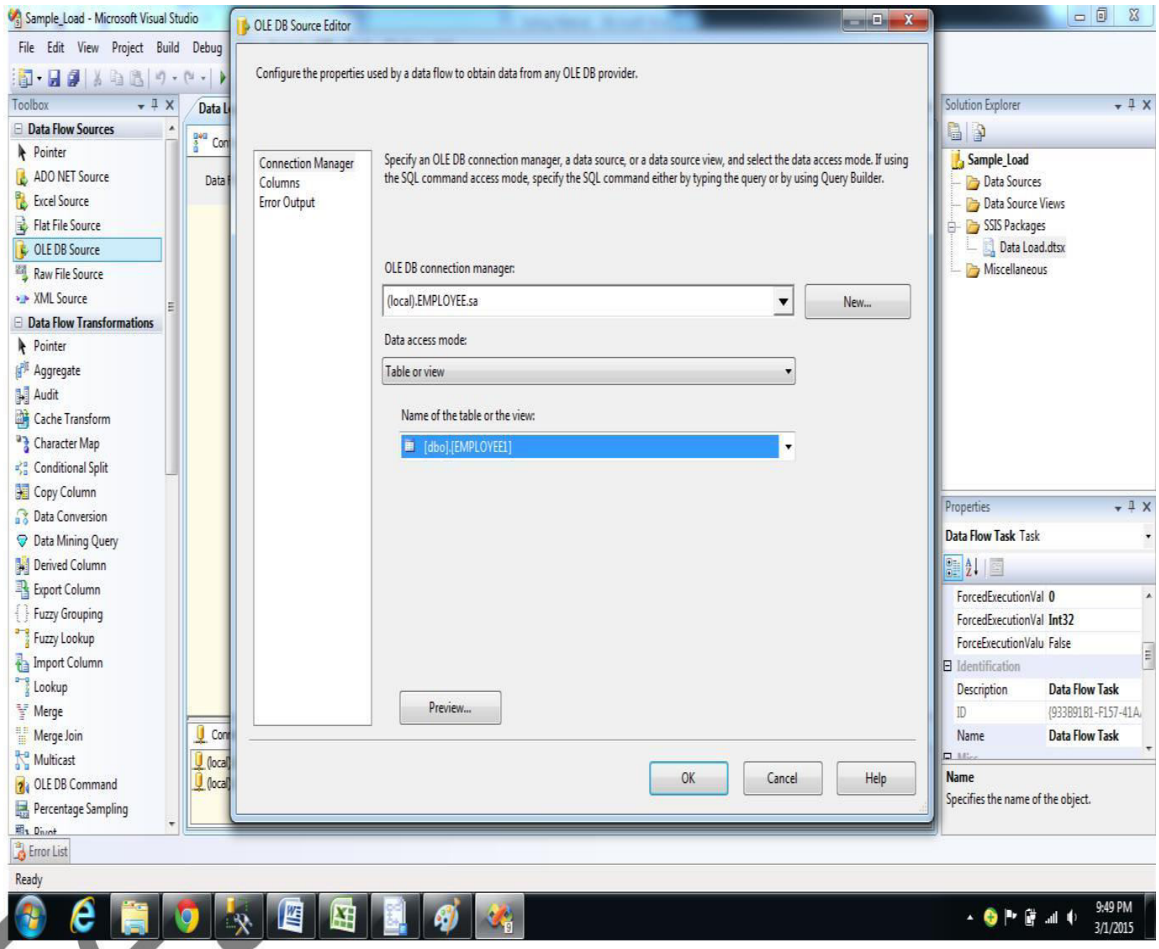
Source Connection Manager:



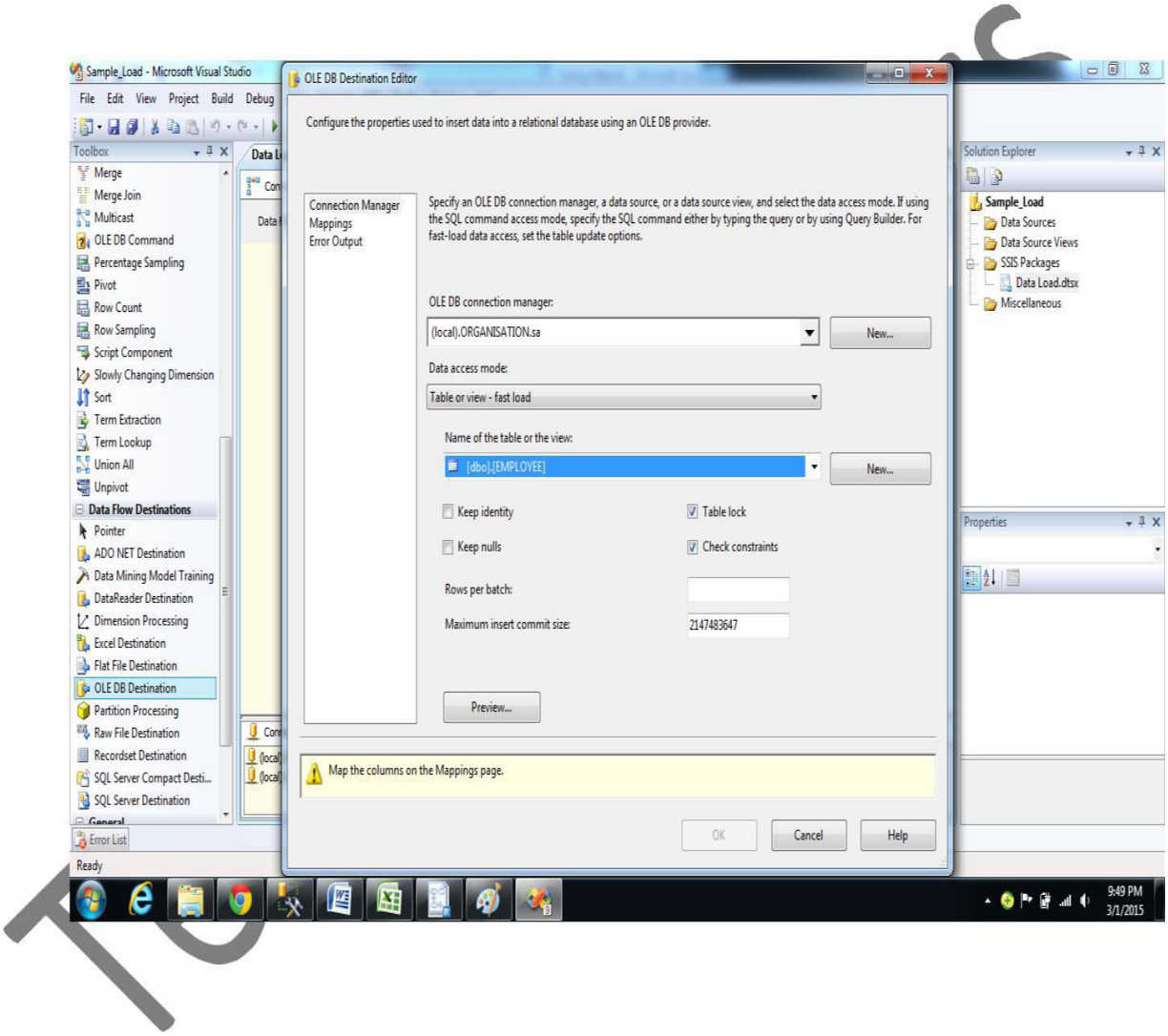
Target Connection Manager:


Connection Manager
Provider:
Native OLE DB\SQL Server Native Client 10.0
Connection
All
Server name:
(local)
Refresh
Log on to the server
Use Windows Authentication
Use SQL Server Authentication
User name:
sa
Password:
Save my password
Connect to a database
Select or enter a database name:
ORGANISATION
Attach a database file:
Browse...
Logical name:

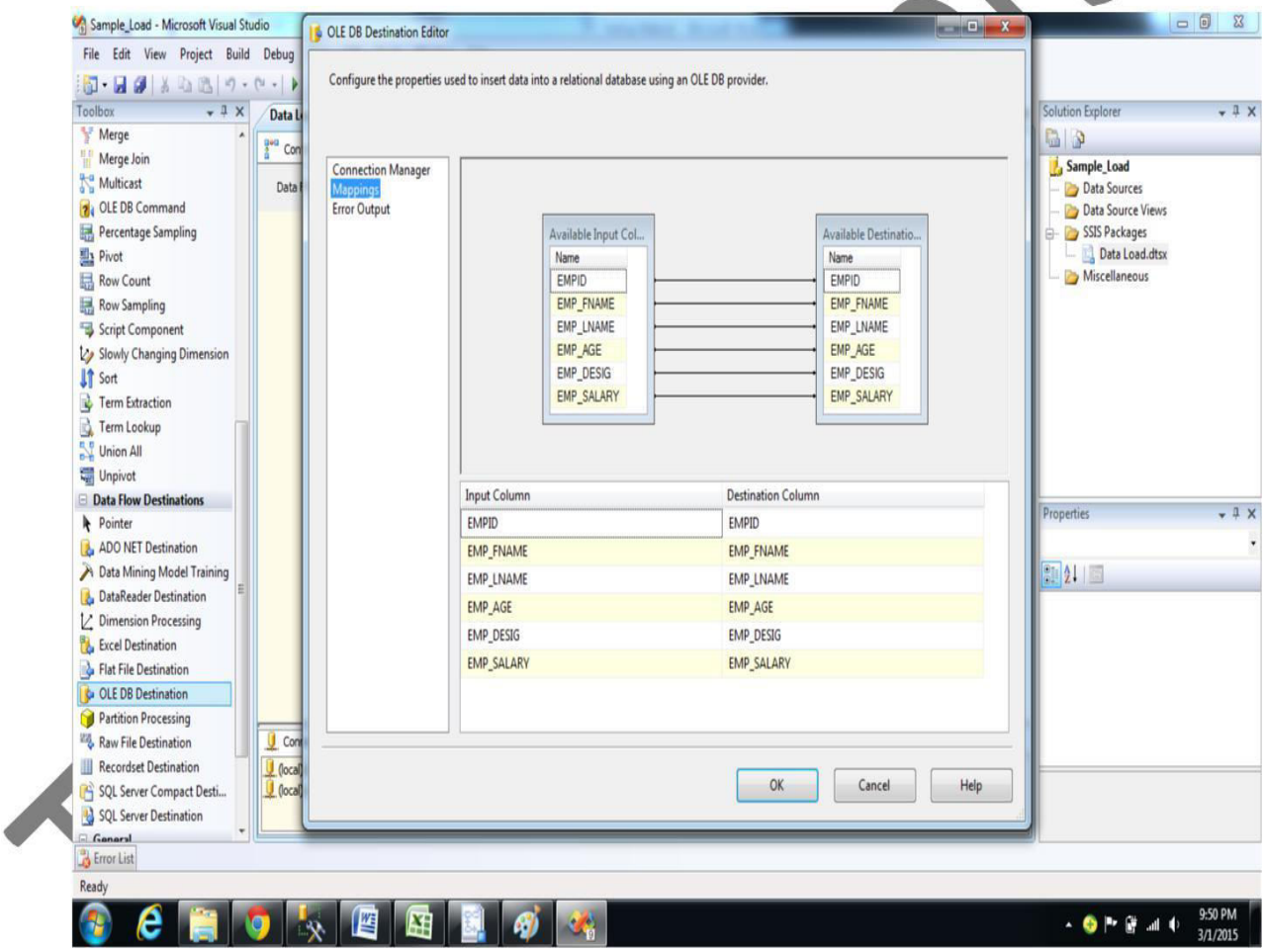
Step7: Double Click on the data flow task present in the Control flow and now drag and drop an OLEDB Source from the Left hand side Tool Box and double click on it to give the Connection Manager details as well as the Source table details.



Step8: Once completed, drag and drop an OLEDB Destination from the Destinations in the tool box and double click on it to give the Connection Manager and the Target Table details.

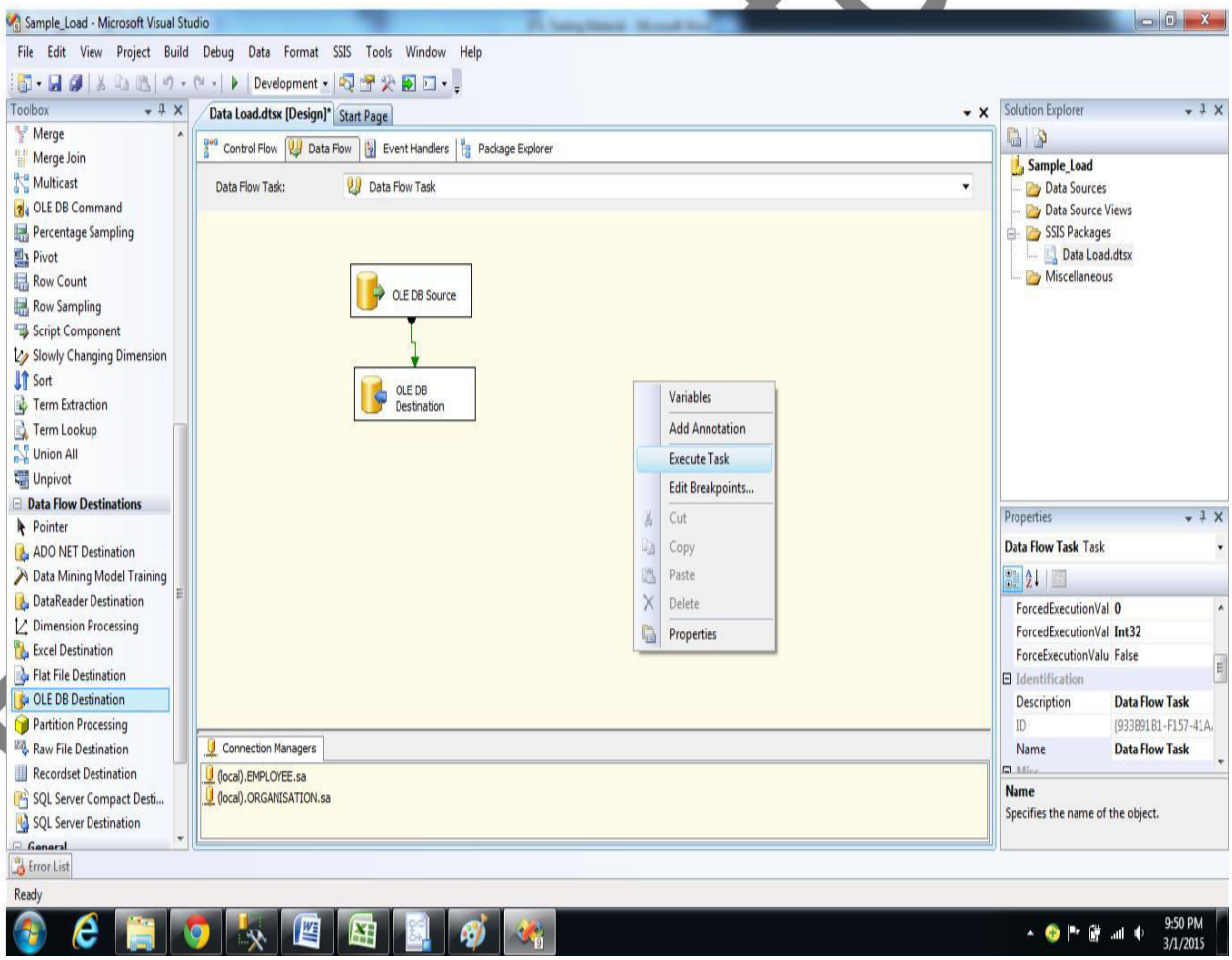


Step9: Now click on the Mappings tab present on the left side panel, to map the columns from the Source to the Target, which means that the Source column is mapped to the Target column other than the ID column, which is automatically generated where the data when executed, will be moved from the Source table to the Target table in the same column.



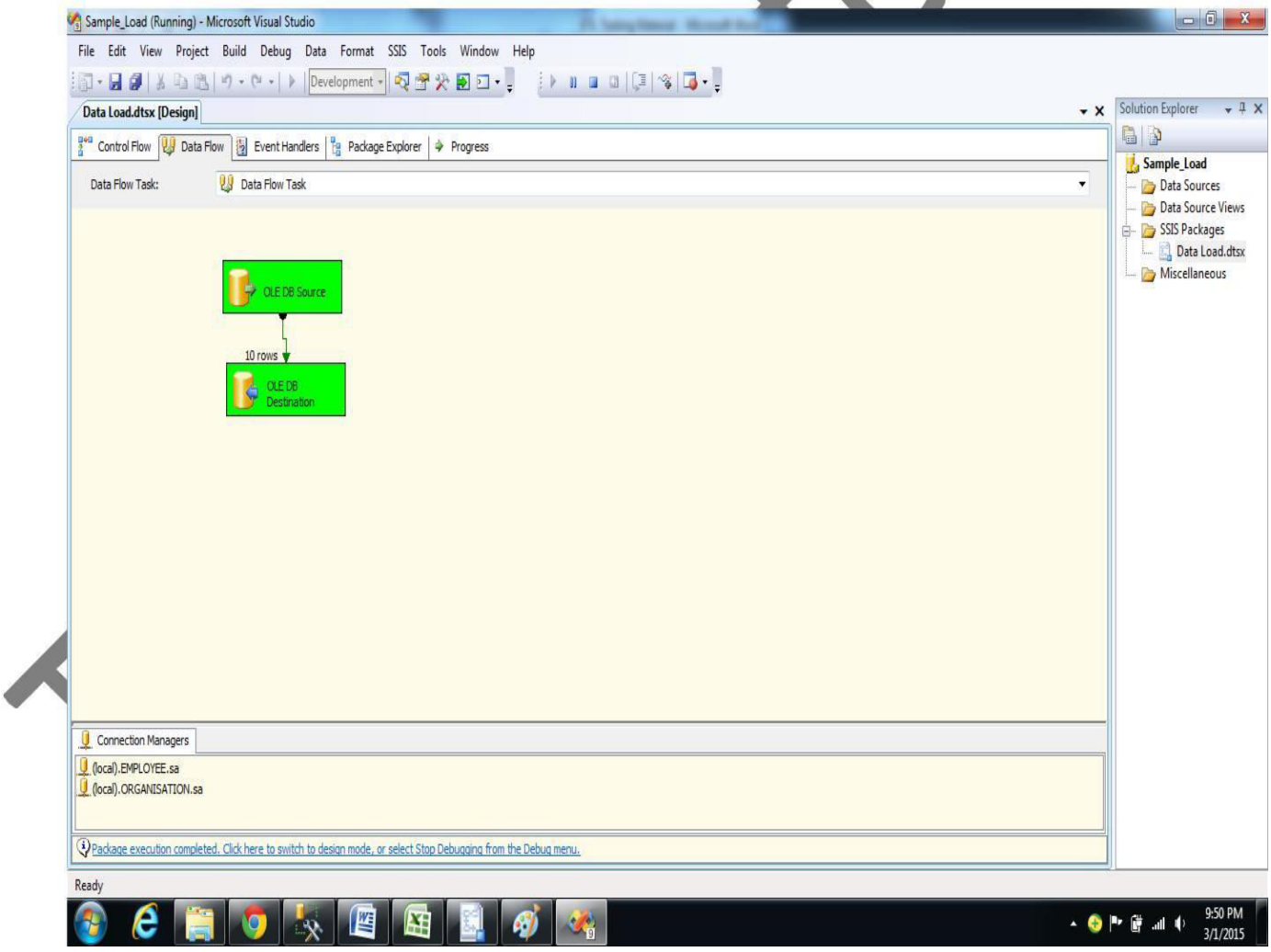
Step10: Now once the link between the Source and target has been established,Right click on the Screen to Execute the Task.

You can also execute directy by right clicking from the Package in the Right hand side if there is only one task.Otherwise all the Taks present in the Control flow will get exected here



I

Step11: After Executing the flow color changed to green which resembles that the task has been executed Successfully.



Chapter 6 - Transformations in SISS:

Transformation is a set of technical and business rules that have been extracted from the source systems and software. Business Intelligence projects present the best opportunities to remove dead and useless data to bring new light to business people information requirements. Data transformation activities should be properly implemented to produce clean, condensed, new, complete and standardized data, respectively.

Areas that are covered by Data transformation include:

Cleansing: It is by definition transformation process in which data that violates business rules is changed to conform these rules. It is usually done by ETL programs that determine or derive correct data values and then write them into the BI target databases.

Derivation: New data is created from existing (detailed) source data during this process by calculations, program logic or table lookups. Some examples of derivation may be: calculating profit from income and expense items or calculating customer's age based on their date of birth and the current year.

Aggregation: Data elements for customers may be aggregated from multiple source files and databases (e.g. Customer Master File, Sales File, and Prospect File). Sometimes (in multidimensional database terminology) this term also refers to roll-ups of data values.

Integration: The expected result of this part is to have each and unique data element known by one standard name with one standard definition and approved name. Data integration forces the need to reconcile different data names and values for the same data element. Also, each element of the data should be associated with its source databases as well as with its BI target databases. Data standardization should always be a business objective.

Diff types of Transformations:

- Sample Load from Source to Target
- Joiner Transformation
- Derived column Transformation
- Lookup Transformation
- Union Transformation
- Sorter Transformation
- Conditional Split Transformation
- Script component

6.1 Sample Load from Source to Target:

ETL Requirement: Load the Data from the Source table to the Target table in the different Database.

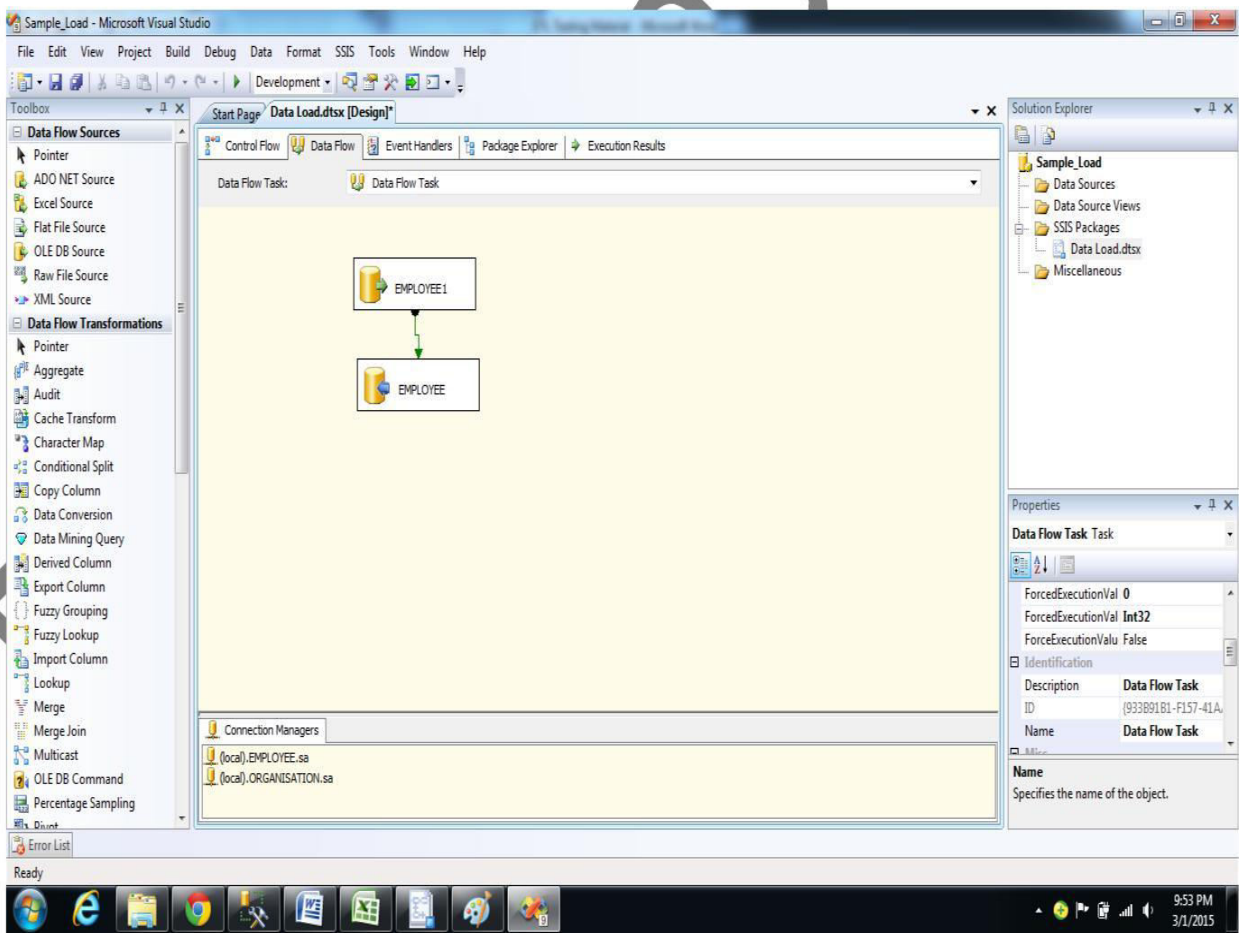
Source: Employee1
DB: EMPLOYEE

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Target: Employee
DB: ORGANISATION

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

The Below screen gives you the look of loading the data directly from Source to the Target. Follow the steps given above to create a package to do this exercise.



6.2 Sort Transformation: This Transformation is used to sort the data set from the Source.

The Sort transformation is mainly used in Merge join because the Data sets needs to sorted to be used for Merge Join.

Let us now take an example of sorting the Source dataset and loading into the Target table.

Source: Employee1
DB: EMPLOYEE

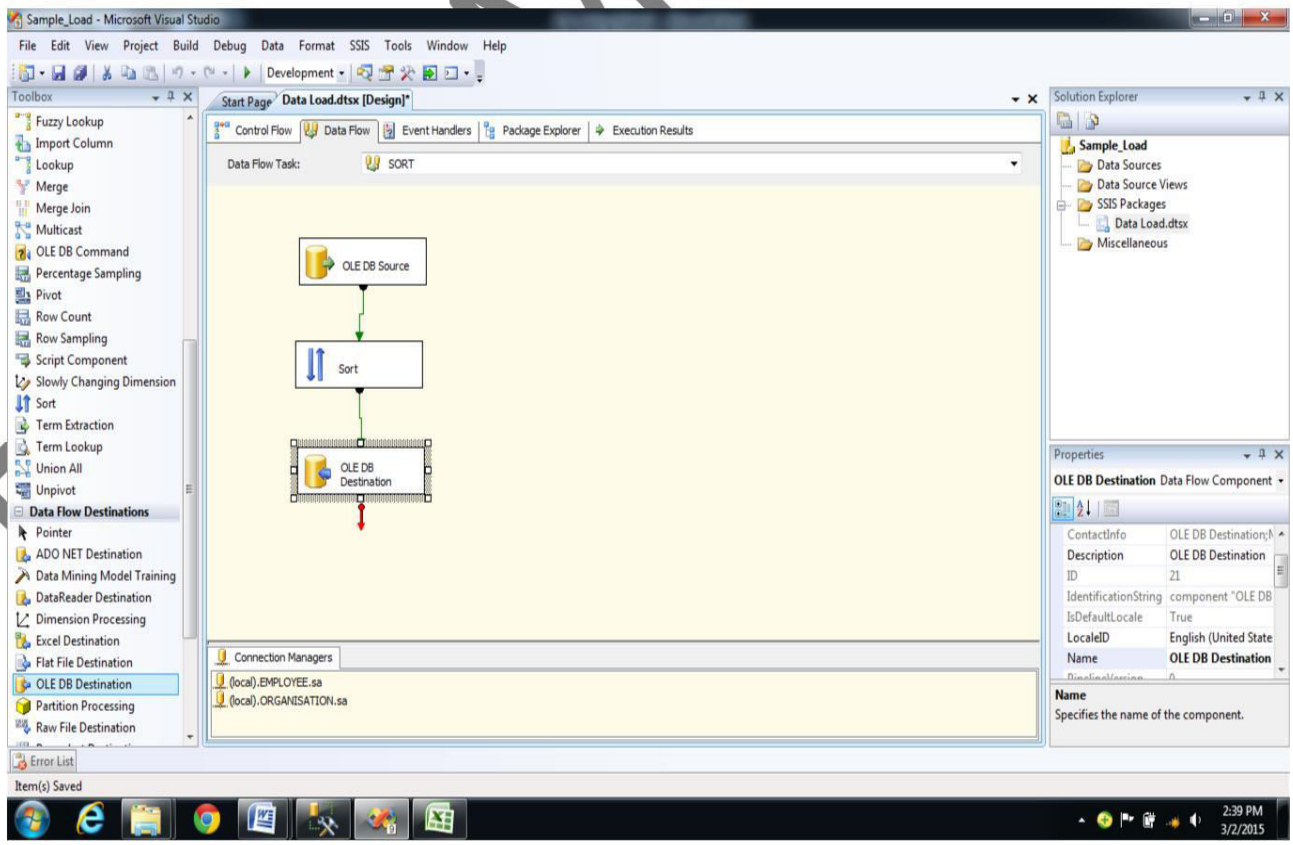
No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Target: Employee
DB: ORGANISATION

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int

2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

The Below screen gives you the look of sorting the data and loading in the Target table.



6.3 Joiner(MERGE JOIN) Transformation:

This joins the data from number of tables that are required to get the correct data to be loaded in the Target.

Let us now take an example of loading the data from the two different tables from the EMPLYOEE database to a table in the ORGANISATION database table.

Source: Employee1
DB: EMPLOYEE

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Project:

No.	Column Name	Column Comment	Column Data Type
1	Project_ID	Project Number	Int
2	Project_Name	Project Name	Varchar(50)

3	Employee_id	Employee number	Int
4	Proj_assign_date	Project Assigned date	Date

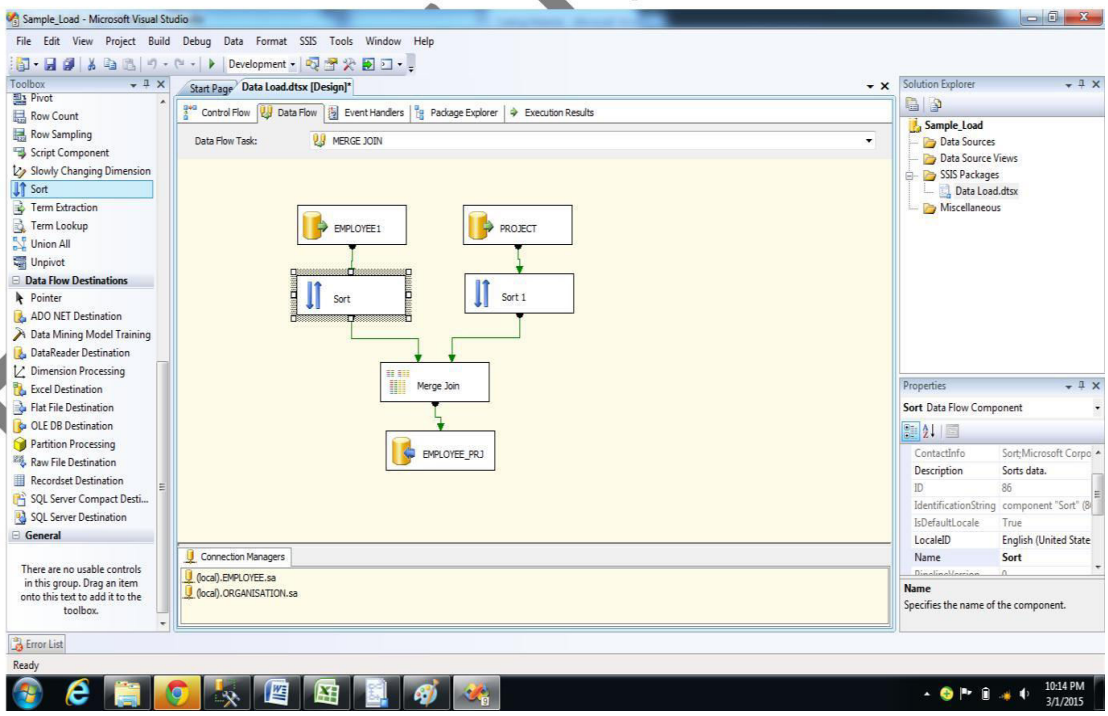
Target: Employee_prj

DB: ORGANISATION

The Target structure will be as below:

No.	Column Name	Column Comment	Column Data Type
1	EMP_ID	Employee number	Int
2	EMP_FNAME	Employee Firstname	Varchar(50)
3	EMP_LNAME	Employee Lastname	Varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	Varchar(50)
6	EMP_PROJECT	Project Name	Varchar(50)
7	EMP_PRJDATE	Project Assigned date	Date

The below screen gives you the look of loading the data from Source to the Target by joining the two tables from the Source and loading the result in to the Target.



6.4 Derived Column Transformation:

This Transformation is used to create a new column in the Transformation so that the intented data is moved on the Target from the Source.

Let us now take an example of loading the data from EMPLOYEE1 table from the EMPLOEE database to a table in the ORGANISATION database table.

Source: Employee1

DB: EMPLOYEE

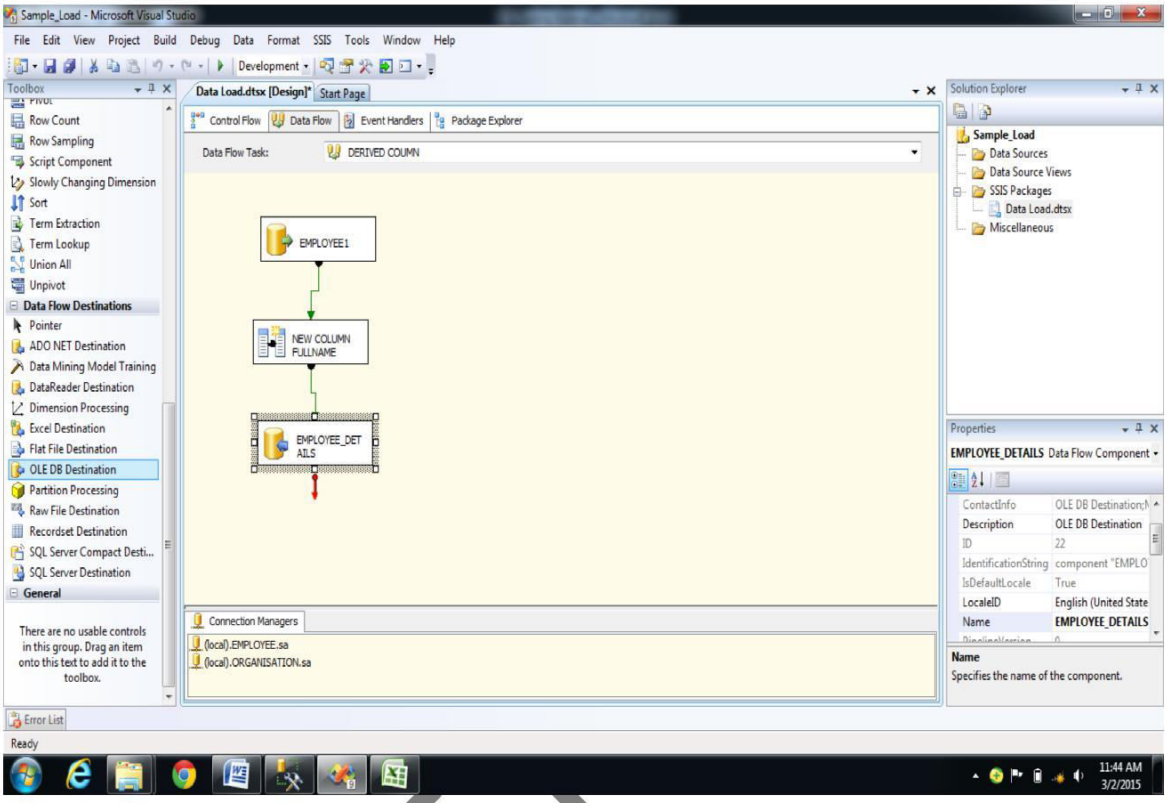
No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Target: Employee_Details
DB: ORGANISATION

The Target structure will be as below:

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_FULLNAME	Employee Fullname	varchar(100)
5	EMP_AGE	Employee Age	Int
6	EMP_DESIG	Employee Designation	varchar(50)
7	EMP_SALARY	Employee Salary	Decimal(10,2)

The Below screen gives you the look of loading the data from Source to the Target along with the new column derived from the Source table.



6.5 Lookup Transformation:

This Transformation is used to lookup a value in a source and do the necessary task to accomplish for like either update while the particular record is present or insert f not present.

Let us now take an example of loading the data from EMPLOYEE1 table from the EMPLOYEE database to a table in the ORGANISATION database table.

Source: Employee1
DB: EMPLOYEE

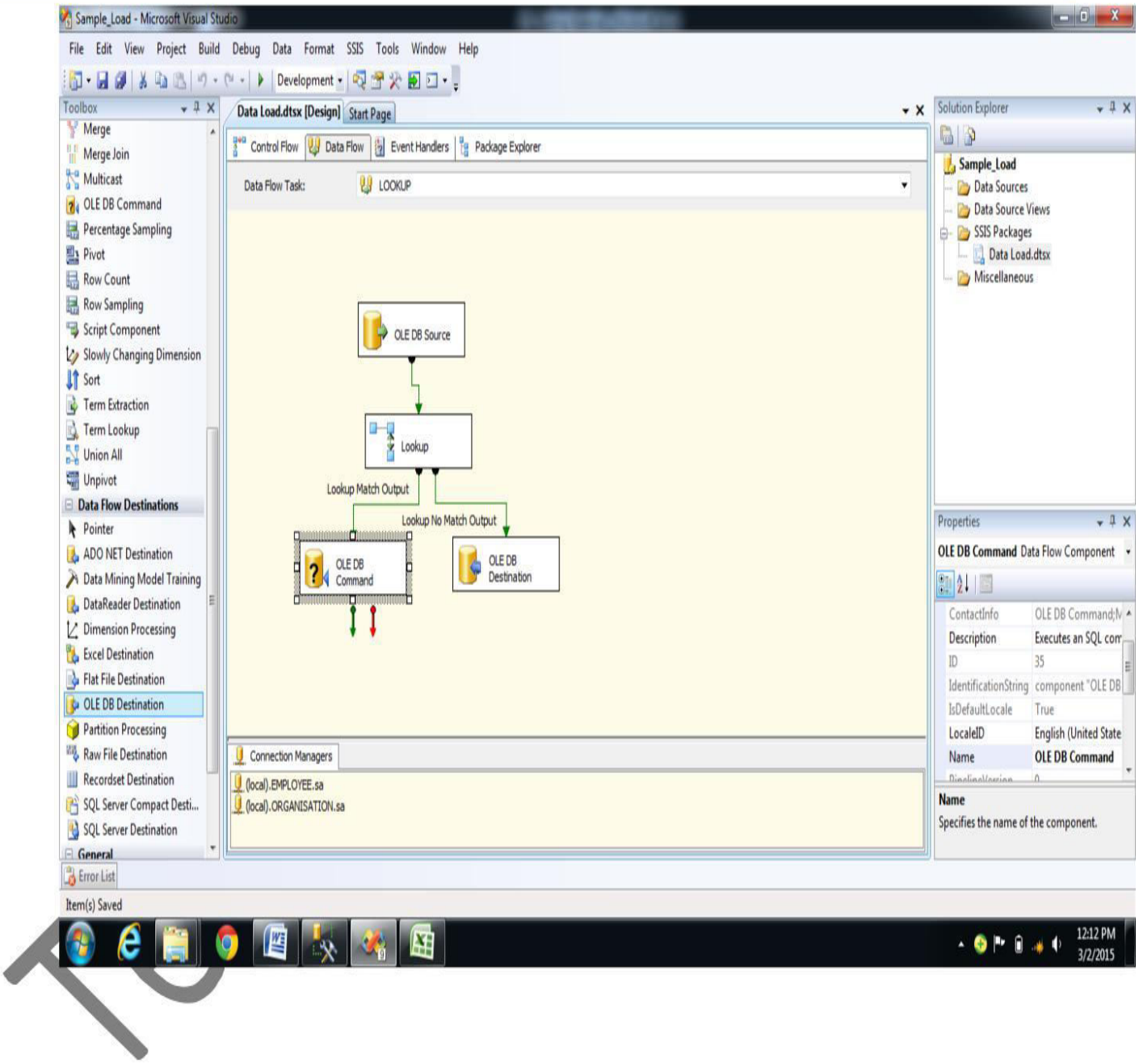
No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Target: Employee
DB: ORGANISATION

The Target structure will be as below:

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

The Below screen gives you the look of loading the data from Source to the Target by looking up the data in the Source (If data is already present in the target it updates and if not present will Insert).



6.6 Union All Transformation:

This Transformation is used to Union all the data from the sources to the target.This can also be achieved by using join. But there are certain differences between Union All and Merge Join which is described below

- Union All does not require the data to be sorted where Merge Join needs the data to be sorted.
- Union all does not remove duplicates where merge join gives out only one row when matched.
- Merge Join can be done only on two inputs where Union all can be on any number of inputs.

Let us now take an example of loading the data from EMPLOYEE1 table from the EMPLOYEE database and EMPLOYEE table in the ORGANISATION database to a different table in the ORGANISATION database

Source 1: Employee1
DB: EMPLOYEE
Source 2: Employee
DB: ORGANISATION

Employee1:

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Employee:

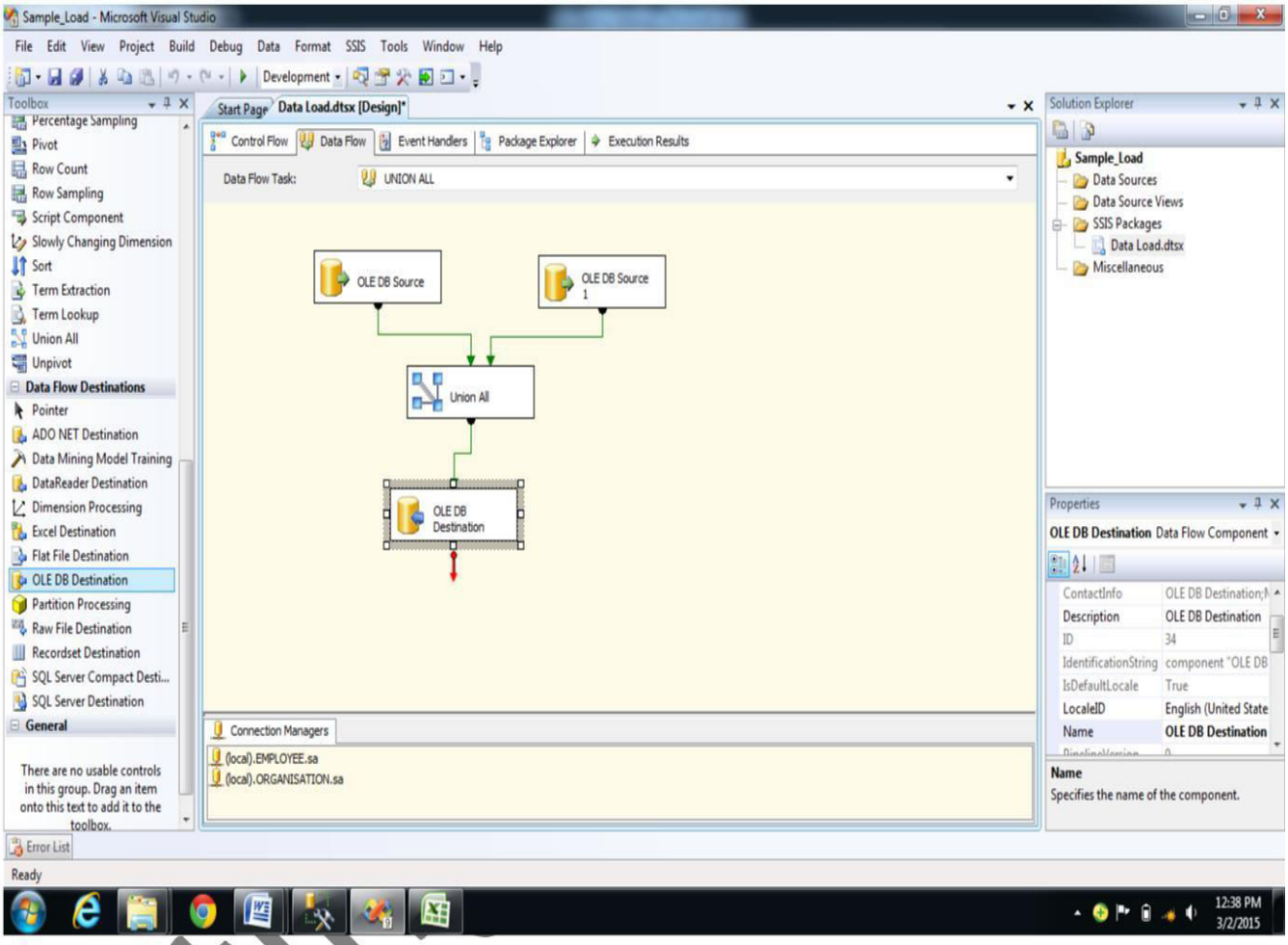
No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

Target: Employee_union
DB: ORGANISATION

The Target structure will be as below:

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

The next screen gives you the look of loading the data from two Sources to the Target by union transformation, which means the data from the 2 input tables are combined and loaded in the Target table.



6.7 Conditional Split Transformation:

This Transformation is used to split the data from the Source based on a specified condition.So if the condition satisfies it goes to a separate flow and for the data set that does not satisfies the condition will move in to another flow for processing.

Let us now take an example of splitting the Source dataset and loading into the Target table for the employees whose salary is equal to 5000.

Source: Employee1

DB: EMPLOYEE

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

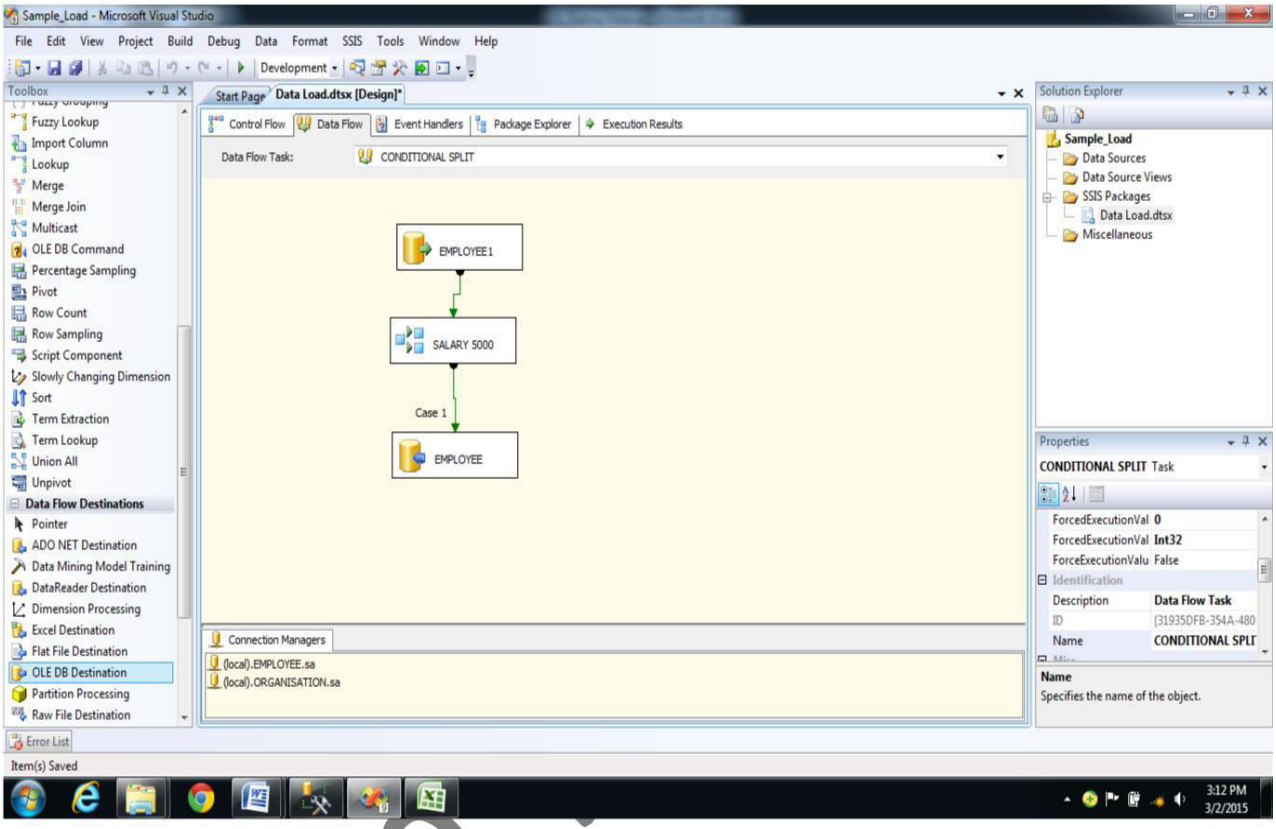
Target: Employee

DB: ORGANISATION

The Target structure will be as below:

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

The Below screen gives you the look of splitting the dataset where the employee's salary is equal to 5000 and load that in the Target table.



6.8 Script Component Transformation:

This Transformation is used to write the VB script for the code that does not exist in the tool by default.

Let us now take an example of creating the script on the Source table and loading it in to the Target

Source: Employee1
DB: EMPLOYEE

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

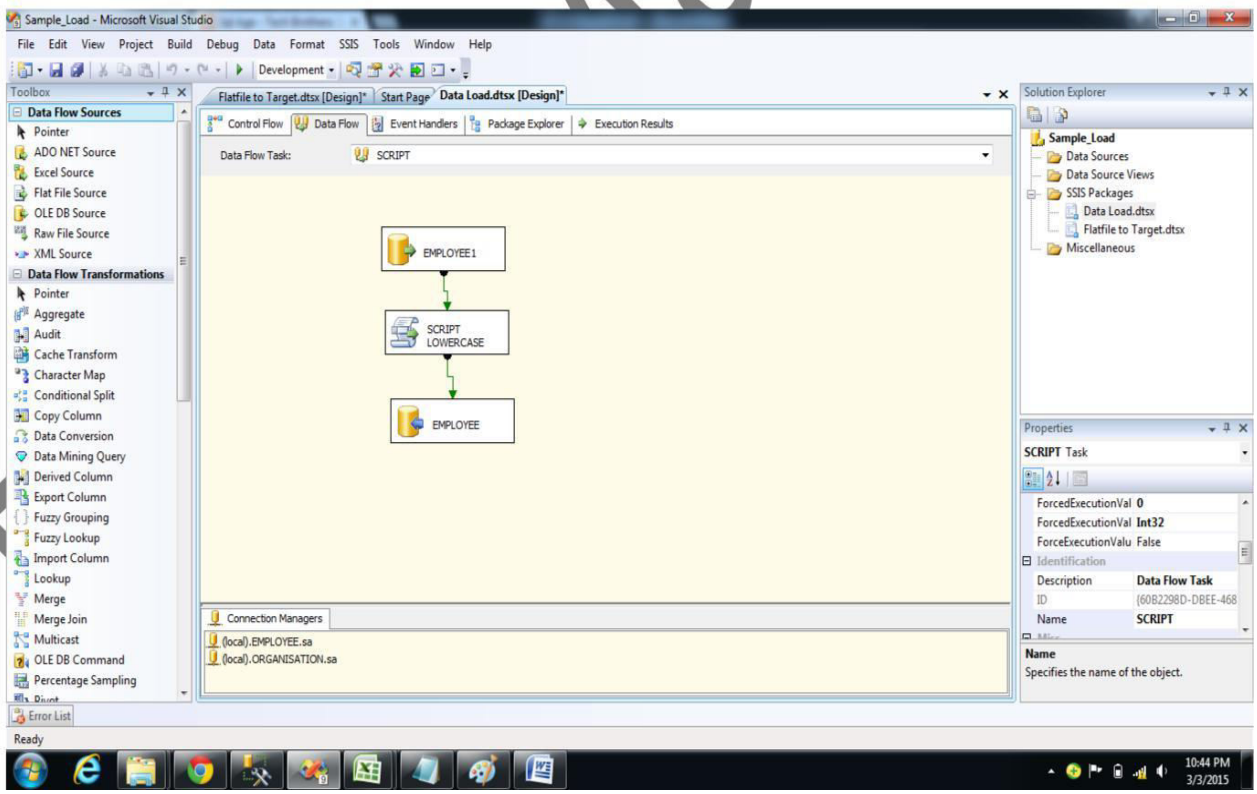
Target: Employee

DB: ORGANISATION

The Target structure will be as below:

No.	Column Name	Column Comment	Column Data Type
1	EMPID	Employee Id	Int
2	EMP_FNAME	Employee Firstname	varchar(50)
3	EMP_LNAME	Employee Lastname	varchar(50)
4	EMP_AGE	Employee Age	Int
5	EMP_DESIG	Employee Designation	varchar(50)
6	EMP_SALARY	Employee Salary	Decimal(10,2)

The Below screen gives you the look of handling a script component in the tool.



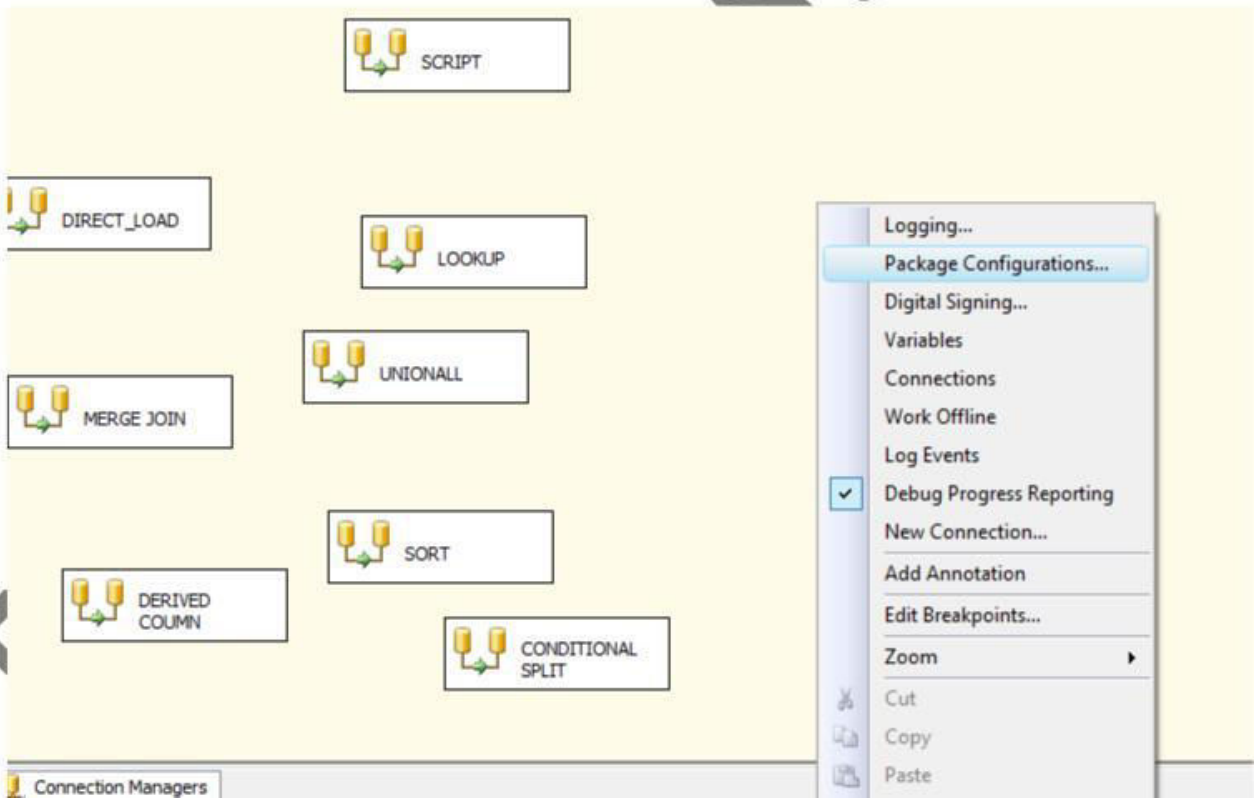
1. **Creation of Configuration File in SSIS Package**

Configuration file is usually an XML file that was created with .dtsconfig extension, which is mainly used when the packages are being transferred into multiple servers and handled. When these is a change of the server for different environments like (DEV,QA,UAT,PROD), It is not always necessary that every environment will have the same folder structure or the same name of the database and the server name will obviously be different

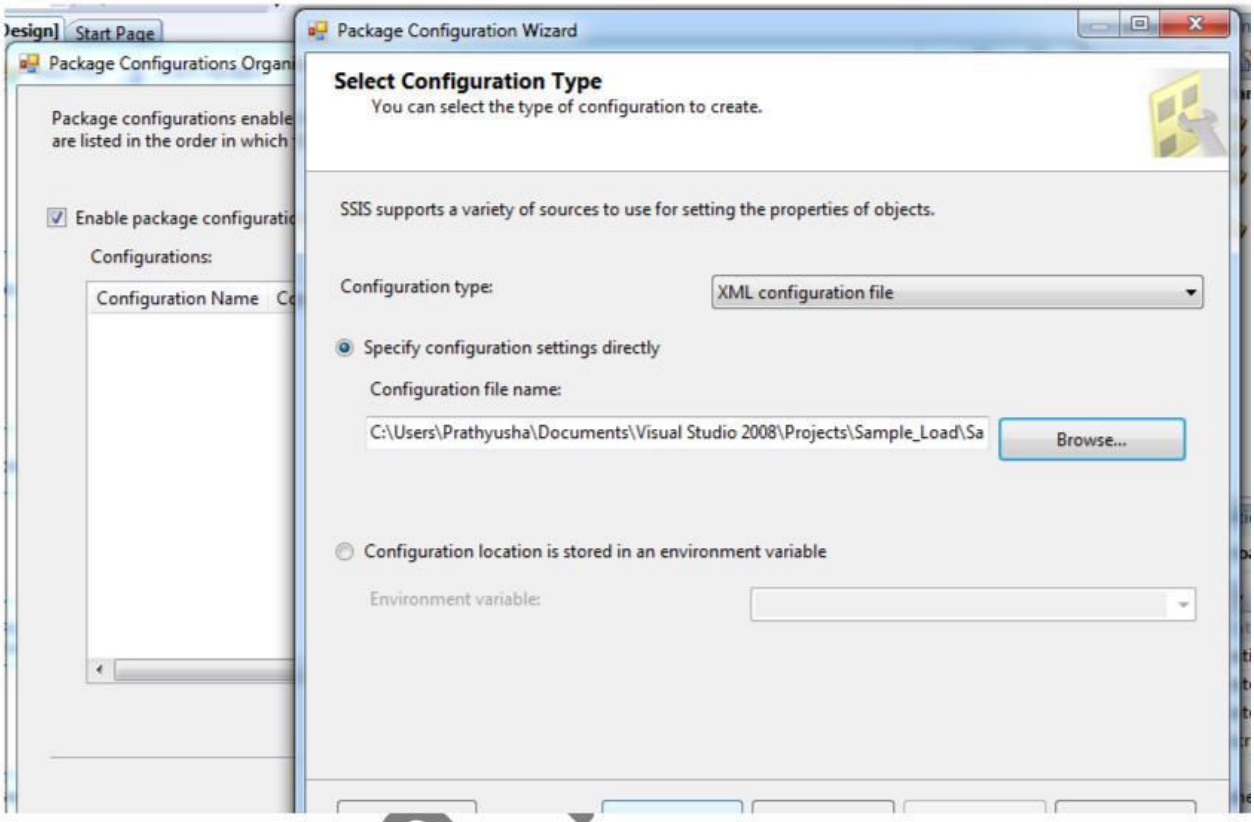
So the configuration file proves more effective when used in the following scenarios

- When the instance name is different
- When the Database name is different
- When the Login ID are different
- When the folder structure is different

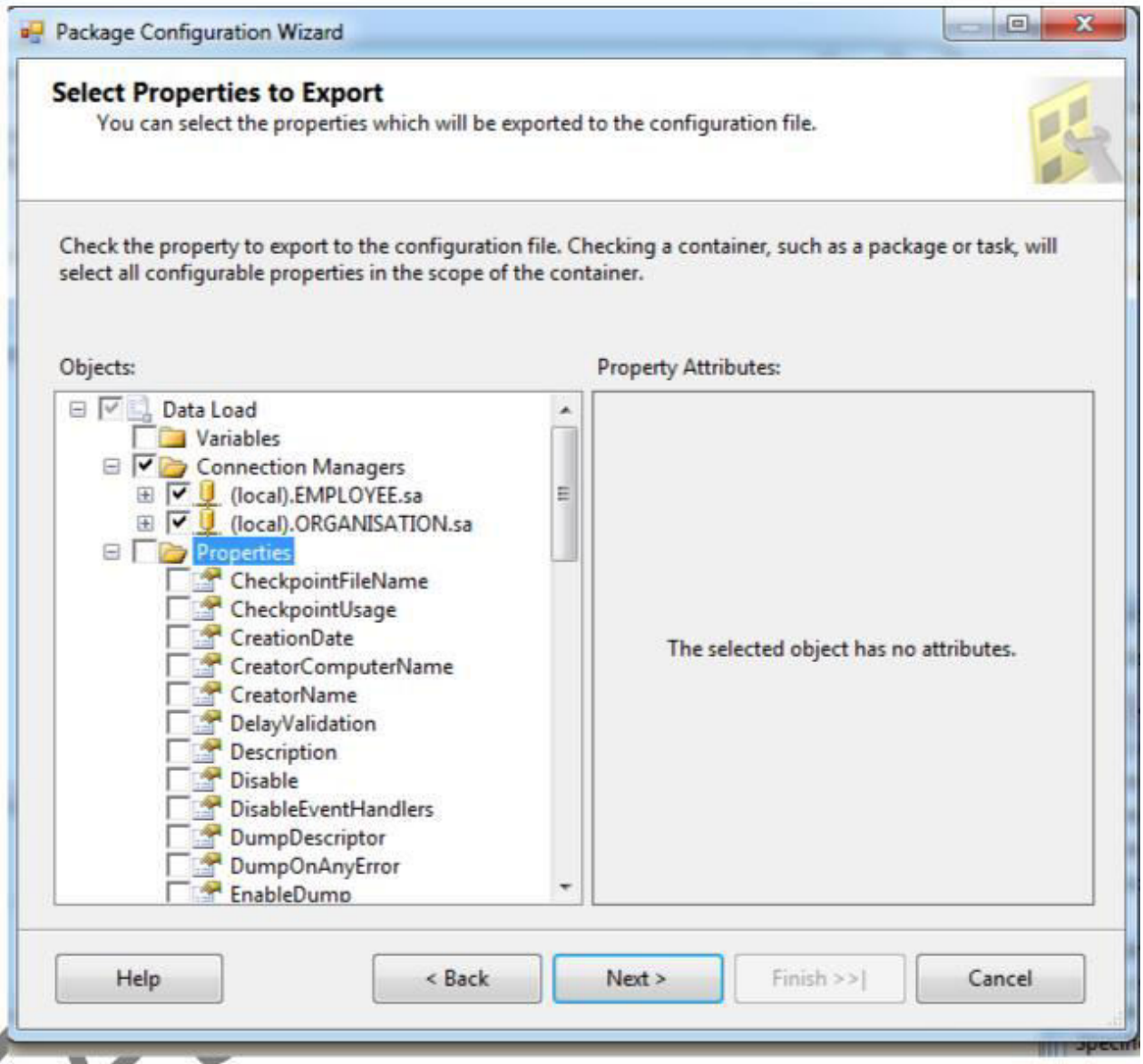
i. Right click on the Control flow and select Package Configurations

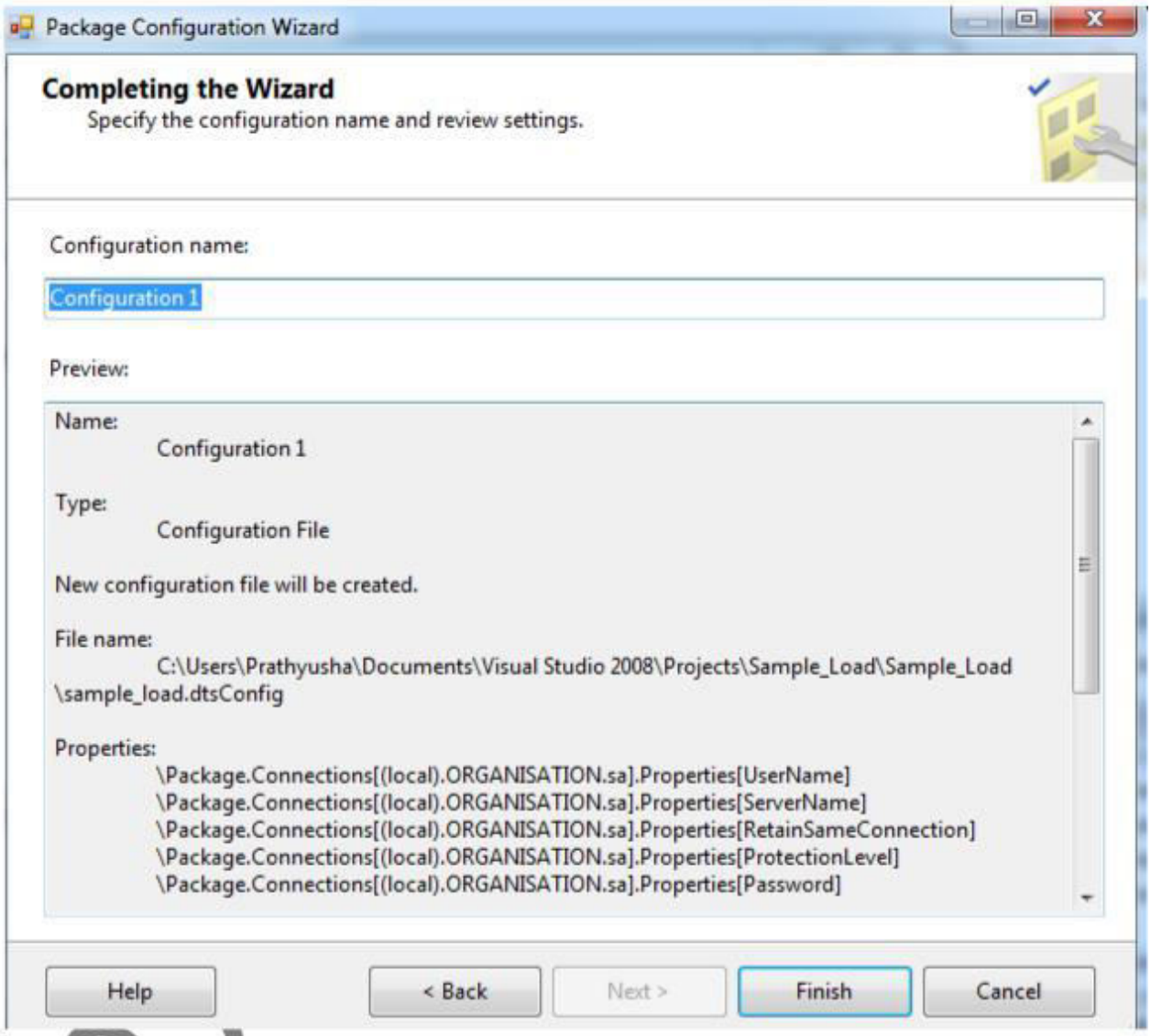


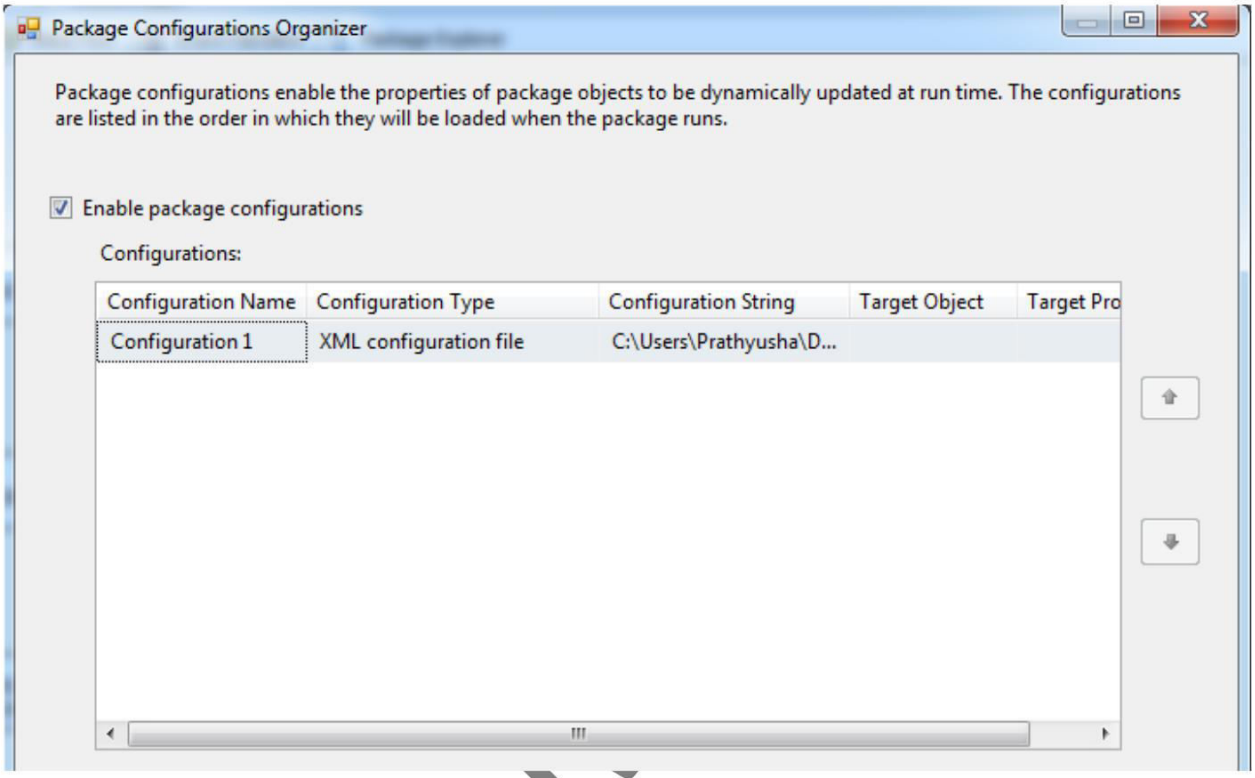
ii. Enable Package configurations and click on Add button and then browse a path for creation of the file.



iii. Click on Next and select the tasks on which the config file needs to be created,Usually we select the Connection Managers and the Variables if any and click on Finish and then the config will be created in the path given


Package Configuration Wizard
Select Properties to Export
You can select the properties which will be exported to the configuration file.
Check the property to export to the configuration file. Checking a container, such as a package or task, will select all configurable properties in the scope of the container.
Objects:
Property Attributes:
Data Load
Variables
Connection Managers
(local).EMPLOYEE.sa
(local).ORGANISATION.sa
Properties
CheckpointFileName
CheckpointUsage
CreationDate
CreatorComputerName
CreatorName
DelayValidation
Description
Disable
DisableEventHandlers
DumpDescriptor
DumpOnAnyError
EnableDump
The selected object has no attributes.
Help
< Back
Next >
Finish >>|
Cancel

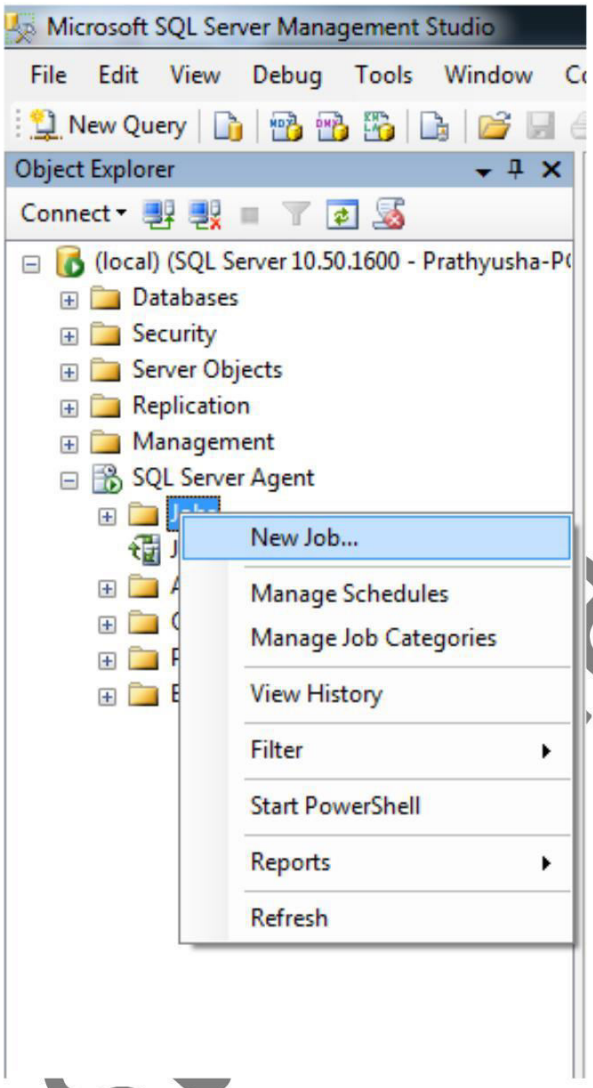

Package Configuration Wizard
Completing the Wizard
Specify the configuration name and review settings.
Configuration name:
Configuration 1
Preview:
Name:
Configuration 1
Type:
Configuration File
New configuration file will be created.
File name:
C:\Users\Prathyusha\Documents\Visual Studio 2008\Projects\Sample_Load\Sample_Load\sample_load.dtsConfig
Properties:
\Package.Connections[(local).ORGANISATION.sa].Properties[UserName]
\Package.Connections[(local).ORGANISATION.sa].Properties[ServerName]
\Package.Connections[(local).ORGANISATION.sa].Properties[RetainSameConnection]
\Package.Connections[(local).ORGANISATION.sa].Properties[ProtectionLevel]
\Package.Connections[(local).ORGANISATION.sa].Properties[Password]
Help
< Back
Next >
Finish
Cancel



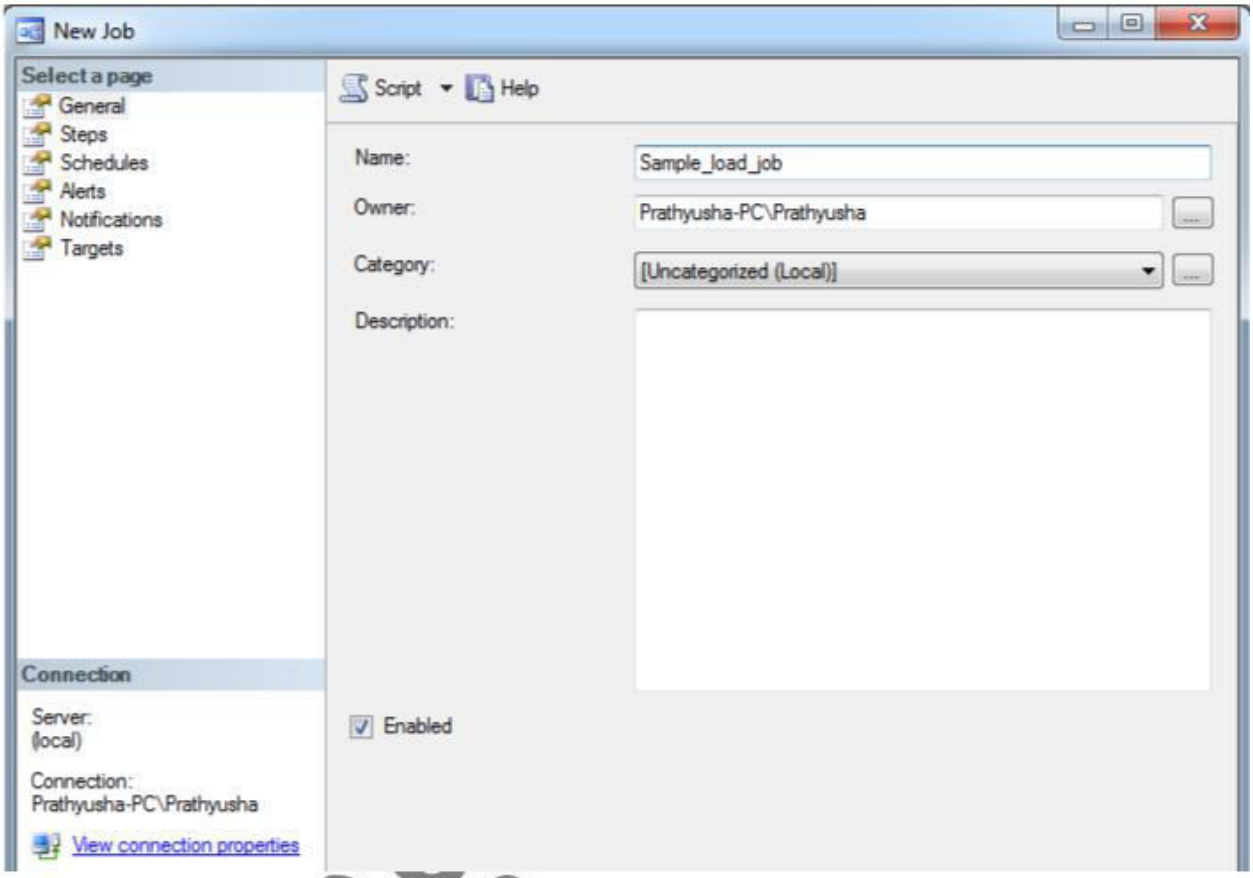
2. Creation of a Sql Server job for SSIS package

i. Open the Sql Server management Studio and click on the SQL Server Agent which is shown on the left side panel in the Object Explorer

ii. Expand it and right click on the jobs to create a new job



iii. Provide the Name for the Job

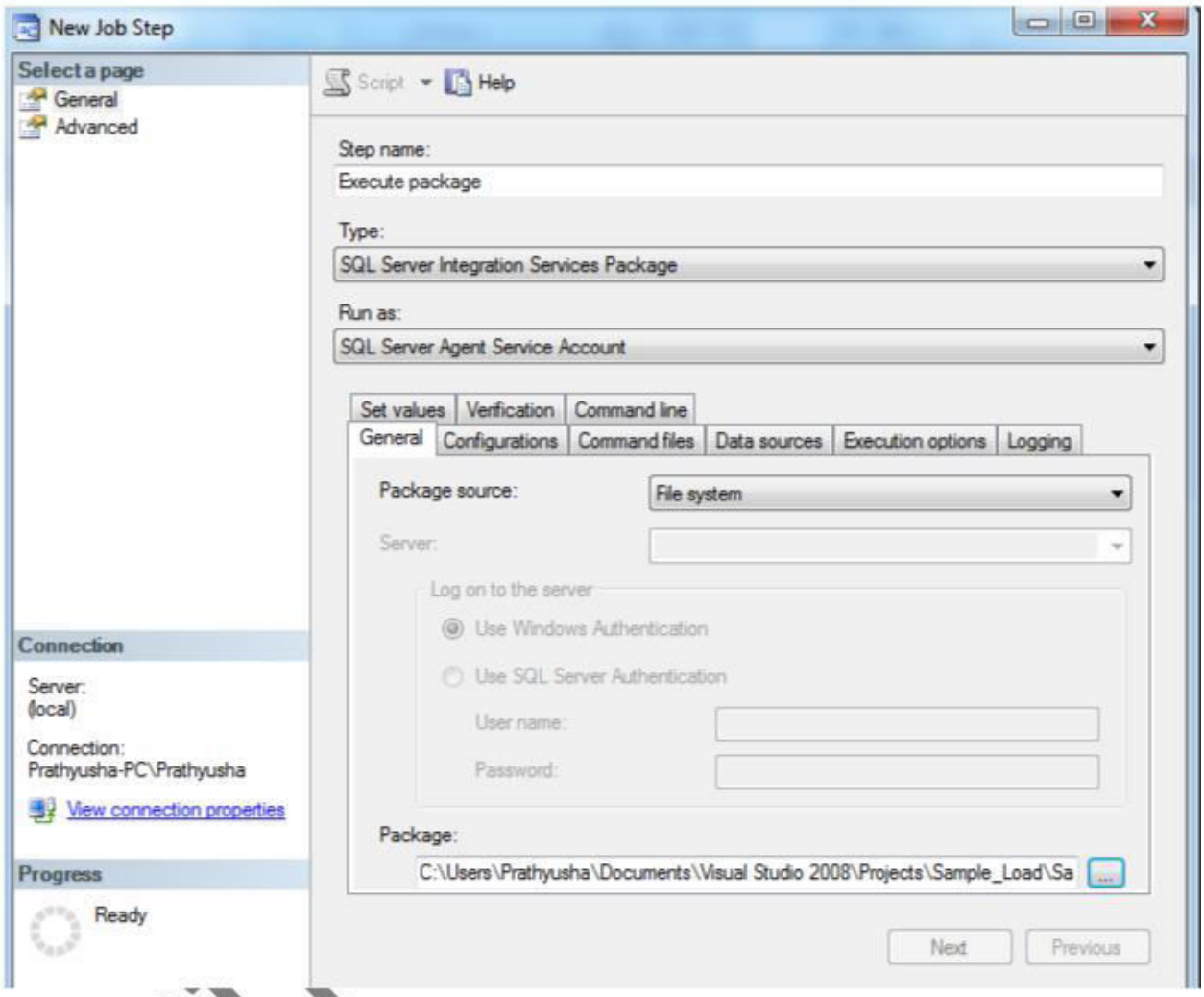


iv. Click on the Steps in the left side panel on the same window and click on the New Button below

v. Write the Step name as required and Select Type as Sql Server Integration Services Package.

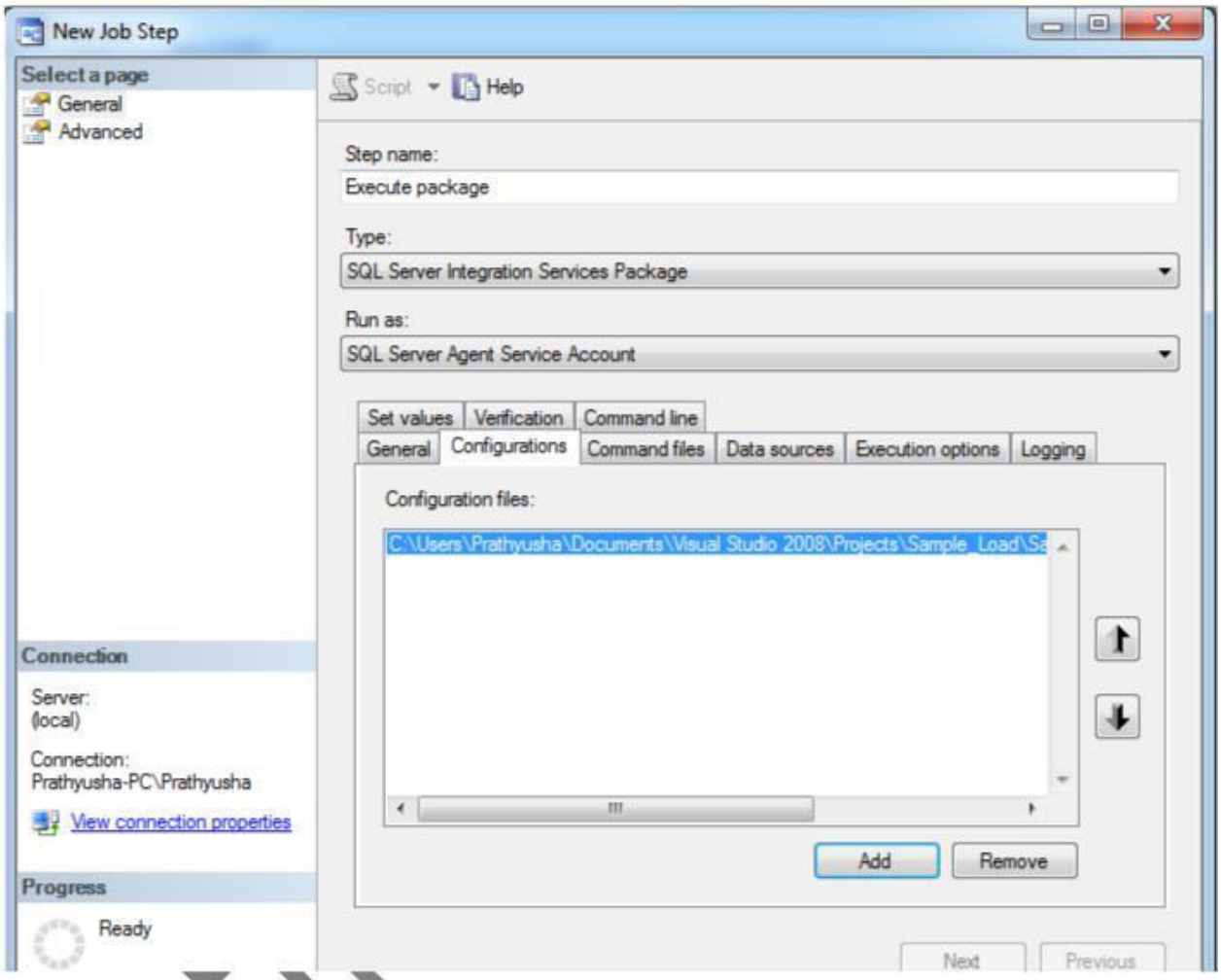
Select the package Source as the File system because all our packages would be stored in the Folder structure

Browse the package path below



vi. Click on the Configurations tab on the same window and add a config file to that by browsing the configuration file already created for the package

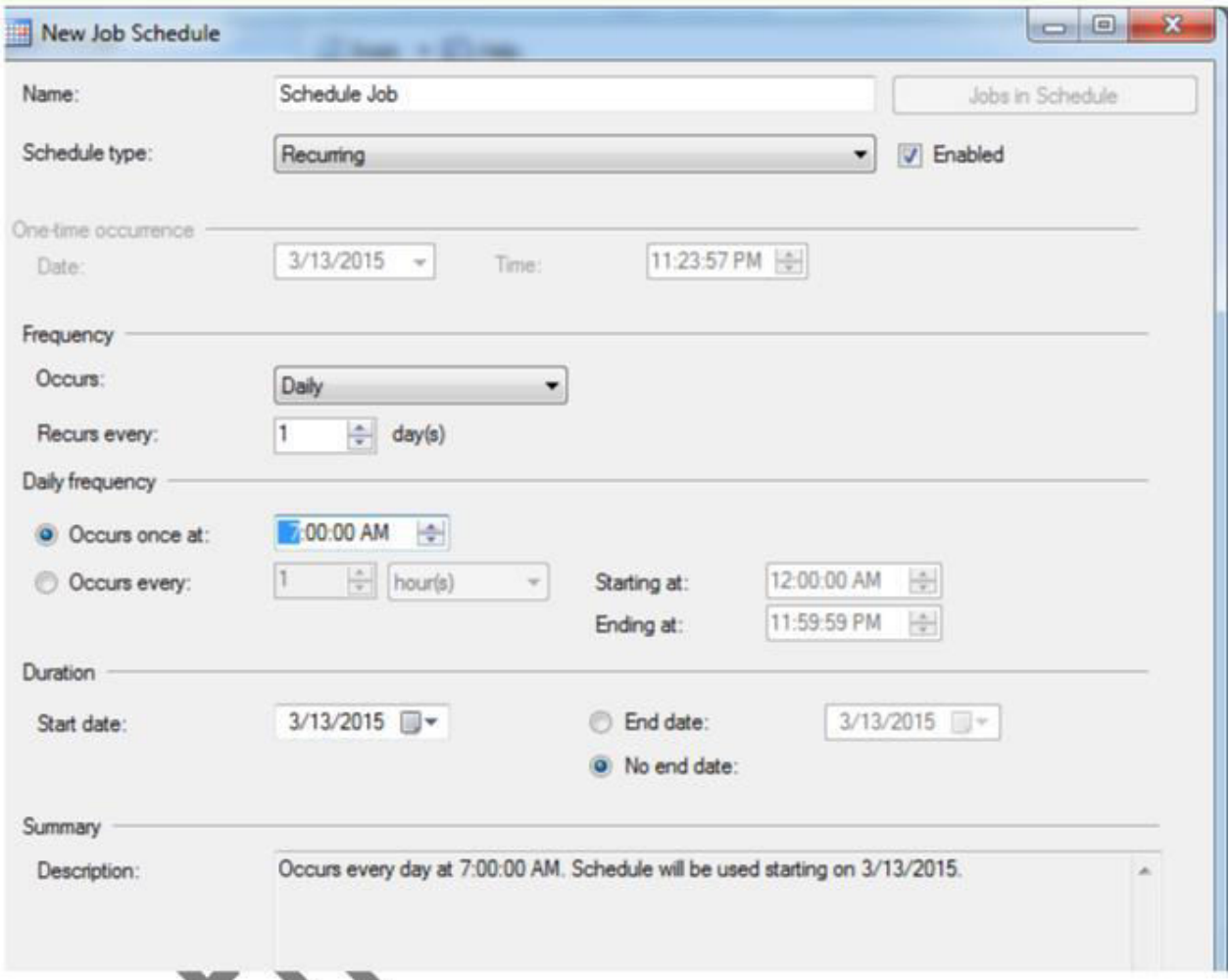
Click on OK.



vii. Select the Schedules in the first window opened to create an job run schedule which executes on itself for a specified time

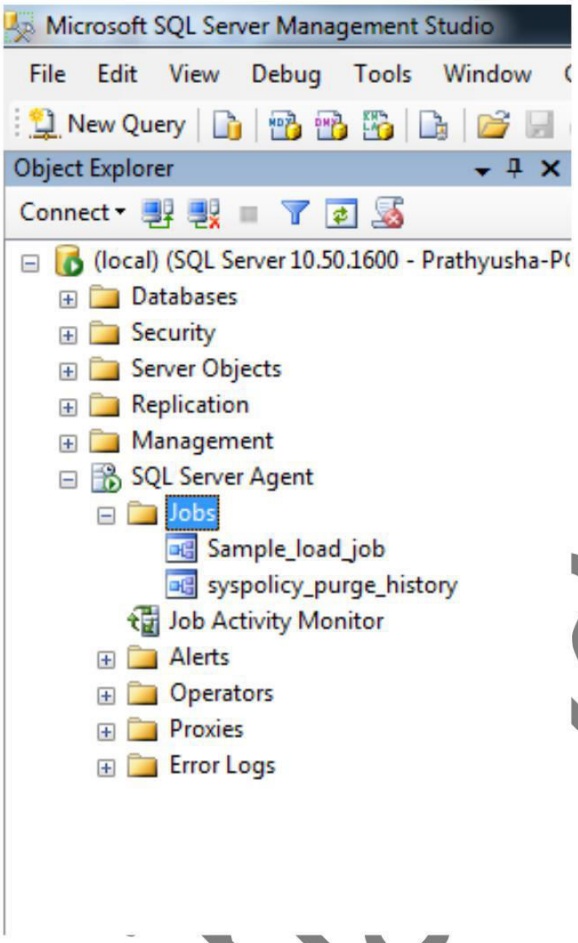
Create a New schedule for the job, Provide the name for the schedule, select whether daily or monthly and the date and time when it has to run

Click on OK and the job is now ready

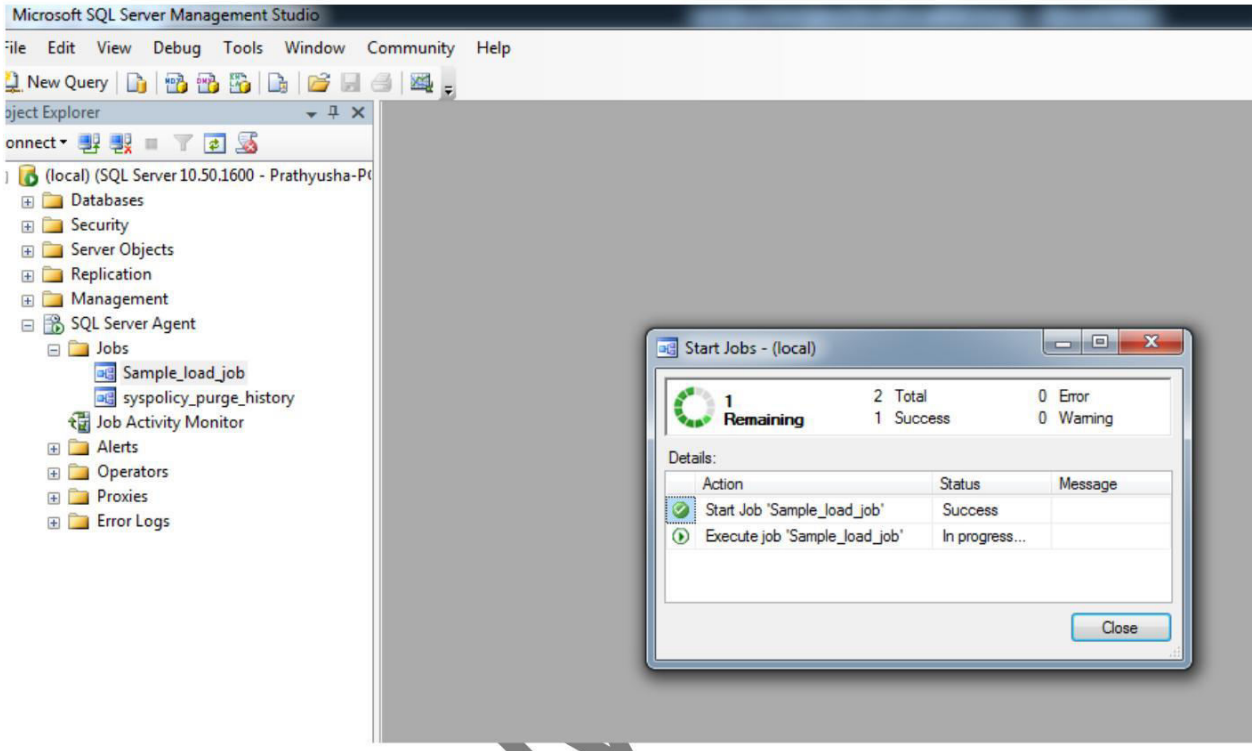


3. Execution of Job

i. You can view the Jobs in the Sql Server agent in the left side panel.

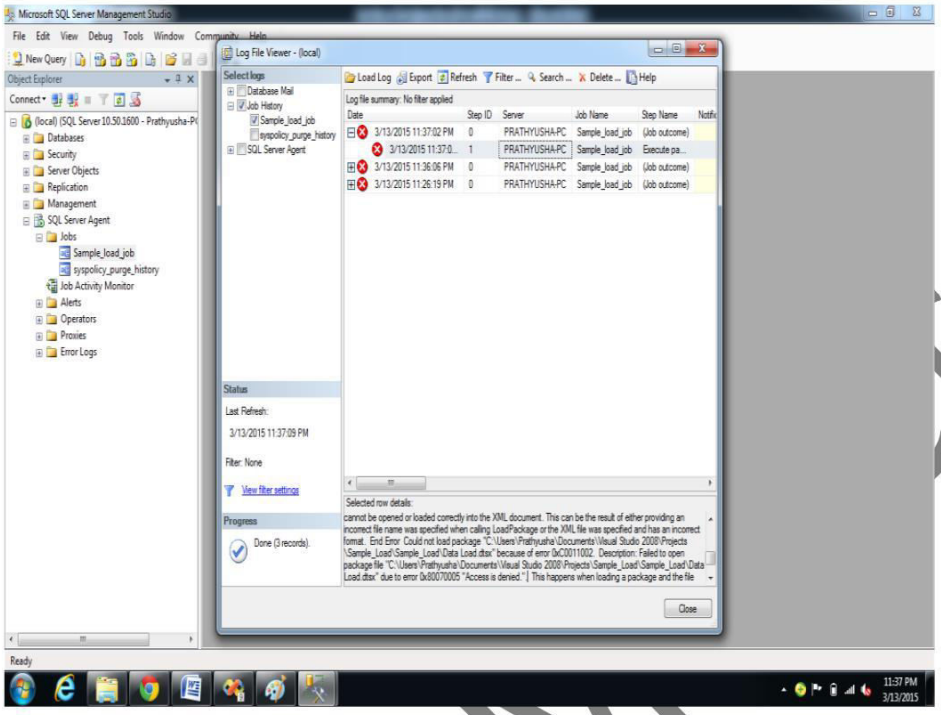


ii. Right click on the Job and select start at step and the job gets on executing



iii. It returns success message if it passes and a Failure if it fails which can identified either through the message or with the Green or Red color respectively

iv. Right Click on the job and select View history to see whether the Job has executed Successfully or not and you can also view the message in the below panel if the job failed and know the reason for failure and raise a defect

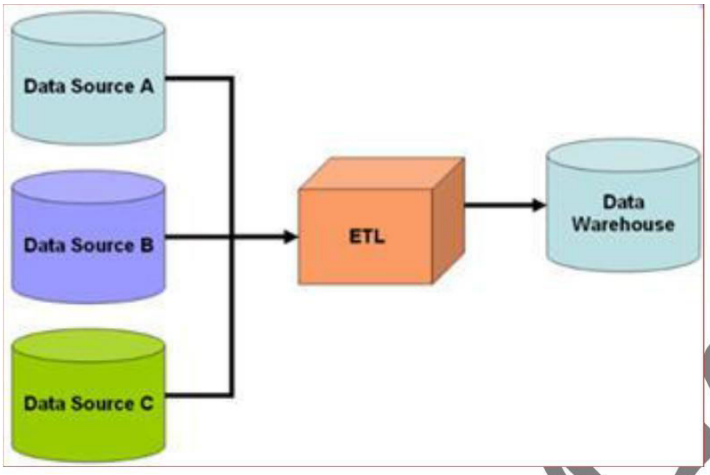


ETL TESTING

Chapter 7 - ETL Testing:

7.1 Introduction to ETL Testing: ETL Testing is a kind of testing technique which has to be done with human interaction where it needs to test the Extraction, Transformation and Loading of data when moved from source to Target based on the Business Requirements that were provided.

Consider the below block in which data is being moved from Source to Target through an ETL tool.ETL testing can be used for testing the Data Accuracy and the Data Completeness.



The ETL Testing Methodology includes following steps:

- Understanding of data to be reported
- Understanding and review of data model
- Understanding and review of source to target mappings (transformations)
- Data Quality Assessment of Source data
- Packages testing
- Schema testing (source and target)
- Verify Data completeness
- Verification of transformation rules
- Comparison of sample data between source and target
- Checking of referential integrities and relations (primary key foreign key)
- Data Quality checks on target warehouse
- Performance tests

7.2 ETL Testing Concepts: Basically to start away with ETL Testing, one should obviously need to understand on the SDLC where the project starts and where it ends.

Below is the Lifecycle for ETL development which shows from Requirements gathering to deployment in Production.

7.2.1 REQUIREMENT ANLAYSIS:

For any project to work on, we actually need set of rules to design the project which are in other words called as Requirements.

Requirements should be clear and specific with no uncertainty, requirements should be measurable in terms of specific values, requirements should be testable having some evaluation criteria for each requirement, and requirements should be complete, without any contradictions.

The project will start itself in the early stages with the Requirements gathering which actually gives us the idea of the Business requirements and based on the confirmation we have to develop the project for their need and move it to Production.

Requirements gathering phase is very crucial because most of the bugs that arise can be due to ambiguities in the requirements.

For a Developer or a tester this Requirement gathering phase is more important such that developer can develop abased on his understanding and Tester has to test in on his understanding and both should be in sync with each other in order to move that to Production.

All the Requirements that were captured as part of initial discussions with the business are documented in the Software Requirement Specification (SRS) document which acts as a bible for the whole project.

Let's take an example of how the requirements could be

Eg 1: Let us suppose we need to work for an web based project which gives the overview of all the employees in an industry.

Requirement: A customer should be able to login with their employee id to view their Employee details.

Discussion: As a project team you will be asking the business of what kind of details and employee should be able to view when they login with their employee id.

Answer: They should be able to view their Firstname,MiddleName,Surname,Date of Birth,Sex, Designation and their achievements.

In this example, The SRS document should consist of all these discussions but with a format.

So, now the Requirement in the SRS document could be as below:

Requirement: A customer should be able to login with their employee id to view their Employee details such as Firstname,MiddleName,Surname,Date of Birth,Sex, Designation and their achievements

There can also be a case where the business will provide the Requirements in a sheet which was agreed in the higher authority and when the team looks at it, everyone has their own conceptions and was not aware of the terminology that was used.

Then the SRS document should be sent back to the business for the clarifications and then modify the document in the next version that is released and then move on to the Design Phase after freezing the Requirements

One more main thing that should be followed in these Requirements is to be clear on the requirements. Nothing should be assumed and done so that it fails in testing and does not move to Production

Let us now consider a simple project to move on with ETL testing complete flow

Requirement: The solution must be able to load the data from the Flat file in to the Table in a database

High Level Design:

Our solution should be able to fetch the fetch the flat file data present in the particular location which is the Company.txt file and load that in the ORGANISATION database

This can be treated as an interface and divided into below flows

- Load the data from the Flat file in to the Staging table
- Fetch the data from the staging table after the transformations and load that in the Target table

Low Level Design:

The Low level design is also document in a document called as Technical Specification document where the Requirement information and the File formats are agreed.

The low level design document contains the description of the requirement of what needs to be done during the design which means what should be achieved in the development

Below are the specifications:

- The fie should be on the Local server or in any server where the file can be accessed through the network.
- The path of the file should be specified as below:D:\Flatfiles\Company.txt

Below are the Source and the Target details:

Source: Flatfile**Company.txt**

company_id|company_name|Address1|Address2|City|State|Postalcode
1001|Northern Drivers|167 Ferndale|Beside church|London|UK|123456
1002|Southern Drivers|124 morgan||Tontown|CA|940432
1024|Transocean|2312 High road||carbovy|CA|940432
1032|Eastern Drivers|1 High Street|Crickalde road|newjersy|CA|940432
1043|Western Drivers|123 Broadstreet|Ferndale road|Fort|CA|944564
1044|Avio Air|57 Donny Street||Newyork|CA|786908
1056|Sun systems|1 Avenue||London|CA|987567
1066|Borco Industries|2 juro view|Mainroad|Tontown|CA|980780
1074|Airvoice|234 backstreet||carbovy|CA|908089
1075|Duro industres|121 high street||Fort|CA|980780

Target DB: ORGANISATION

Target Stagingtable: Company_STG

No.	Column Name	Column Comment	Column Data Type
1	company_id	Company Number	BIGINT
2	company_name	Company Name	VARCHAR(60)
3	Address1	Address line 1	VARCHAR(60)
4	Address2	Address Line 2	VARCHAR(60)
5	City	City	VARCHAR(50)
6	State	State	VARCHAR(50)
7	Postalcode	Postcode	VARCHAR(8)

Target Table: Company

No.	Column Name	Column Comment	Column Data Type
1	company_id	Company Number	BIGINT
2	company_name	Company Name	VARCHAR(60)
3	Address1	Address line 1	VARCHAR(60)
4	Address2	Address Line 2	VARCHAR(60)
5	City	City	VARCHAR(50)
6	State	State	VARCHAR(50)
7	Postalcode	Postcode	VARCHAR(8)
8	Date_inserted	Inserted Date	Date
9	Date_modified	Modified Date	Date

Complete Mapping from the Source to Staging is as below:

Source **Staging**

No.	Column Name	Column Comment	Column Data Type	No.	Column Name	Column Comment	Column Data Type
1	company_id	Company Number	BIGINT	1	company_id	Company Number	BIGINT
2	company_name	Company Name	VARCHAR(60)	2	company_name	Company Name	VARCHAR(60)
3	Address1	Address line 1	VARCHAR(60)	3	Address1	Address line 1	VARCHAR(60)
4	Address2	Address Line 2	VARCHAR(60)	4	Address2	Address Line 2	VARCHAR(60)
5	City	City	VARCHAR(50)	5	City	City	VARCHAR(50)
6	State	State	VARCHAR(50)	6	State	State	VARCHAR(50)
7	Postalcode	Postcode	VARCHAR(8)	7	Postalcode	Postcode	VARCHAR(8)

The Complete mapping from Staging to Target is as below:

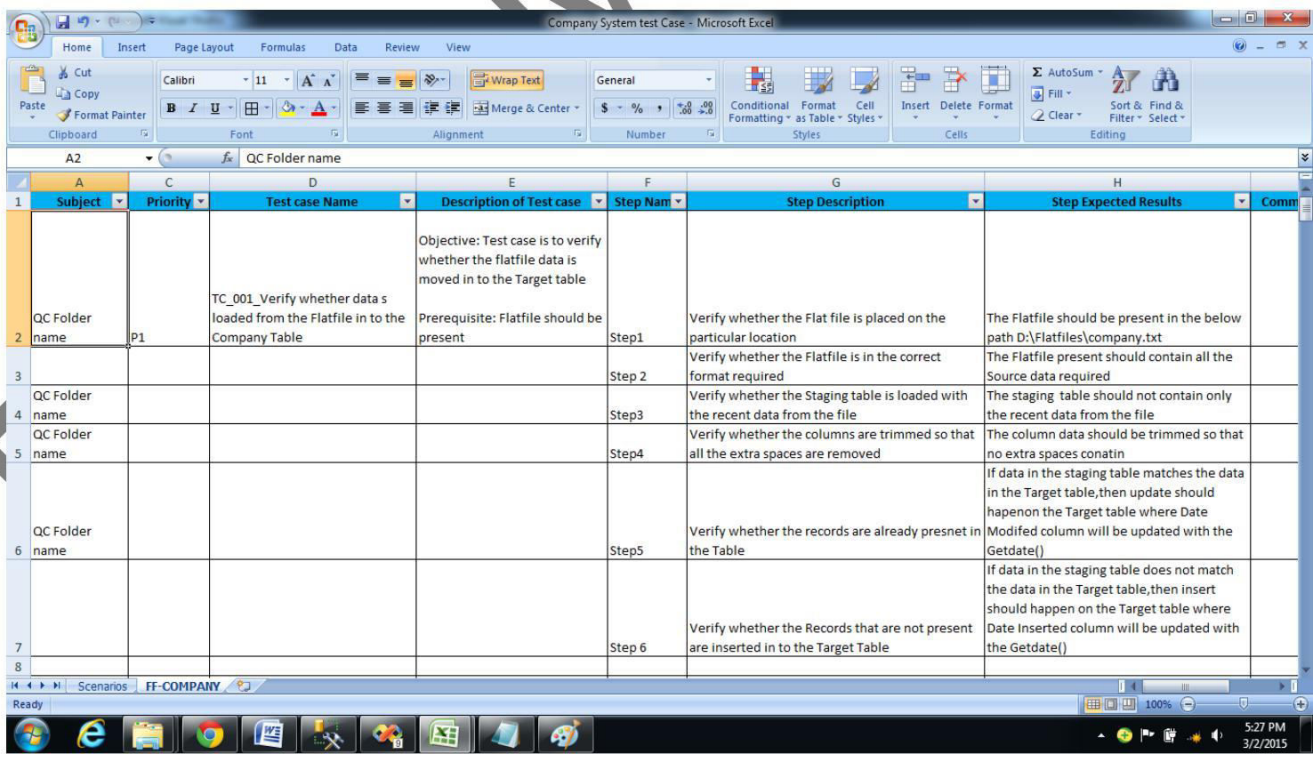
Staging				Target			
No.	Column Name	Column Comment	Column Data Type	No.	Column Name	Column Comment	Column Data Type
1	company_id	Company Number	BIGINT	1	company_id	Company Number	BIGINT
2	company_name	Company Name	VARCHAR(60)	2	company_name	Company Name	VARCHAR(60)
3	Address1	Address line 1	VARCHAR(60)	3	Address1	Address line 1	VARCHAR(60)
4	Address2	Address Line 2	VARCHAR(60)	4	Address2	Address Line 2	VARCHAR(60)
5	City	City	VARCHAR(50)	5	City	City	VARCHAR(50)
6	State	State	VARCHAR(50)	6	State	State	VARCHAR(50)
7	Postalcode	Postcode	VARCHAR(8)	7	Postalcode	Postcode	VARCHAR(8)

The main testing that will be done to validate whether the package is working good or not is through the unit testing which will be again created by the developers. Below is the sample of the unit test cases that were created.

Designer	Status	Subject	Test case Name	Description of Test case	Step Name	Step Description	Step Expected Results	Comments/Input Data
Prathyusha	new	Sample_Load_Flatfile to Company Table	TC001_Test case to load the data from the Flatfile to Staging table	Validation of the data load from Flatfile to Staging table	Step1	select the columns from the Flatfile	All the input data from the Flatfile should be selected by the connection manager specified	
					Step2	Load the data from Flatfile into the Staging table	The data from the flatfile should be loaed in to the staging in the respective columns	
			TC001_Test case to load the data from the Staging table into Company table in the Target	Validation of the data load from Staging table to Target	Step1	select the columns from the Staging table	All the input data from the Staging table should be selected by the connection manager specified	
					Step2	Trim all the columns data	All the extra spaces present for that columns should be removed	
					Step 3	Update the data in the Target if already exists	Check the already existing data in the Target table and if exists update the table data based on the Key column	
					Step 4	Insert the new data in the target	If the data doesnot exist in the table,insert the new data	

After the Unit Testing is done, the next step is the System testing which now comes completely on the testers to create the overall system test cases for each requirement and test the functionality and then pass or fail the test cases based on the data validation.

Below is the reference for the System test case



Subject	Priority	Test case Name	Description of Test case	Step Nam	Step Description	Step Expected Results	Comm
QC Folder name	P1	TC_001_Verify whether data s loaded from the Flatfile in to the Company Table	Objective: Test case is to verify whether the flatfile data is moved in to the Target table Prerequisite: Flatfile should be present	Step1	Verify whether the Flat file is placed on the particular location	The Flatfile should be present in the below path D:\Flatfiles\company.txt	
				Step 2	Verify whether the Flatfile is in the correct format required	The Flatfile present should contain all the Source data required	
QC Folder name				Step3	Verify whether the Staging table is loaded with the recent data from the file	The staging table should not contain only the recent data from the file	
QC Folder name				Step4	Verify whether the columns are trimmed so that all the extra spaces are removed	The column data should be trimmed so that no extra spaces conatin	
QC Folder name				Step5	Verify whether the records are already presnet in the Table	If data in the staging table matches the data in the Target table,then update should hapenon the Target table where Date Modifed column will be updated with the Getdate()	
				Step 6	Verify whether the Records that are not present are inserted in to the Target Table	If data in the staging table does not match the data in the Target table,then insert should happen on the Target table where Date Inserted column will be updated with the Getdate()	

Once the System test cases are ready, the testers should be aware of the testing approach which has to be performed on how the testing should be done to achieve the requirement.

Once the testing approach is confirmed, Testers will have to validate the data,prerequisites,required software and hardware requirements as well as the knowledge about the developed requirements.

Chapter 8- Types of ETL Testing:

8.1 Constraint Testing:

In the phase of constraint testing, the test engineers identifies whether the data is mapped from source to target or not.

The Test Engineer follows the below scenarios in ETL Testing process.

8.1.1 NOT NULL: The tester needs to test the data in the Target table whether it's a not null column if the data is there in the table and should be a Null column if there is NULL in the table column.

Query used for validating the data:

Select * from Company where company_id is null

Company _id is the primary key column where it cannot accept null value.Check the same for every column.See whether there are any not null columns and you have null there in the table.

See whether there is a null value in the column data and check the table designs whether it can accept nulls or not.

8.1.2 UNIQUE: If the column has a unique constraint, the data should also be unique in the table.

8.1.3 Primary Key: The Primary key in the target table should be identified and there should be no duplicates in the table.

Query used for validating the data:

Select company_id,count(*) from company group by company_id where count(*)>1

This query should result in no rows which prove that there are no duplicates in the table.

8.1.4 Foreign key: The foreign key relation is checked in the table where the column is mapped to a column which is a primary key in the other table.

Here in our example, company _Id column is a Primary key in the Company table.

Usually when there is such kind of relation, the job usually fails if the value is not present in the Primary key table and tries to insert in the table with a foreign key.So the tester's duty is to check if the company_Id in the Company table exists when you are trying to insert any data in the other table where Company_Id is a foreign Key.

8.1.5 Check: This will check all the values if any supplied to the specific column and if the values are not present in the check list specified, will return an error.

If there is any check constraint on the table, check whether the data in the column meets the check condition, if not rise a defect.

8.1.6 Default: Check whether there is any default value specified in the column and check if there are no particular data that is not inserted then that column should be populated with the default value.

8.2 Source to Target Count Testing: In the Source to Target data is matched or not. A Tester can check in this view whether it is ascending order or descending order it doesn't matter .Only count is required for Tester.

Due to lack of time a tester can follow this type of Testing.

Query used for validating the data:

Select count(*) from company_stg
Select Count(*) from company

In our example, The flat file data is directly loaded in the staging table where we can query on the table, Otherwise see the count from the flat file only.

8.3 Source to Target Data Validation Testing: In this Testing, a tester can validate the each and every point of the source to target data.

Query used for validating the data

Select * from company_stg
Select * from company

Now validate whether all the data has flown from the Flat file to staging and then from Staging in to Target.

Write a except query for doing the data validation

Select * from Company_stgExceptSelect * from Company

Most of the financial projects, a tester can identify the decimal factors.

8.4 Field to Field Testing: In the field to field testing, a test engineer can identify that how much space is occupied in the database. The data is integrated in the table cum data types. Validate the data field by filed for at least some of the adhoc records

NOTE: To check the order of the columns and source column to target column.

8.5 Duplicate Check Testing: In this phase of ETL Testing, a Tester can face duplicate value very frequently so, at that time the tester follows database queries why because huge amount of data is present in source and Target tables.

Select count(*) FROM company GROUP BY company_id HAVING COUNT (*) >1;

Note:

- There are no mistakes in Primary Key or no Primary Key is allotted then the duplicates may arise.

- Sometimes, a developer can do mistakes while transferring the data from source to target at that time duplicates may arise.
- Due to Environment Mistakes also duplicates arise (Due to improper plug-in in the tool).

8.6 Incremental and Historical Process Testing: In the Incremental data, the historical data is not corrupted. When the historical data is corrupted then this is the condition where bugs raise.

This is where you need to test the already existing data and the new data that is entering in to the table. That is the reason we have put the Date_Inserted and Date_modified columns where you can see whether the existing data has been updated and the new data is inserted.

Select * from Company where date_inserted=getdate()

The above query gives you how many records that are inserted today.

Select * from Company where date_inserted=getdate()

The above query gives you how many records that are modified today.

8.7 Transformation Testing: At the time of mapping from source table to target table, Transformation is not in mapping condition, then the Test Engineer raises bugs.

8.8 Regression Testing: Code modification to fix a bug or to implement a new functionality which makes us to to find errors.

These introduced errors are called regression. Identifying for regression effect is called regression testing.

8.9 Retesting: Re executing the failed test cases after fixing the bug.

8.10 System Integration Testing: Integration testing: After the completion of programming process. Developer can integrate the modules there are 3 models

- Top Down
- Bottom Up
- Hybrid

Chapter 9 - Types of ETL Bugs:

9.1 User interface bugs/cosmetic bugs:

- Any Cosmetic errors that can occur if the data is related to any GUI application.
- Any bugs if the Navigation cannot be done correctly, may be the font size, alignment etc.

9.2 Input/output bugs:

- If the Valid values as per the data type are not being present in the Source and the Target tables.
- If there are any invalid values or the special characters present in the Input or in the Output table

9.3 Calculation bugs:

- Mathematical errors-If the calculations that needs to be performed re not in the correct format.
- Final output is wrong – If the Expected output is not present.

9.4 **Load condition bugs:-**

- Does not allow multiple users.
- Does not allows customer expected load.

9.5 Race condition bugs:

- System crash & hang.
- System cannot run client plat forms

9.6 Version control bugs:

- No logo matching.
- No version information available

This occurs usually in regression testing.

9.7 H/W bugs:

- Device is not responding to the application.

9.8 Source bugs:

- Mistakes in help documents.

9.9 **Performance Testing:** To test the Server response with different user loads. The Purpose of performance testing is to find bottle neck in the application. This is usually done with the loads of data to validate whether the ETL is working fine with no performance gaps and not for a longer time

ETL Performing Life cycle:

- **ETL Workflow requirements:**

In the Phase of work flow requirement, ETL Tester can identify the performing scenarios how to connect the database to server which environment supports the performance testing and to check the front end and back end environment and batch jobs, data merging, file system components finally reporting events.

- **Performing Objective:**

The performing objective is to start end to end performance testing most of the time performing objective will be decided by the client.

- **Performing Testing:**

To calculate the speed of the project, ETL Tester can test the Database level. The data base is loading the target properly or not. When ETL Developer doesn't loads the data in proper conditions then some damage is caused in the performance of the system.

- **Performance Tuning:**

It is a mechanism to get a fixed performance related issues as a Performance tester, we are going to give some suggest recommendations to tuning department.

The below is the Point of Contact in case of any discrepancy

Error at Level	Point of Contact
Code Level	Developer
Data Base Level	DBA
Network Level	Administrator
System Level	S/A
Server Level	Server side People

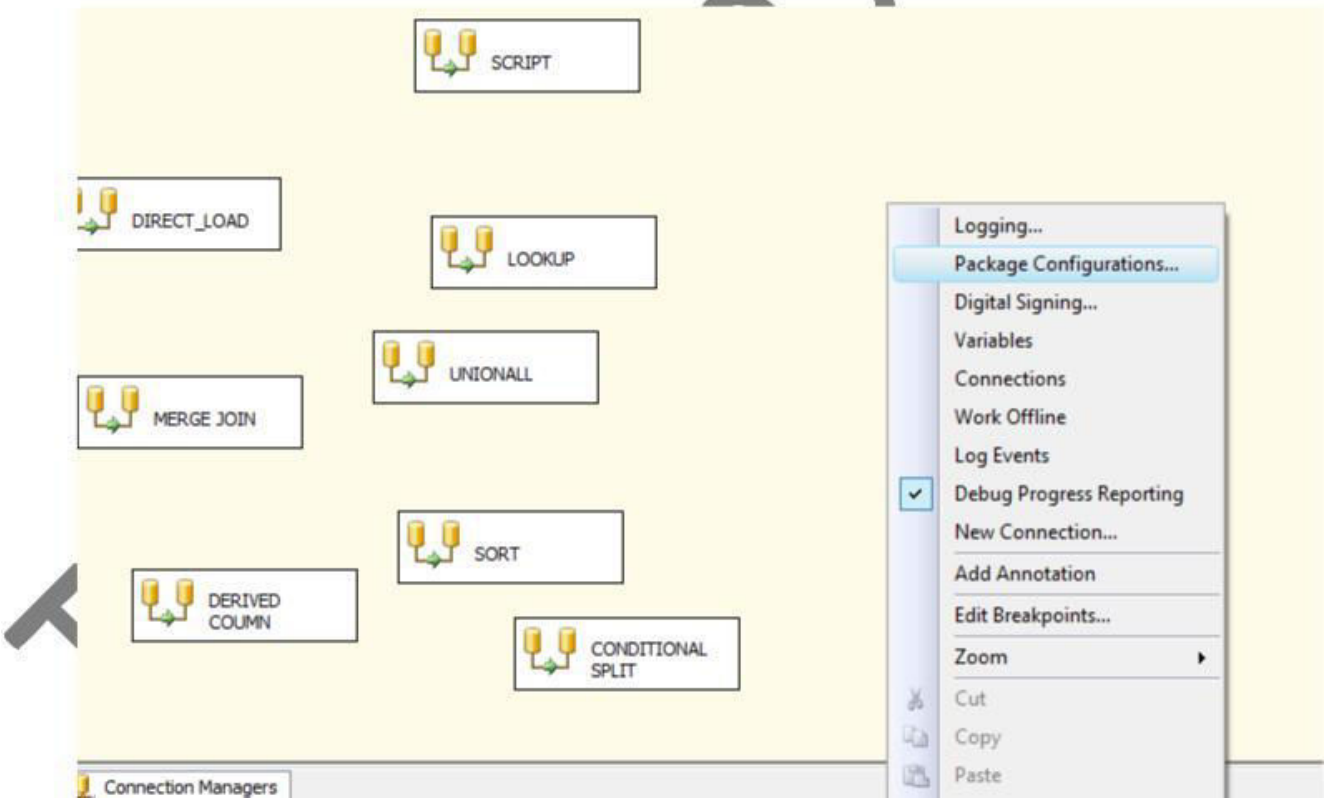
4. Creation of Configuration File in SSIS Package

Configuration file is usually an XML file that was created with .dtsconfig extension, which is mainly used when the packages are being transferred into multiple servers and handled. When these is a change of the server for different environments like (DEV,QA,UAT,PROD), It is not always necessary that every environment will have the same folder structure or the same name of the database and the server name will obviously be different

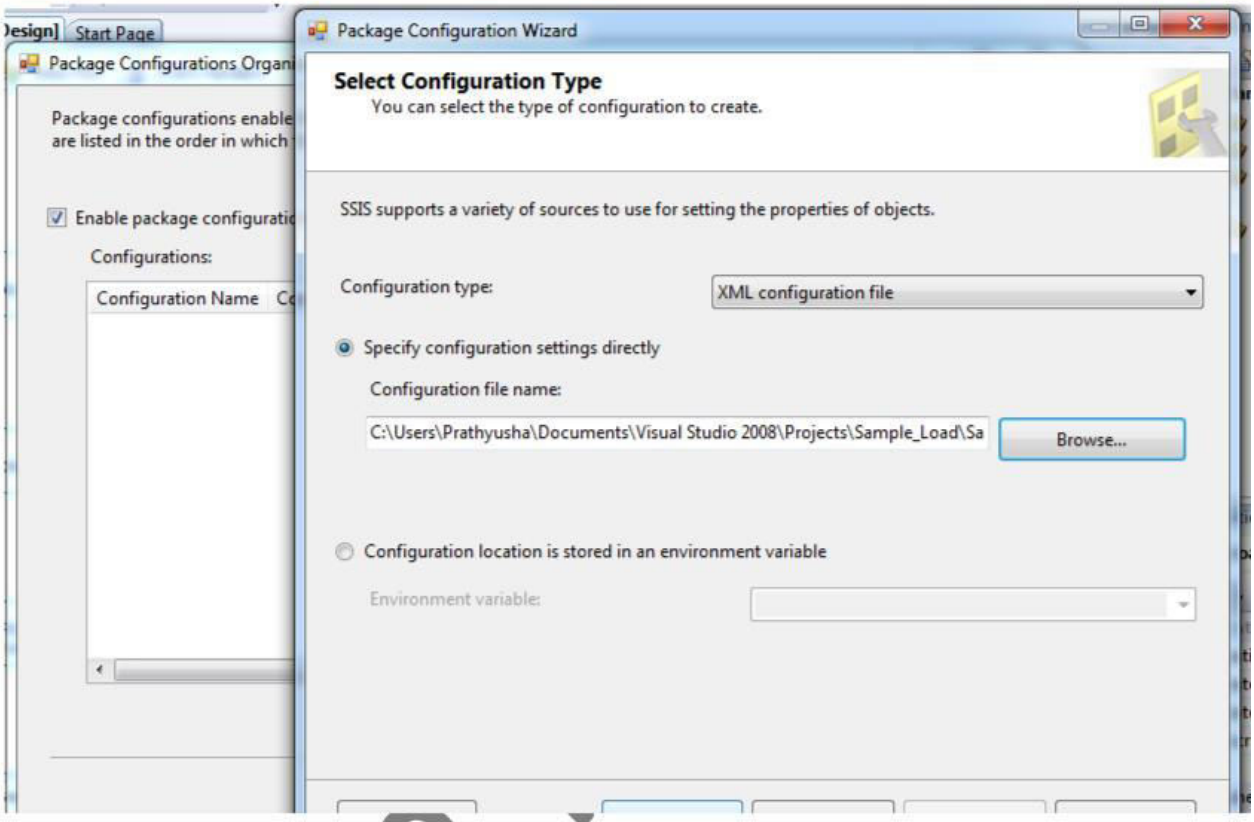
So the configuration file proves more effective when used in the following scenarios

- When the instance name is different
- When the Database name is different
- When the Login ID are different
- When the folder structure is different

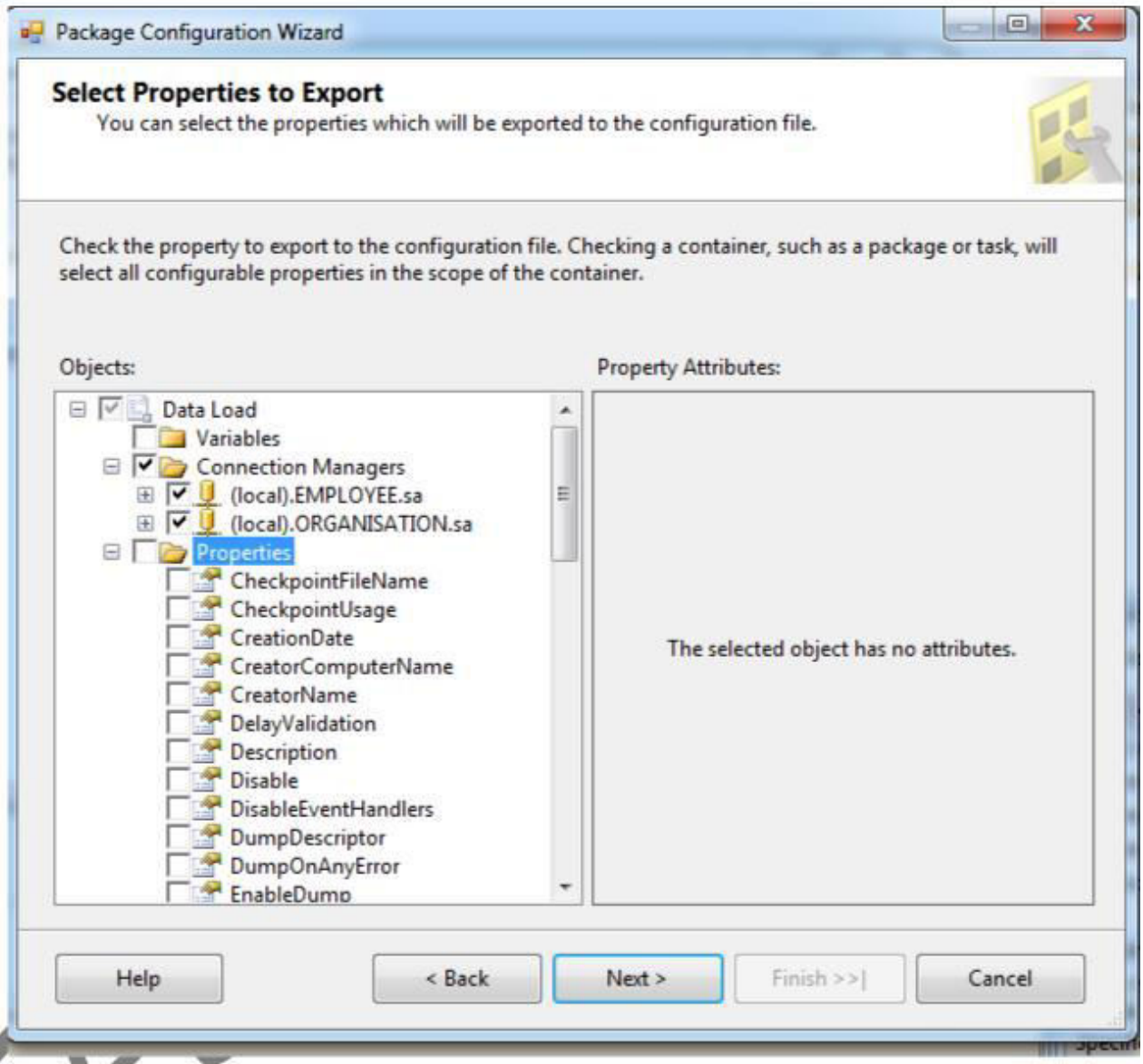
iv. Right click on the Control flow and select Package Configurations

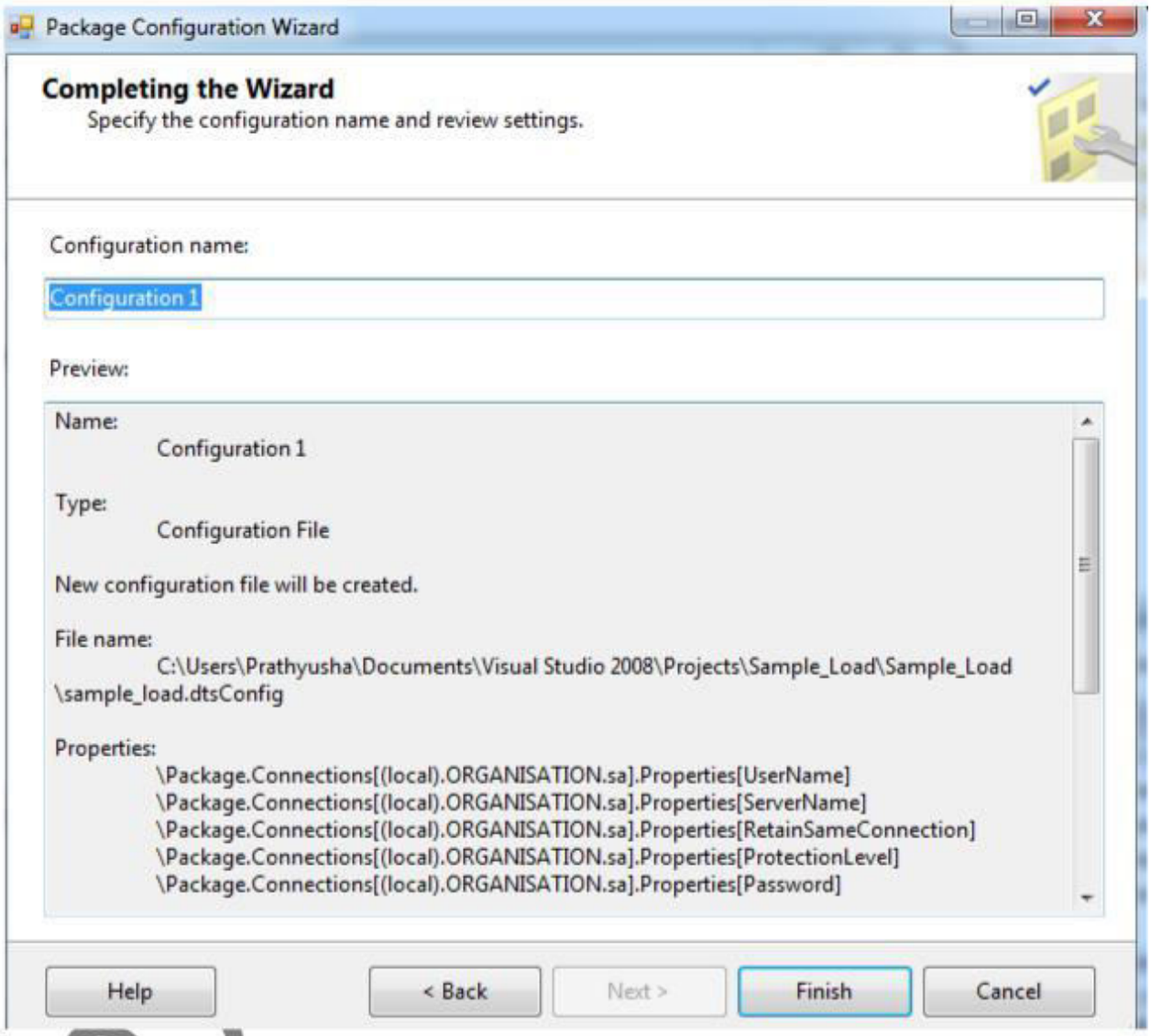


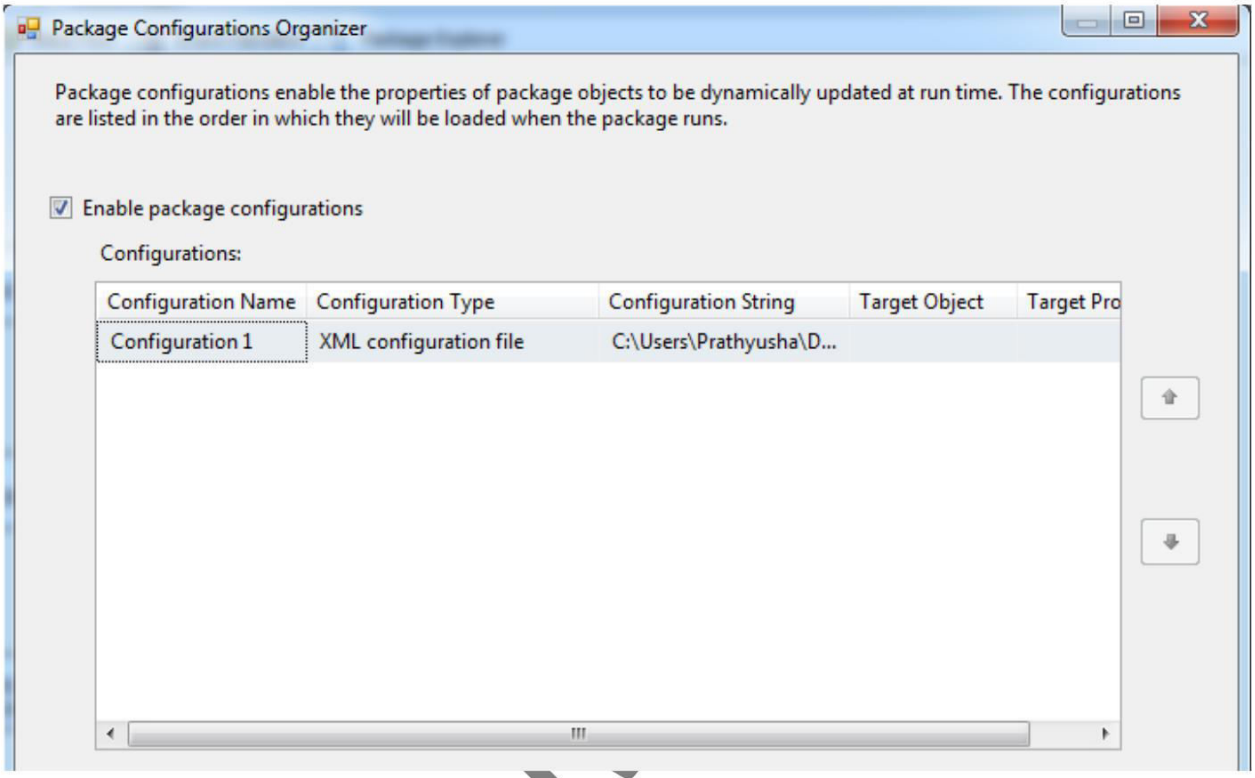
v. Enable Package configurations and click on Add button and then browse a path for creation of the file.



vi. Click on Next and select the tasks on which the config file needs to be created, Usually we select the Connection Managers and the Variables if any and click on Finish and then the config will be created in the path given


Package Configuration Wizard
Select Properties to Export
You can select the properties which will be exported to the configuration file.
Check the property to export to the configuration file. Checking a container, such as a package or task, will select all configurable properties in the scope of the container.
Objects:
Property Attributes:
Data Load
Variables
Connection Managers
(local).EMPLOYEE.sa
(local).ORGANISATION.sa
Properties
CheckpointFileName
CheckpointUsage
CreationDate
CreatorComputerName
CreatorName
DelayValidation
Description
Disable
DisableEventHandlers
DumpDescriptor
DumpOnAnyError
EnableDump
The selected object has no attributes.
Help
< Back
Next >
Finish >>|
Cancel

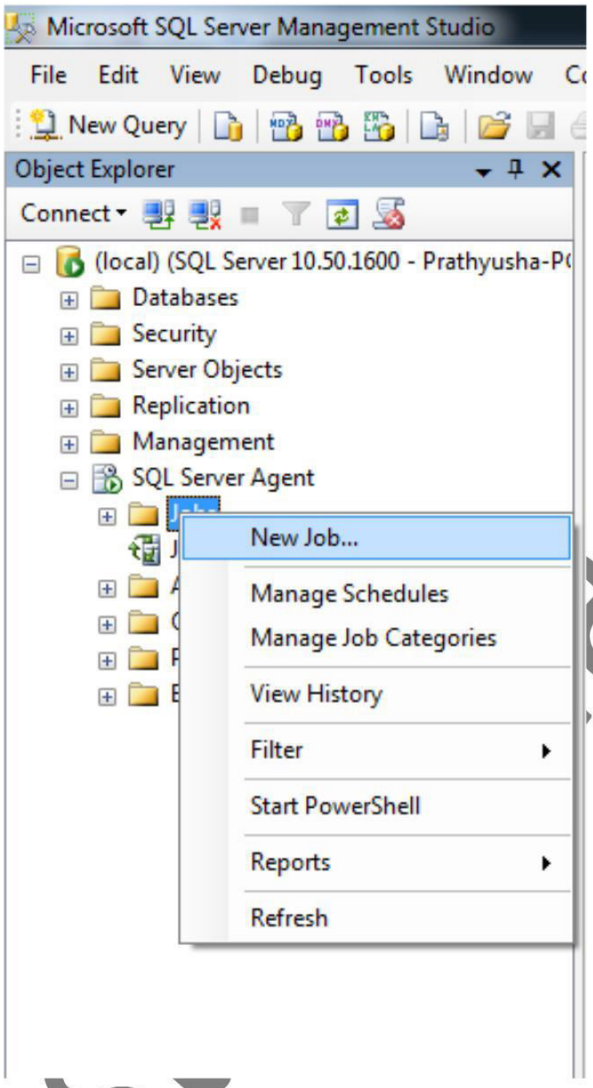

Package Configuration Wizard
Completing the Wizard
Specify the configuration name and review settings.
Configuration name:
Configuration 1
Preview:
Name:
Configuration 1
Type:
Configuration File
New configuration file will be created.
File name:
C:\Users\Prathyusha\Documents\Visual Studio 2008\Projects\Sample_Load\Sample_Load\sample_load.dtsConfig
Properties:
\Package.Connections[(local).ORGANISATION.sa].Properties[UserName]
\Package.Connections[(local).ORGANISATION.sa].Properties[ServerName]
\Package.Connections[(local).ORGANISATION.sa].Properties[RetainSameConnection]
\Package.Connections[(local).ORGANISATION.sa].Properties[ProtectionLevel]
\Package.Connections[(local).ORGANISATION.sa].Properties[Password]
Help
< Back
Next >
Finish
Cancel



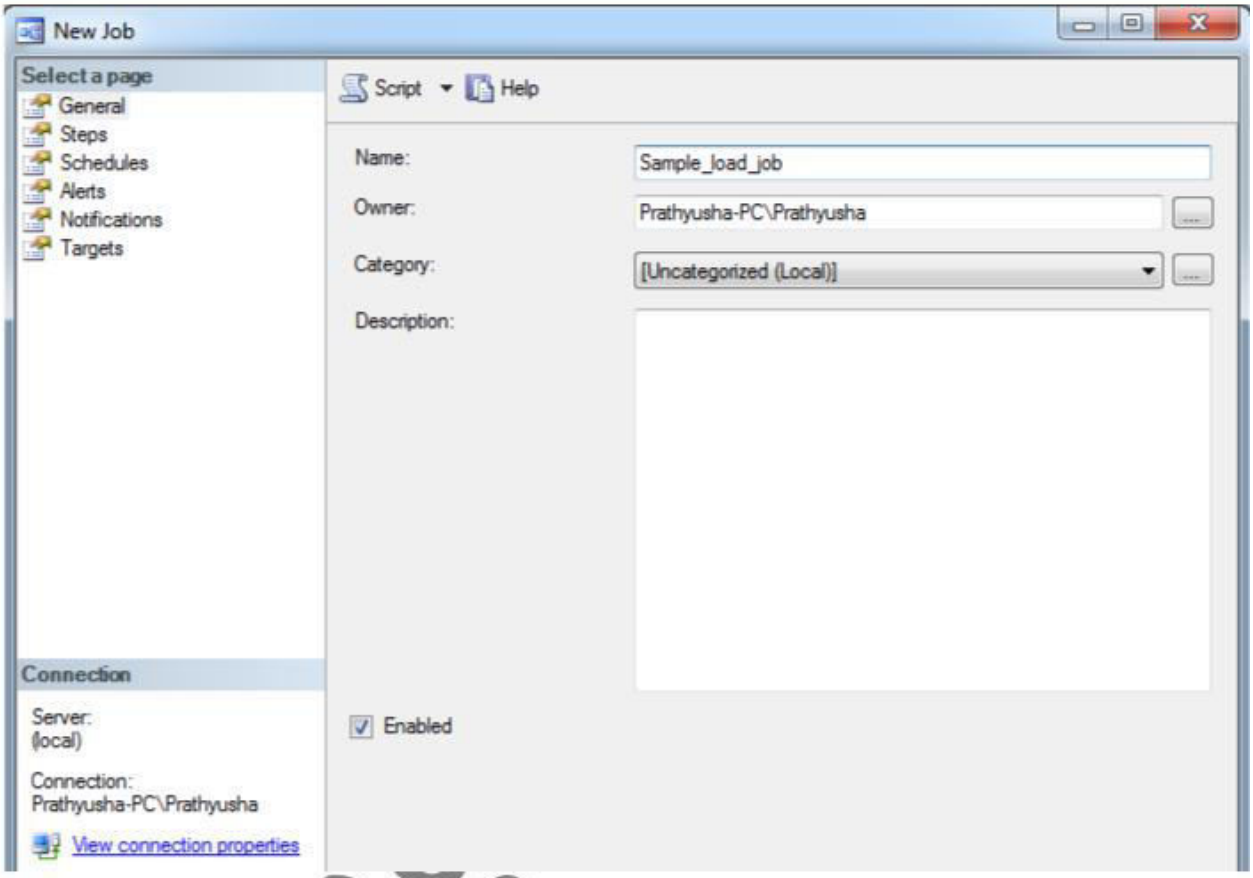
5. Creation of a Sql Server job for SSIS package

viii. Open the Sql Server management Studio and click on the SQL Server Agent which is shown on the left side panel in the Object Explorer

ix. Expand it and right click on the jobs to create a new job



x. Provide the Name for the Job

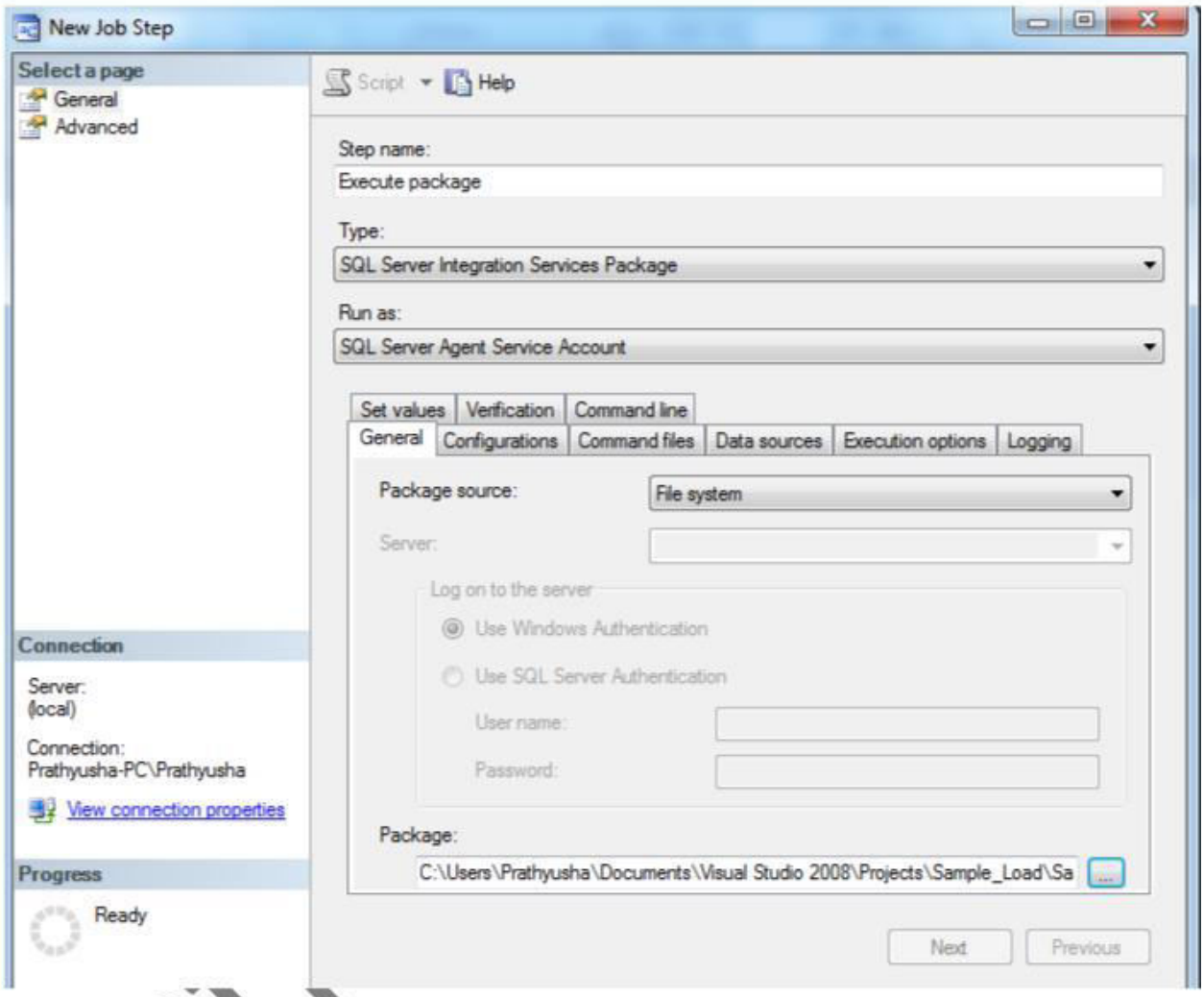


xi. Click on the Steps in the left side panel on the same window and click on the New Button below

xii. Write the Step name as required and Select Type as Sql Server Integration Services Package.

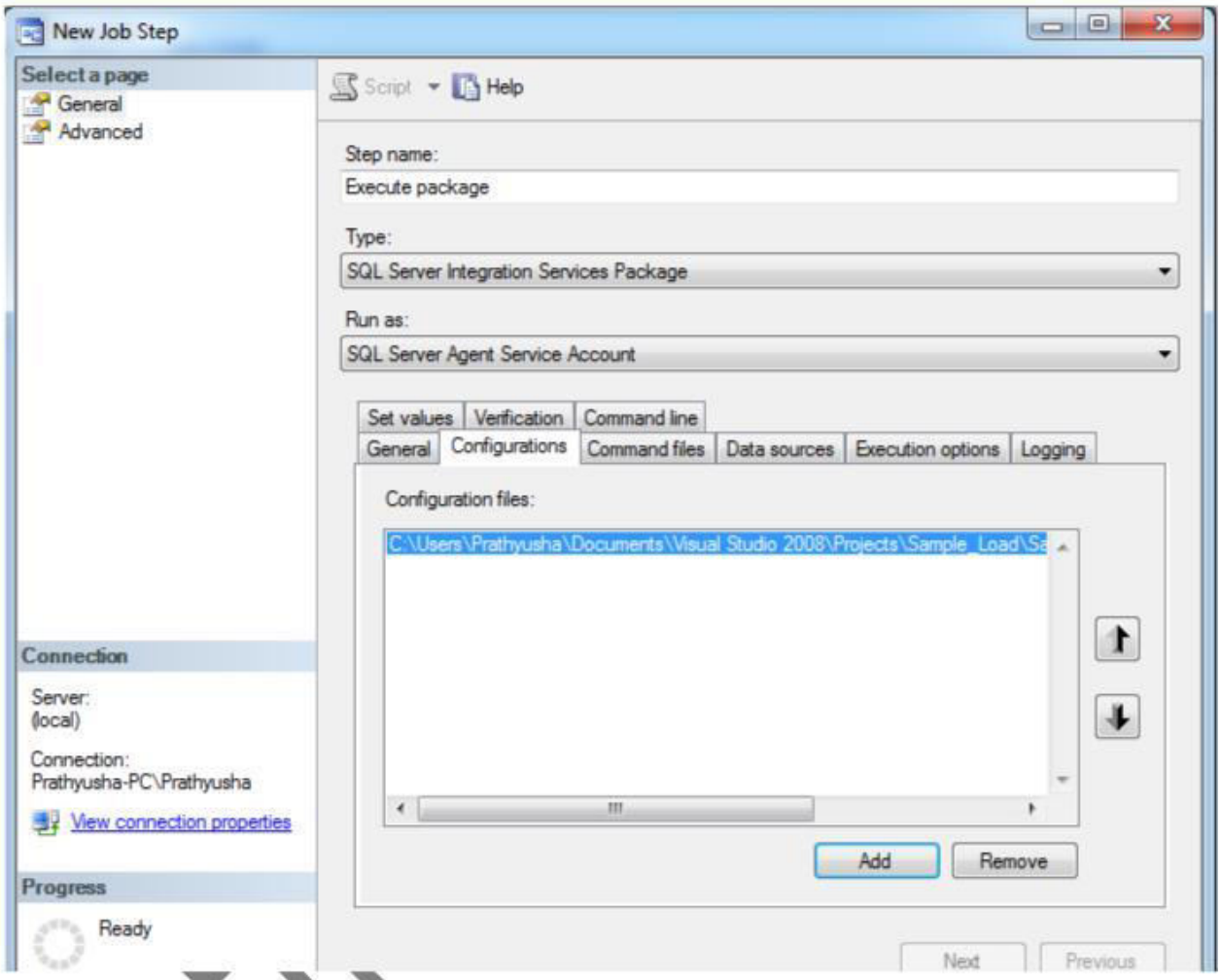
Select the package Source as the File system because all our packages would be stored in the Folder structure

Browse the package path below



xiii. Click on the Configurations tab on the same window and add a config file to that by browsing the configuration file already created for the package

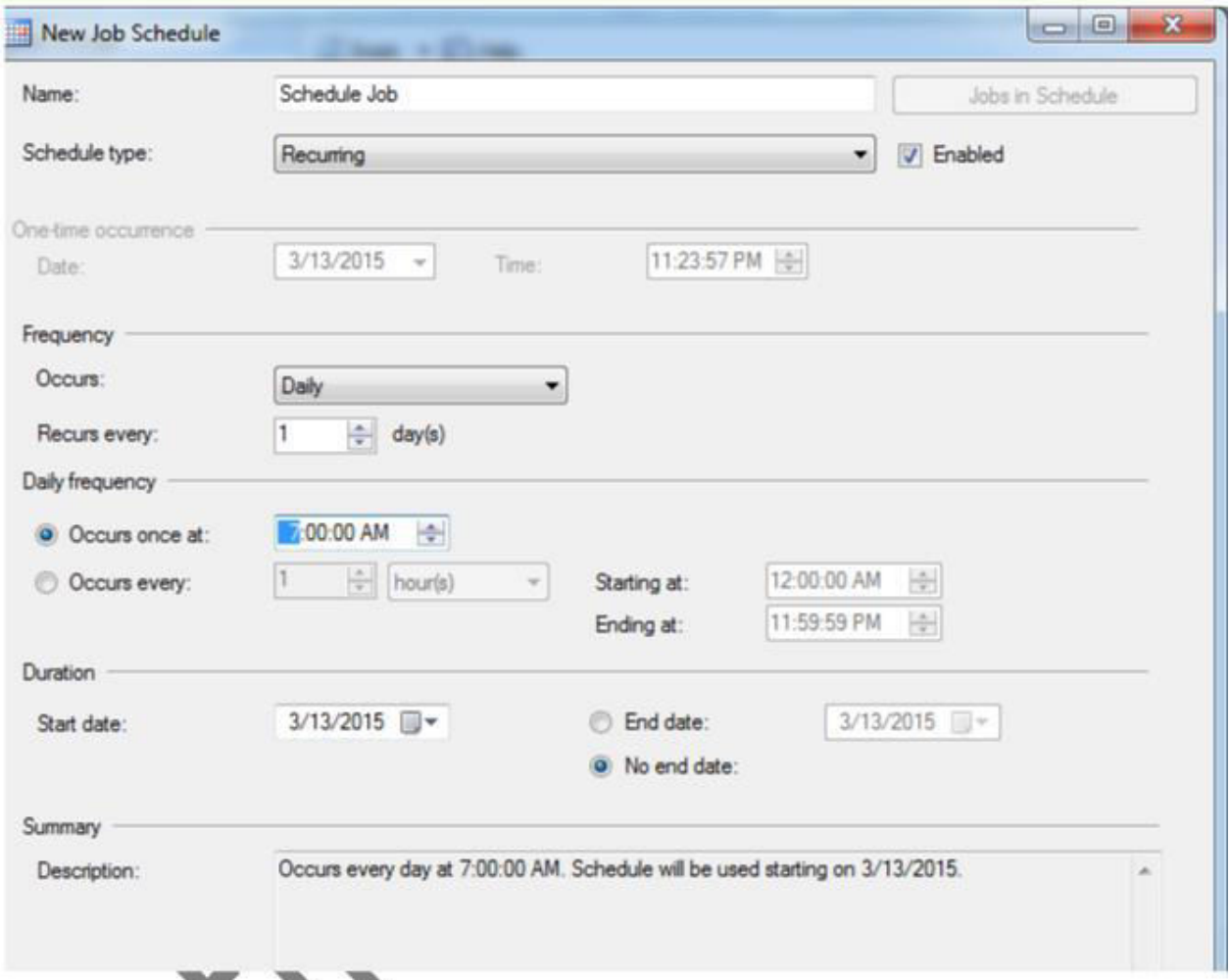
Click on OK.



xiv. Select the Schedules in the first window opened to create an job run schedule which executes on itself for a specified time

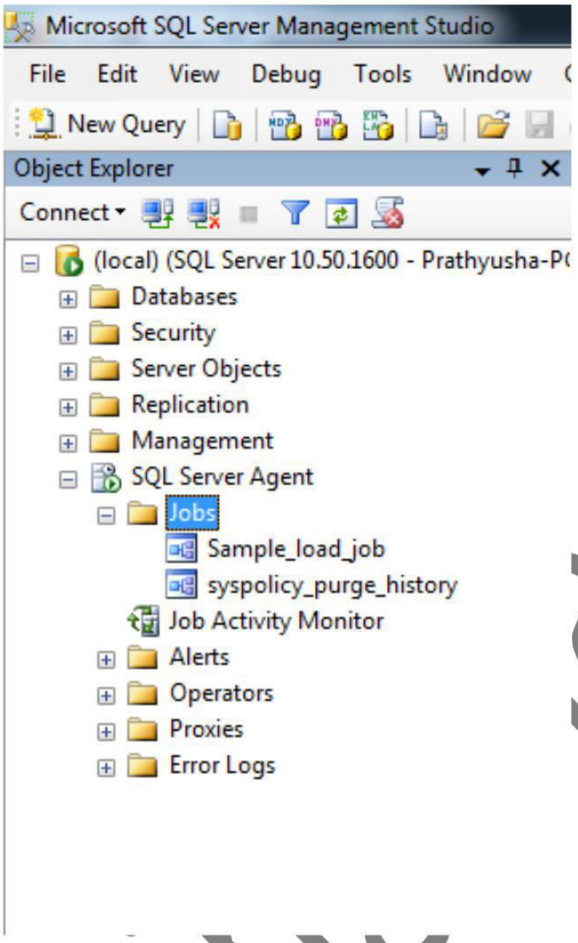
Create a New schedule for the job, Provide the name for the schedule, select whether daily or monthly and the date and time when it has to run

Click on OK and the job is now ready

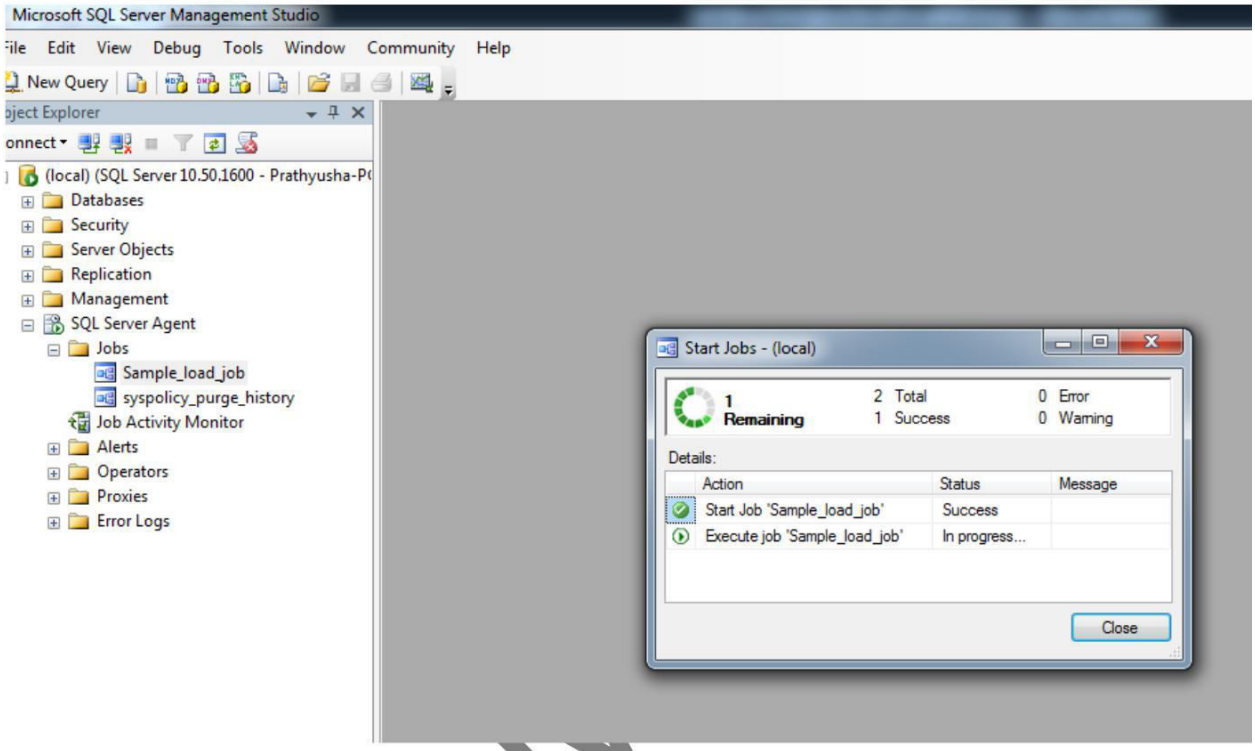


6. **Execution of Job**

v. You can view the Jobs in the Sql Server agent in the left side panel.



vi. Right click on the Job and select start at step and the job gets on executing



vii. It returns success message if it passes and a Failure if it fails which can identified either through the message or with the Green or Red color respectively

viii. Right Click on the job and select View history to see whether the Job has executed Successfully or not and you can also view the message in the below panel if the job failed and know the reason for failure and raise a defect