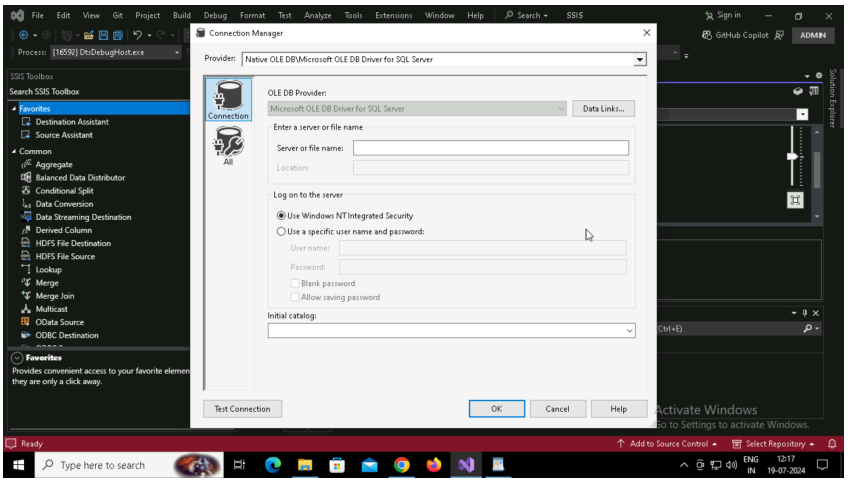
ASSIGNMENT SSIS

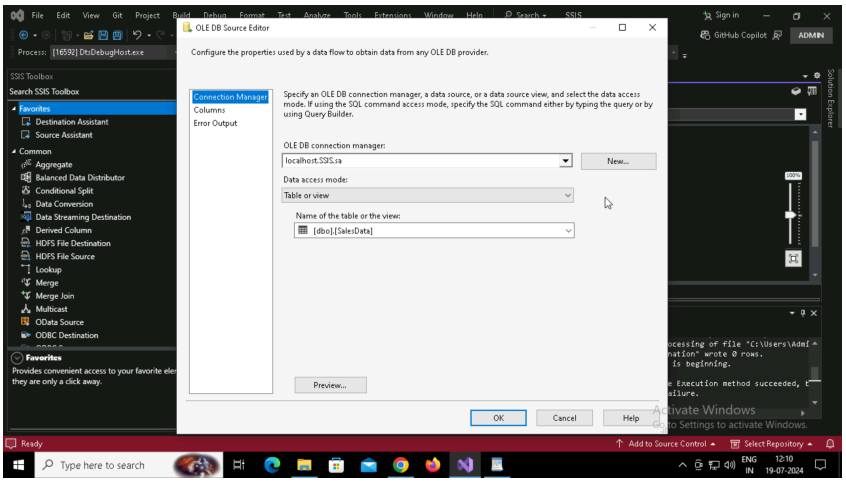
Task 1: Integration with ETL Data Warehouse (DWH)

Scenario: Your company has a data warehouse designed to consolidate data from various sources for analytical purposes. You need to create an SSIS package that extracts data from a transactional database and loads it into the data warehouse.

1. Create a Connection Manager to connect to the transactional database and the data warehouse.



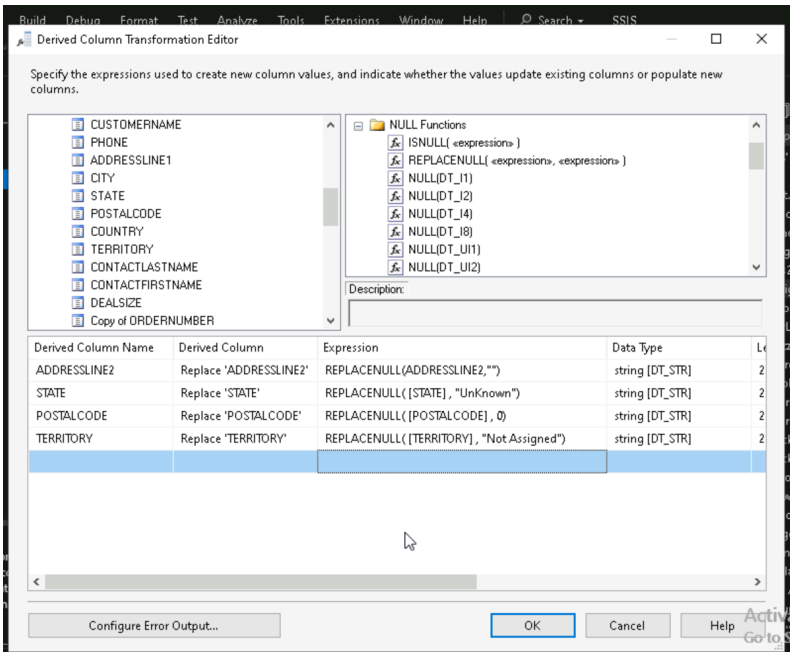
2. Extract Data from a transactional table (e.g., SalesData) using an OLE DB Source



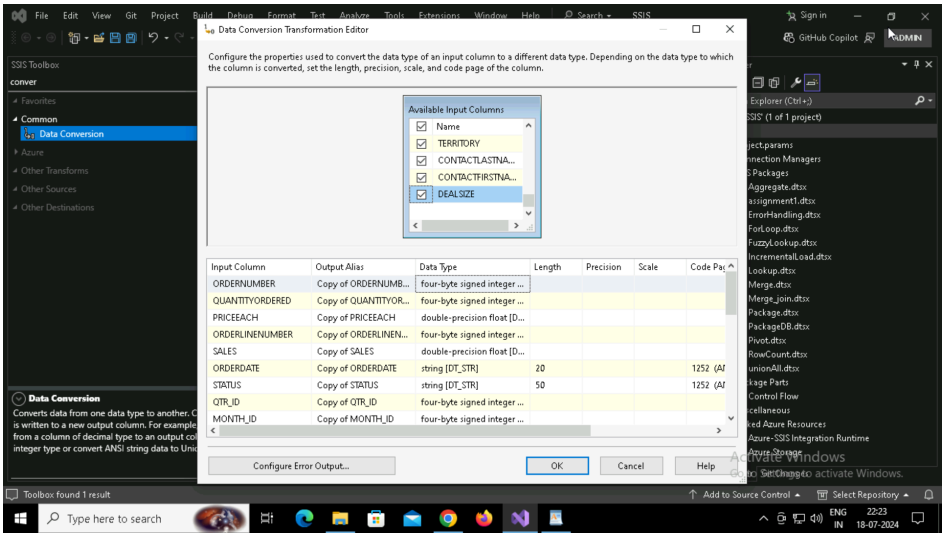
3. Transform Data:
   Apply necessary transformations such as data type conversions, data cleansing, and calculations.

```
PRODUCTCODE          0
CUSTOMERNAME         0
PHONE                0
ADDRESSLINE1         0
ADDRESSLINE2      2521
CITY                 0
STATE             1486
POSTALCODE          76
COUNTRY              0
TERRITORY         1074
CONTACTLASTNAME      0
CONTACTFIRSTNAME     0
DEALSIZE             0
dtype: int64
```

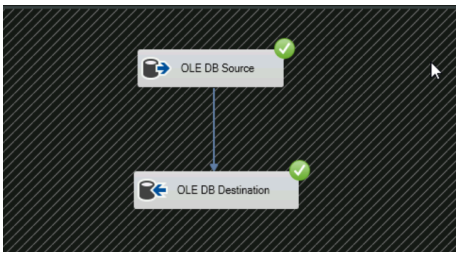
Null Values in addressline2, state,postalcode and territory,
So Replace Addressline2 with blank,
State with ‘Unknown’, postalcode with 0,
And territory with ‘Not Assigned’



Change datatype using data conversion



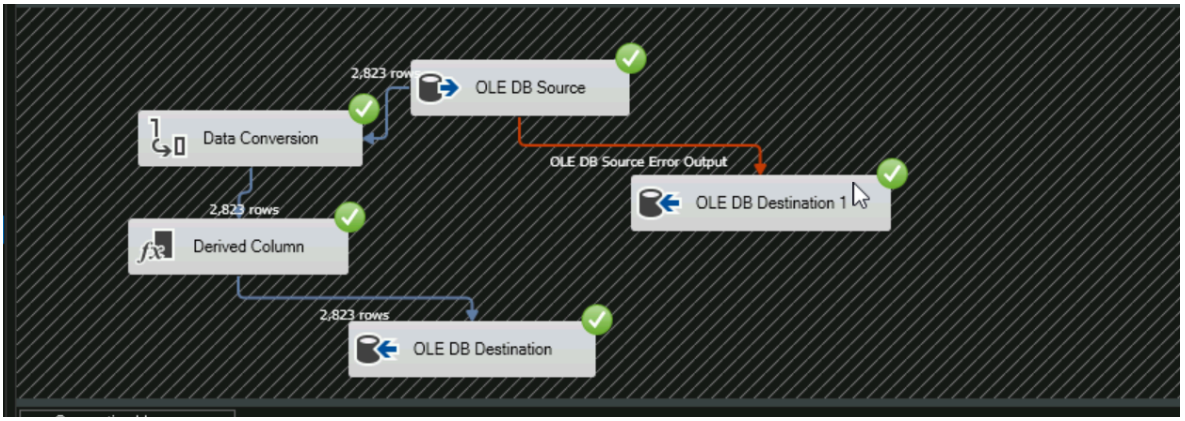
4. Load Data into the data warehouse (e.g., FactSales table).



Task 2: Data Warehouse Migrations

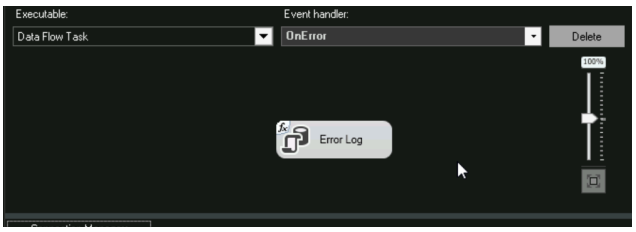
Scenario: Your organization is migrating its data warehouse from one server to another. You need to create an SSIS package that facilitates this migration.

1. Create Connection Managers for both the source and destination data warehouses.
2. Transfer Data from the source data warehouse to the destination using the Data Flow Task.

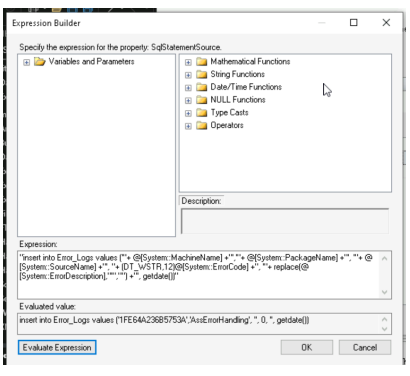


3. Ensure Data Integrity:
   a) Include checks and balances to ensure data is correctly migrated.
   b) Log the success or failure of the migration process.

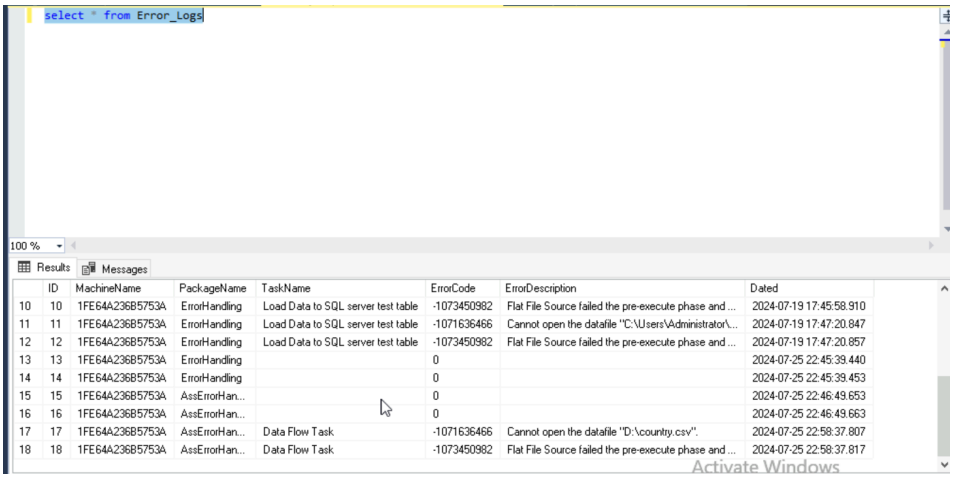
   a)Add Error Log in Event Handler



   b)Create expression



If any error then it shows in created error_logs table



```
select * from Error_Logs
```

100 %

Results | Messages

	ID	MachineName	PackageName	TaskName	ErrorCode	ErrorDescription	Dated
10	10	1FE64A236B5753A	ErrorHandling	Load Data to SQL server test table	-1073450982	Flat File Source failed the pre-execute phase and ...	2024-07-19 17:45:58.910
11	11	1FE64A236B5753A	ErrorHandling	Load Data to SQL server test table	-1071636466	Cannot open the datafile "C:\Users\Administrator\...	2024-07-19 17:47:20.847
12	12	1FE64A236B5753A	ErrorHandling	Load Data to SQL server test table	-1073450982	Flat File Source failed the pre-execute phase and ...	2024-07-19 17:47:20.857
13	13	1FE64A236B5753A	ErrorHandling		0		2024-07-25 22:45:39.440
14	14	1FE64A236B5753A	ErrorHandling		0		2024-07-25 22:45:39.453
15	15	1FE64A236B5753A	AssErrorHan...		0		2024-07-25 22:46:49.653
16	16	1FE64A236B5753A	AssErrorHan...		0		2024-07-25 22:46:49.663
17	17	1FE64A236B5753A	AssErrorHan...	Data Flow Task	-1071636466	Cannot open the datafile "D:\country.csv".	2024-07-25 22:58:37.807
18	18	1FE64A236B5753A	AssErrorHan...	Data Flow Task	-1073450982	Flat File Source failed the pre-execute phase and ...	2024-07-25 22:58:37.817

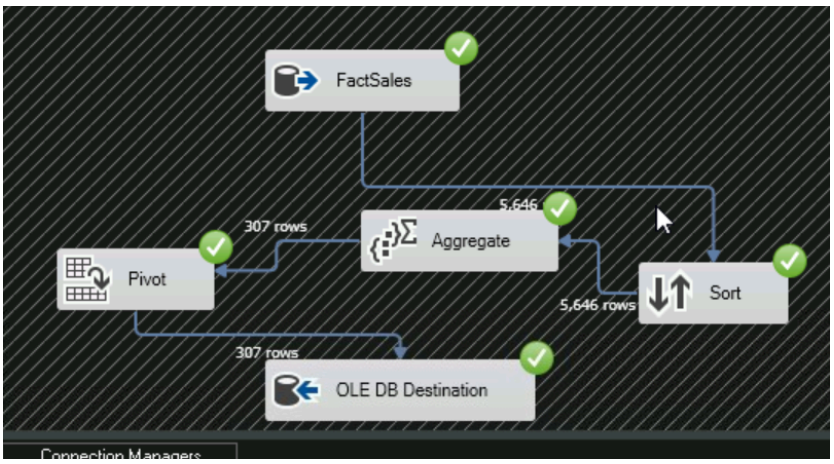
Activate Windows

Task 3: Implementing a Pivot Transformation

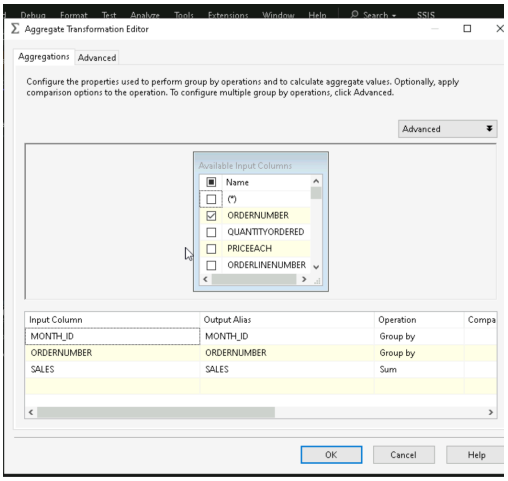
Scenario: You have data in a normalized format and need to pivot it for reporting purposes.

Requirements:

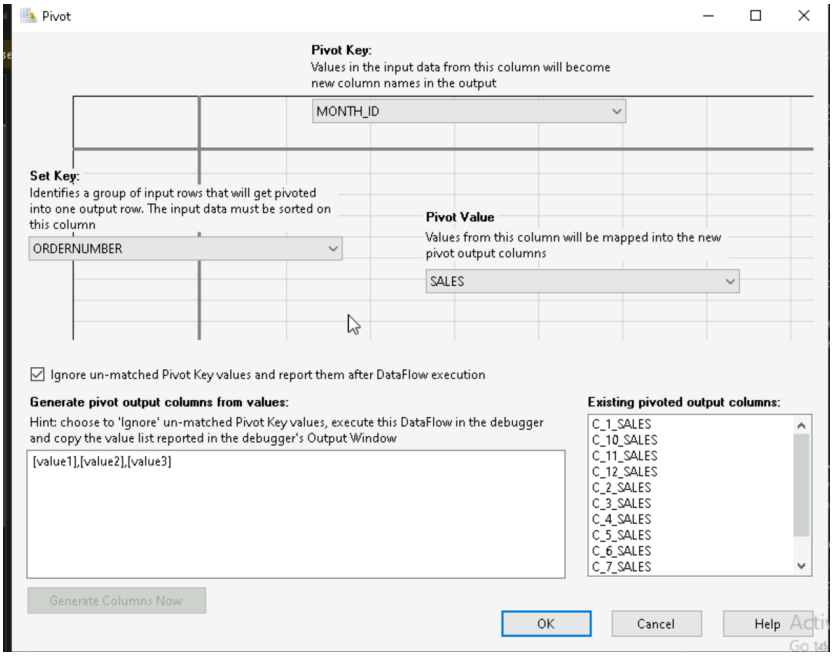
1. Extract Data from the source table using an OLE DB Source.
2. Apply a Pivot Transformation to transform the normalized data into a pivoted format.
3. Load the Pivoted Data into a destination table



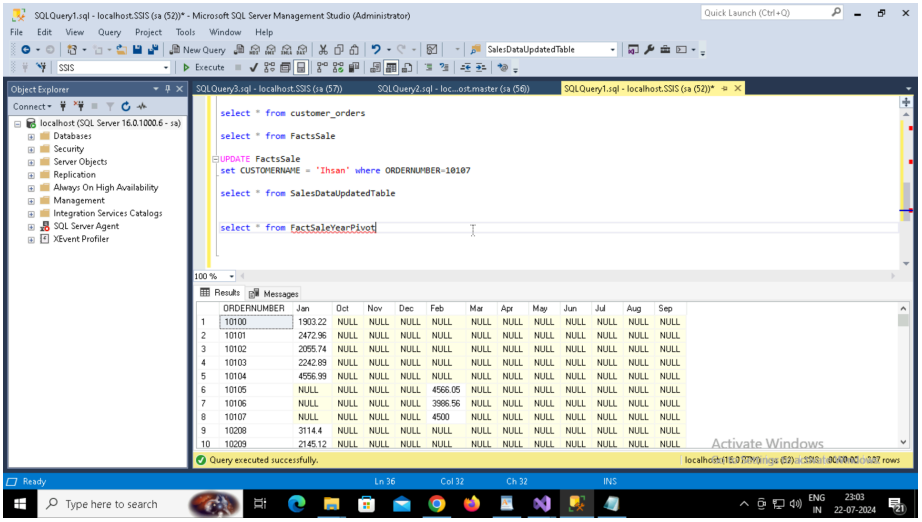
Step1 : Read Sales table
Step2: sort according to ordernumber
Step3 : Aggregate using ordernumber,year and month



Step 4 : Pivot Table Changes



Output



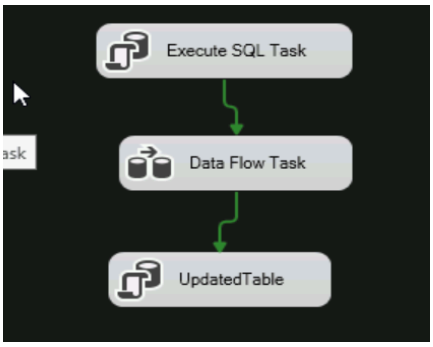
Task 4: Incremental Load

Scenario: To optimize ETL processes, you need to implement an incremental load to update only the changed data in the data warehouse.

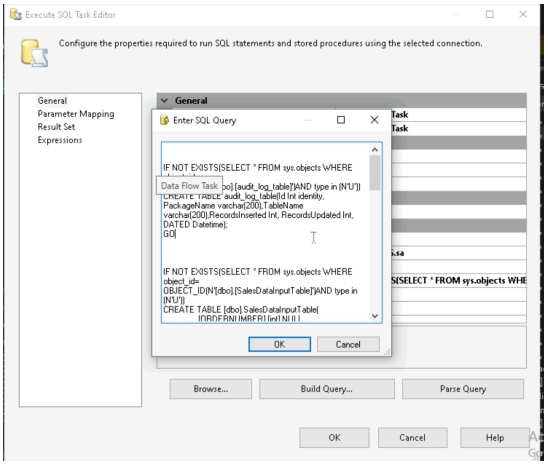
Requirements:

1. Identify Changed Data: Use methods such as timestamps, change data capture using lookup, or checksums.
2. Extract Only the Changed Data from the source.
3. Update the Data Warehouse with the new and changed data only

Control Flow

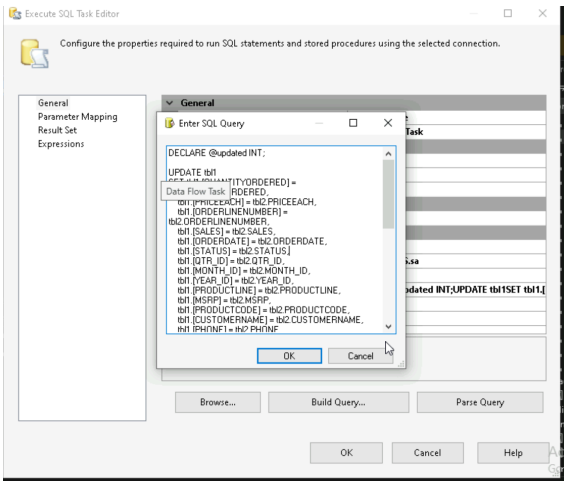


Step 1: Execute SQL task for creating input table, updated table and log table

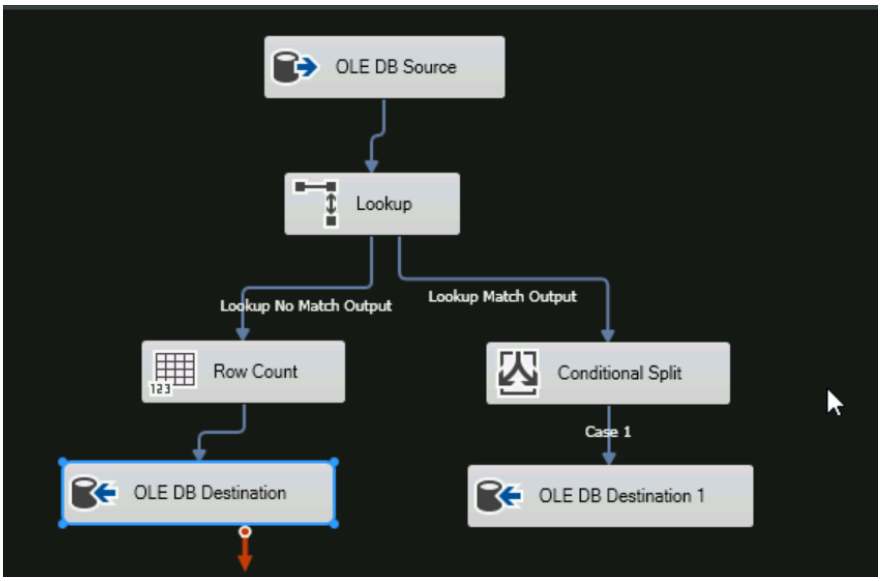


Step 2: Data Flow Task

Step3 :SQL task for add and update Updated Table

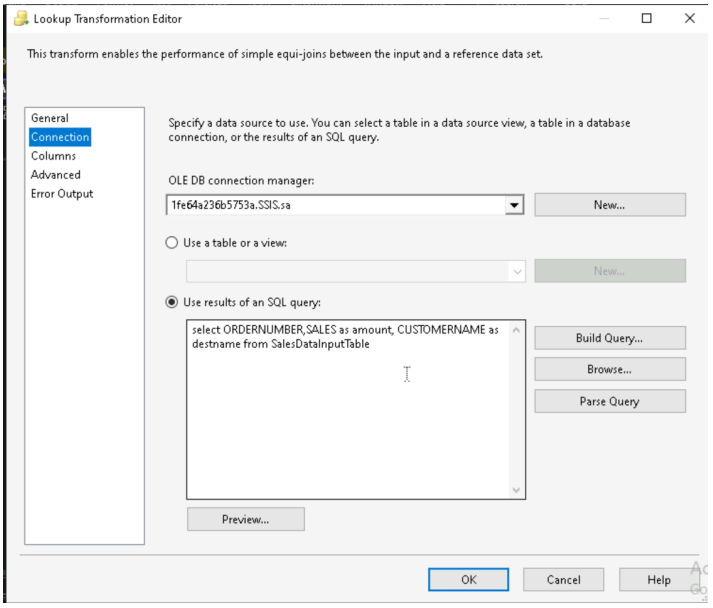


Data Flow Task



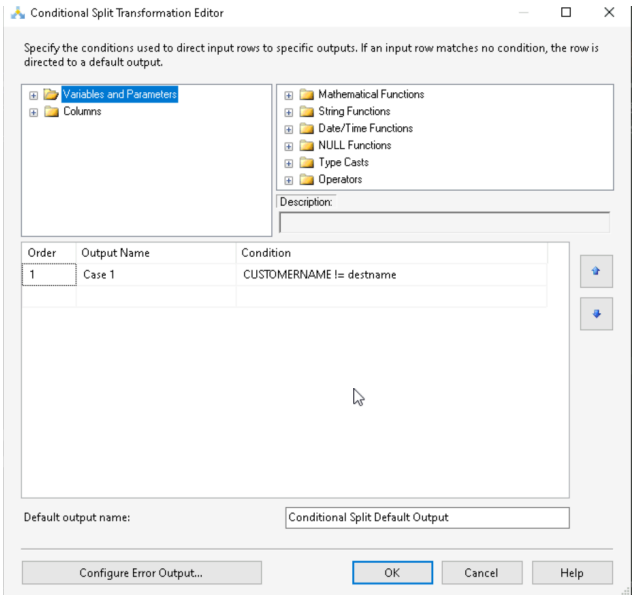
Step 1 : Read Table From Database

Step2 : Add Lookup connection Query (here updating customername column)



Step3 :

a) Conditional split for lookup matching output



b) No lookup match output go to row count(for that we need to create new variable)

Using variable calculate row count

Step 4 : Destination Tables

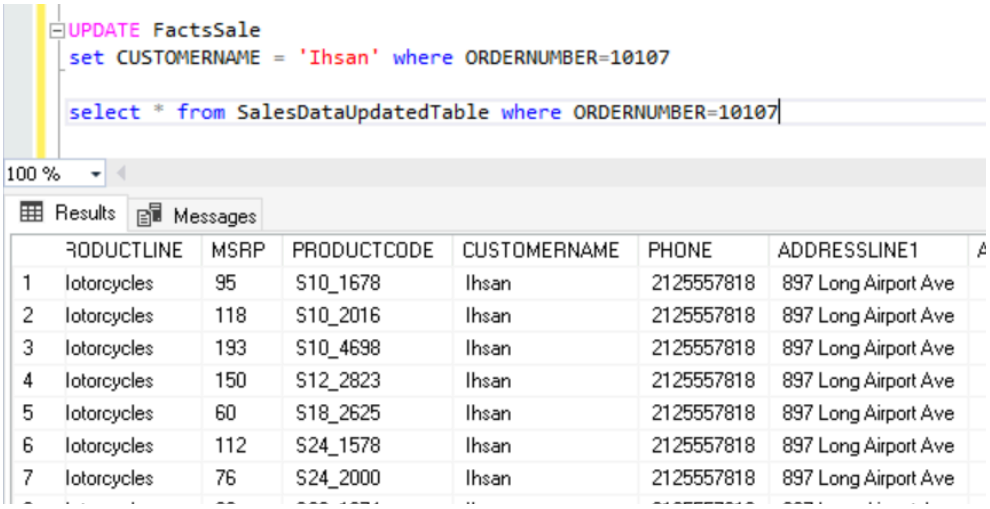
a) Right(lookup match) data goes to updatetable that contain the latest and previous updates
b) Left(No Lookupmatch) data goes to inputtable that update the latest changes and it contain the latest data only

c) Log table contain all the tracks of updating and adding data to table with timestamps

Output

After first run all the data goes to left inputtable and after making changes and run the changed data goes to right updatetable(only contain changed data) and new inputtable updated

Here is the UpdatedTable



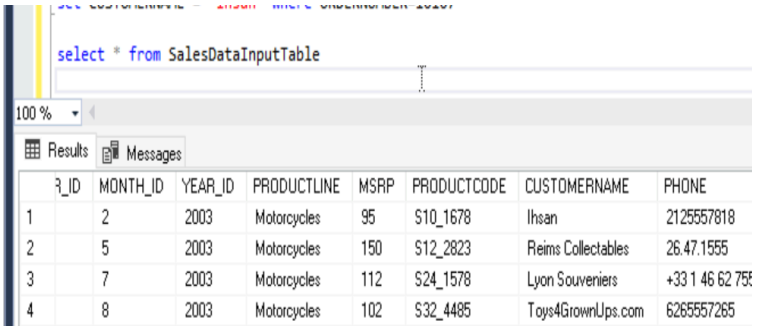
```
UPDATE FactsSale
set CUSTOMERNAME = 'Ihsan' where ORDERNUMBER=10107

select * from SalesDataUpdatedTable where ORDERNUMBER=10107
```

100 %

Results | Messages

	RODUCTLINE	MSRP	PRODUCTCODE	CUSTOMERNAME	PHONE	ADDRESSLINE1
1	lotorcycles	95	S10_1678	Ihsan	2125557818	897 Long Airport Ave
2	lotorcycles	118	S10_2016	Ihsan	2125557818	897 Long Airport Ave
3	lotorcycles	193	S10_4698	Ihsan	2125557818	897 Long Airport Ave
4	lotorcycles	150	S12_2823	Ihsan	2125557818	897 Long Airport Ave
5	lotorcycles	60	S18_2625	Ihsan	2125557818	897 Long Airport Ave
6	lotorcycles	112	S24_1578	Ihsan	2125557818	897 Long Airport Ave
7	lotorcycles	76	S24_2000	Ihsan	2125557818	897 Long Airport Ave

Here is the InputTable



```
select * from SalesDataInputTable
```

100 %

Results | Messages

	R_ID	MONTH_ID	YEAR_ID	PRODUCTLINE	MSRP	PRODUCTCODE	CUSTOMERNAME	PHONE
1		2	2003	Motorcycles	95	S10_1678	Ihsan	2125557818
2		5	2003	Motorcycles	150	S12_2823	Reims Collectables	26.47.1555
3		7	2003	Motorcycles	112	S24_1578	Lyon Souveniers	+33 1 46 62 75
4		8	2003	Motorcycles	102	S32_4485	Toys4GrownUps.com	6265557265

Task 5: Transformations

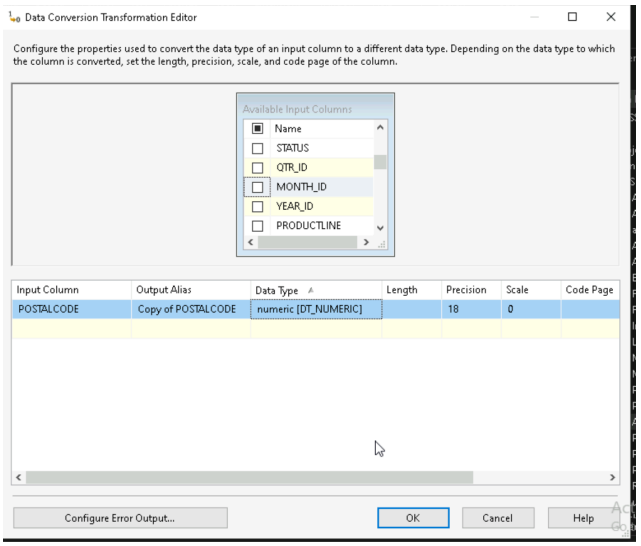
Scenario: Your company needs to transform raw data into a format suitable for reporting. You need to perform multiple transformations within an SSIS package.

Requirements:

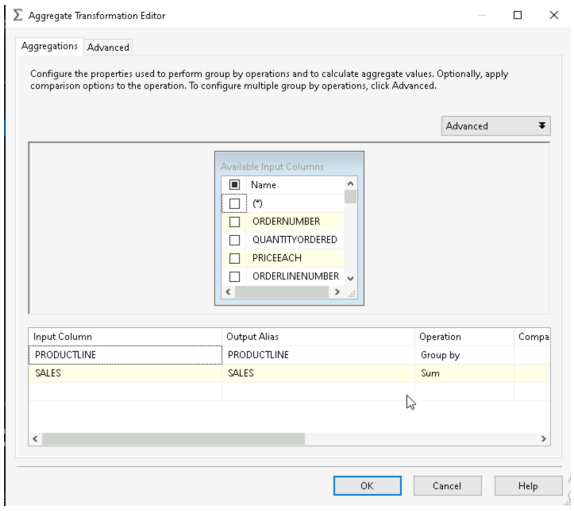
1. Extract Data from a source table using an OLE DB Source.
2. Apply Transformations such as:

    - Data Conversion

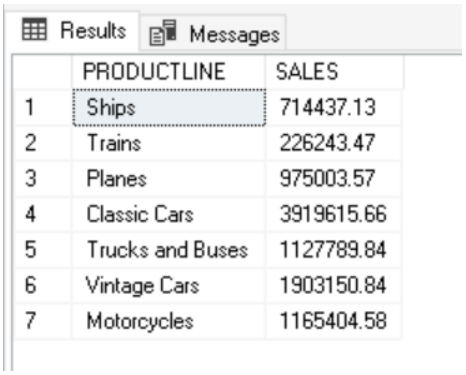
Using Data Conversion Change Postalcode String type to Numeric type



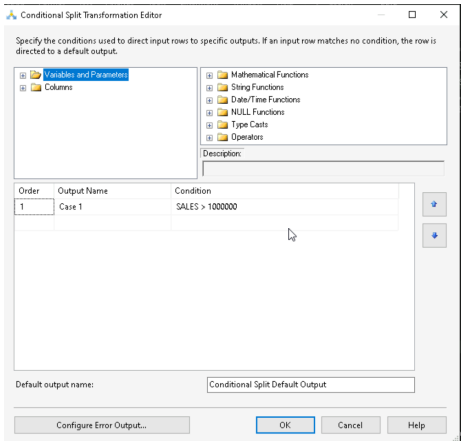
- Aggregate (by productline and sales)



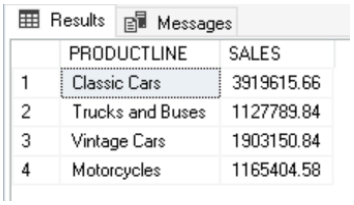
Output like this,



- Conditional Split



Add condition to the above aggregation then output is like this,

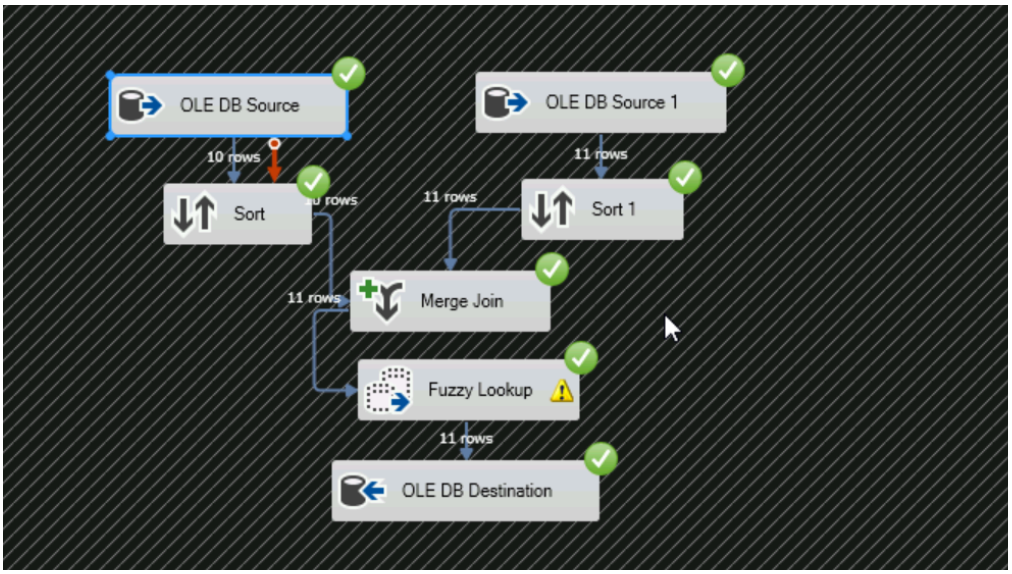


Task 6: MERGE & FUZZY LOOKUP

Scenario: You need to merge two datasets and use fuzzy matching to handle potential duplicates.

Requirements:

1. Extract Data from two source tables using OLE DB Sources.
2. Apply a Merge Join to combine the datasets based on a common key.
3. Use Fuzzy Lookup to identify and resolve duplicates in the merged data.
4. Load the Cleaned Data into a destination table.



Step 1 :Get Two source tables using OLE DB Sources.

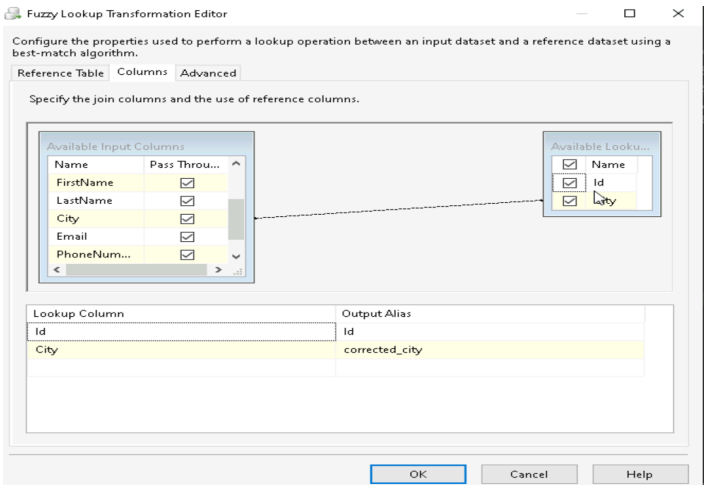
Step 2: Sort using CustomerID (Here the tables are customers and orders)

Step 3: Merge Both Using Common Column CustomerID

Step 4 : Add Fuzzy LookUp

Create reference table City_ref that contain all the cities with id

Using that city column match the given combined table



Here is The Updation and output

```
UPDATE customer
set city = 'Bang3loru' where CustomerID=1

UPDATE customer
set city = 'Mum bai' where CustomerID=3
```

City	Email	PhoneNumber	Id	corrected_city	_Similarity	_Confidence	_Similarity_City
Bang3loru	aarav.sharma@example.com	9876543210	2	Bangalore	0.7709482	0.4934723	0.7709482
Bang3loru	aarav.sharma@example.com	9876543210	2	Bangalore	0.7709482	0.4934723	0.7709482
Chennai	vihaan.reddy@example.com	8765432109	3	Chennai	1	1	1
Mum bai	aditya.patel@example.com	7654321098	8	Mumbai	0.8571429	0.9826712	0.8571429
Delhi	diya.agarwal@example.com	6543210987	4	Delhi	1	1	1

Task 7: Using Script Task

Scenario: You need to perform a complex data transformation that is not supported by the standard SSIS components. A Script Task can be used to achieve this.

Requirements:

1. Add a Script Task to the Control Flow.
2. Write a Script: that performs the required transformation. e.g. Reading data from a file, processing it, and writing the results to a database table.
3. Execute the Script Task within an SSIS package.

Solution

Step1 : Add a Script Task to the Control Flow

Step2 : Open Edit Script And Make changes in c# code for read the csv file and save it into

Sql table

```
public void Main()
{
    // TODO: Add your code here
    string sourceFilePath = Dts.Variables["User::SourceFilePath"].Value.ToString();
    string destinationTable = Dts.Variables["User::DestinationTable"].Value.ToString();
    string connectionString = Dts.Variables["User::ConnectionString"].Value.ToString();

    try
    {
        DataTable dataTable = new DataTable();

        using (StreamReader sr = new StreamReader(sourceFilePath))
        {
            string[] headers = sr.ReadLine().Split(',');

            foreach (string header in headers)
            {
                dataTable.Columns.Add(header);
            }

            while (!sr.EndOfStream)
            {
                string[] rows = sr.ReadLine().Split(',');
                DataRow dr = dataTable.NewRow();
                for (int i = 0; i < headers.Length; i++)
                {
                    dr[i] = rows[i];
                }
                dataTable.Rows.Add(dr);
            }
```

Activate Windows

```
            for (int i = 0; i < headers.Length; i++)
            {
                dr[i] = rows[i];
            }
            dataTable.Rows.Add(dr);
        }
    }

    using (SqlConnection sqlConnection = new SqlConnection(connectionString))
    {
        sqlConnection.Open();

        using (SqlBulkCopy sqlBulkCopy = new SqlBulkCopy(sqlConnection))
        {
            sqlBulkCopy.DestinationTableName = destinationTable;

            foreach (DataColumn column in dataTable.Columns)
            {
                sqlBulkCopy.ColumnMappings.Add(column.ColumnName, column.ColumnName);
            }

            sqlBulkCopy.WriteToServer(dataTable);
        }
    }

    Dts.TaskResult = (int)ScriptResults.Success;
}
catch (Exception ex)
{
    Dts.Events.FireError(0, "Script Task", ex.Message + "\r" + ex.StackTrace, String.Empty, 0);
```

Step3 : Create necessary Variables to specify Source file path, destination table and connection string

Variables

Name	Scope	Data type	Value
ConnectionString	Package6	String	Server=localhost;Database=SSIS;User Id=sa;Password=pass@word1;
DestinationTable	Package6	String	smallSale
SourceFilePath	Package6	String	D:\sale_small.csv

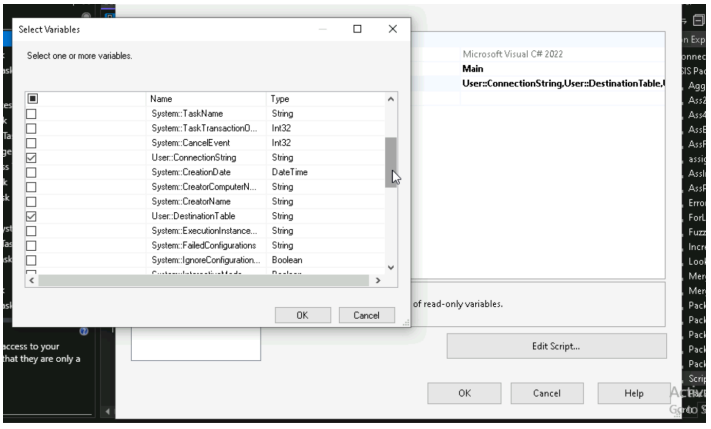
Step4 : Create Destination Table

```
CREATE TABLE smallSale (
    ORDERNUMBER VARCHAR(50),
    QUANTITYORDERED VARCHAR(50),
    PRICEEACH VARCHAR(50),
    ORDERLINENUMBER VARCHAR(50),
    SALES VARCHAR(50),
    ORDERDATE VARCHAR(50),
    STATUS VARCHAR(50),
    QTR_ID VARCHAR(50),
    MONTH_ID VARCHAR(50),
    YEAR_ID VARCHAR(50),
    PRODUCTLINE VARCHAR(50),
    MSRP VARCHAR(50),
    PRODUCTCODE VARCHAR(50),
    CUSTOMERNAME VARCHAR(100),
    PHONE VARCHAR(50),
    ADDRESSLINE1 VARCHAR(100),
    ADDRESSLINE2 VARCHAR(100),
    CITY VARCHAR(50),
    STATE VARCHAR(50),
    POSTALCODE VARCHAR(50),
    COUNTRY VARCHAR(50),
    TERRITORY VARCHAR(50),
```

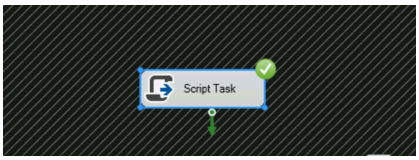
Results Messages

Query executed successfully.

Step 5: Mention variables in ReadOnlyVariable option in Script Task



OutPut




```
select * from smallSale

CREATE TABLE smallSale (
    ORDERNUMBER VARCHAR(50),
    QUANTITYORDERED VARCHAR(50),
    PRICEEACH VARCHAR(50),
    ORDERLINENUMBER VARCHAR(50),
    SALES VARCHAR(50),
    ORDERDATE VARCHAR(50),
    STATUS VARCHAR(50),
```

100 %

Results | Messages

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	STATUS	QTR_ID	MONTH_ID	YEAR_ID	PRODUCTLINE	MSRP
1	10107	30	95.7	2	2871	2/24/2003 0:00	Shipped	1	2	2003	Motorcycles	95
2	10121	34	81.35	5	2765.9	5/7/2003 0:00	Shipped	2	5	2003	Motorcycles	95
3	10134	41	94.74	2	3884.34	7/1/2003 0:00	Shipped	3	7	2003	Motorcycles	95
4	10145	45	83.26	6	3746.7	8/25/2003 0:00	Shipped	3	8	2003	Motorcycles	95
5	10159	49	100	14	5205.27	10/10/2003 0:00	Shipped	4	10	2003	Motorcycles	95

Activate Windows

Query executed successfully. localhost (16.0 RTM) sa (56) SSIS 00:00:00 5 rows