SRI LANKA INSTITUTE OF INFORMATION TECHNOLOGY

Data Warehouse and Business Intelligence

Assignment 01

2022

Submitted By:

Rajapaksha D.S.D

IT20012410

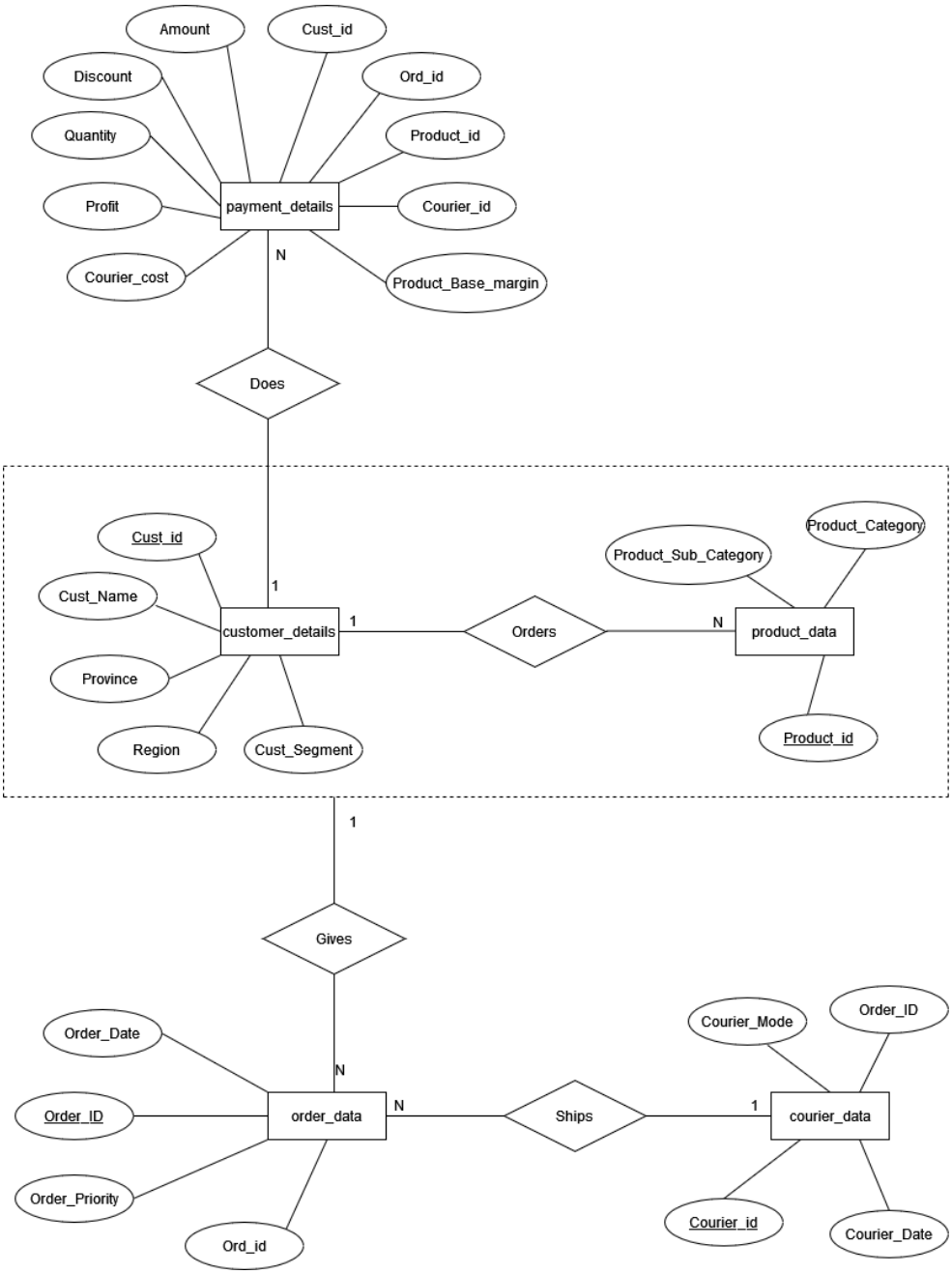
Table of Contents

1) Data set selection

I selected a data set of an online shopping system which includes the product details that the system has, the customer details, order details and courier details of the products which have ordered by the customers and the details of payments.

https://www.kaggle.com/datasets/tanyadayanand/online-shopping?select=shipping.csv

The ER Diagram for the chosen data set



This diagram shows the connection between the entities in the data set

2) Preparation of data sources

Table Name	Column Name	Data Type	Description
courier_data	Order_ID Courier_Mode Courier_Date Courier_id	int varchar (50) datetime string	Details of the courier of products
customer_details	Cust_Name Province Region Cust_Segment Cust_id	varchar (50) varchar (50) varchar (50) varchar (50) string	Details of the customer
order_data	Order_ID Order_Date Order_Priority Ord_id	int datetime varchar (50) string	Details of the product orders
payment_details	Ord_id Product_id Courier_id Cust_id Amount Discount Quantity Profit Courier_Cost Product_Base_Margin Payment_id Order_Date	string string string string float float int float float float int date	Details of payments
product_data	Product_Category Product_Sub_Category Product_id	varchar (50) varchar (50) string	Details of the product ordered by customers

From the above link, I received data sets in csv formats. The tables are courier_data, customer_details, order_data, payment_details and product_data. Then I separated them into two kinds of source types as text files and csv files.

The two main sources are listed below:

SQL Database

One text file – order data

Also, the below mentioned CSV files were imported to the SQL source database.

Courier data

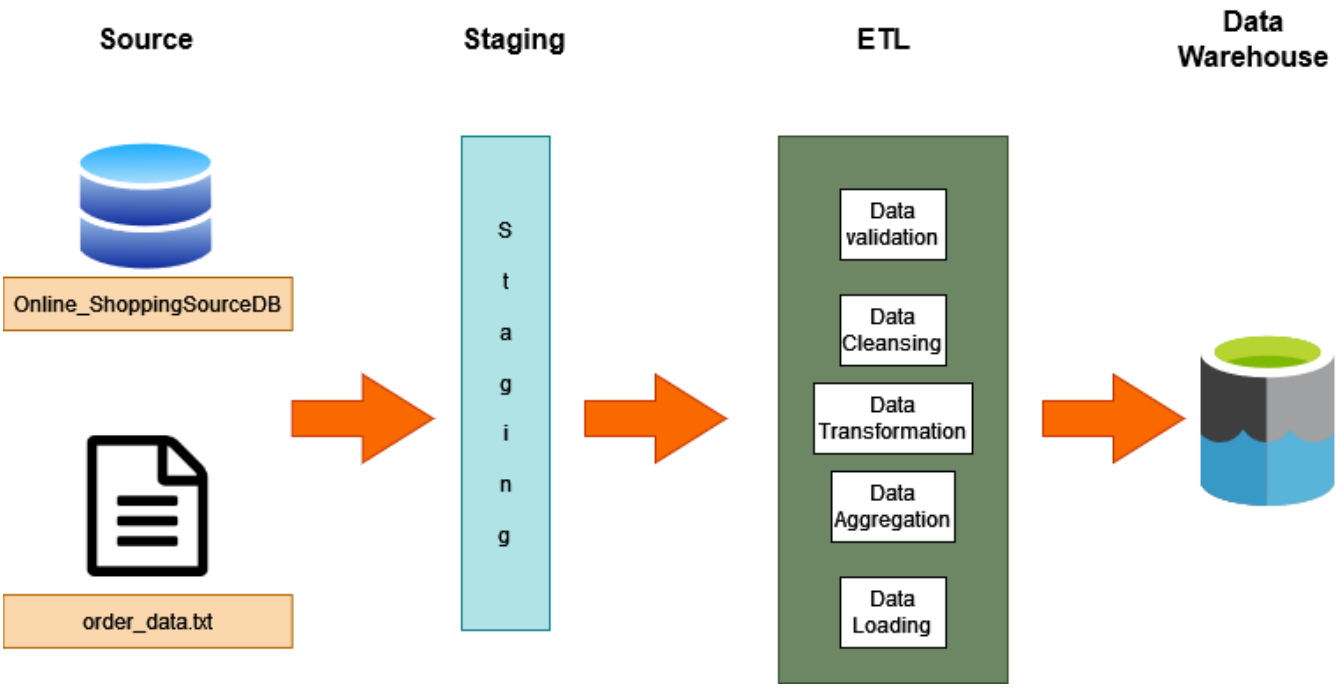
Customer details

Payment details

Product data

In each table I included a primary key. And in Courier data table I added a foreign key reference for the order data table.

3) Solution Architecture

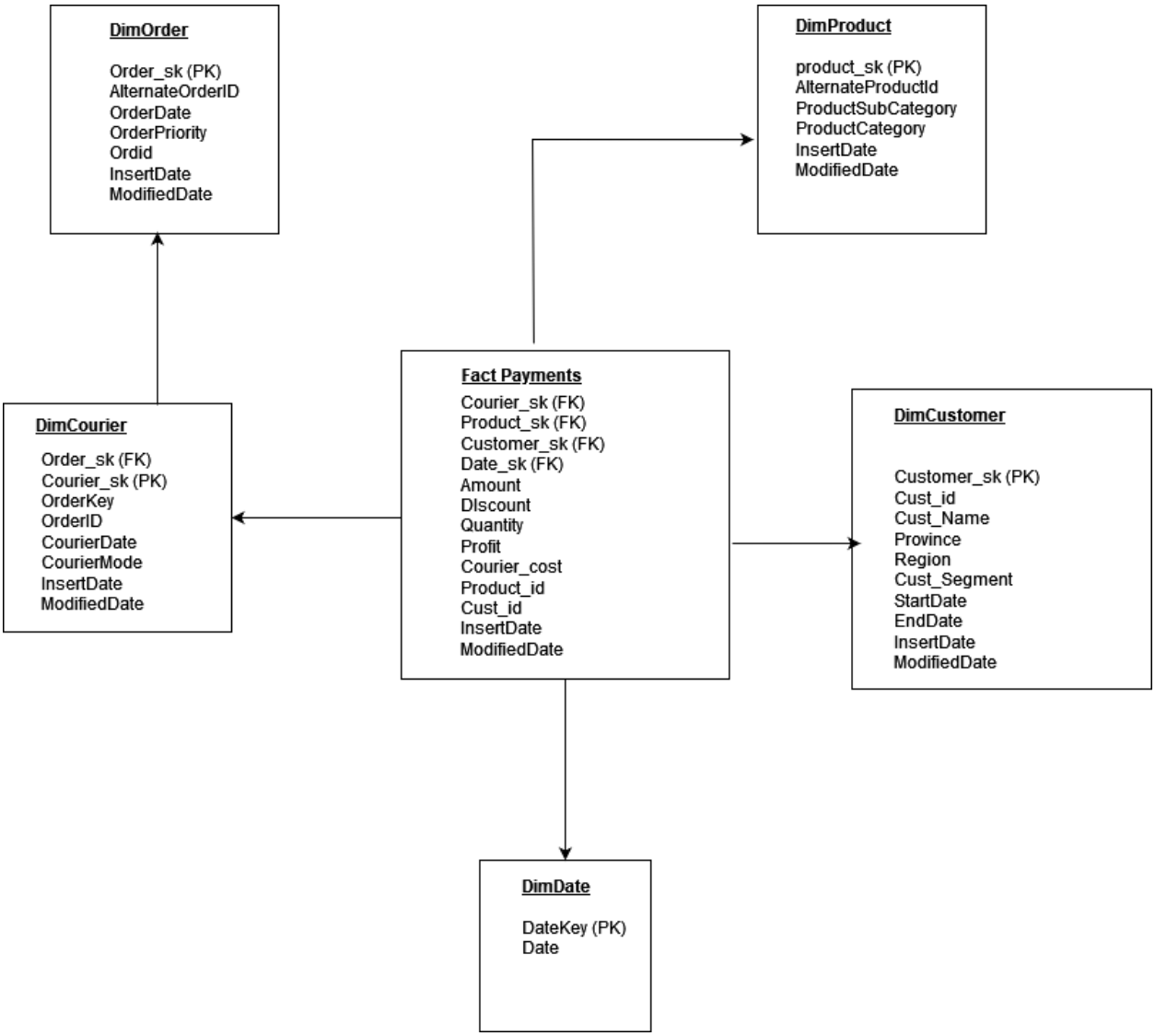


First step is creating the source data set. After the staging layer the below mentioned staging tables are created:

1. Courier data staging
2. Customer details staging
3. Order data staging
4. Payment details staging
5. Product data staging

Next staged tables are profiled. As the next step data is transformed and loaded. After completing the described stages, data is tested and validated and the Datawarehouse is created. After the warehouse is created BI results such as OLAP analysis, Reports, Data visualization, Data mining can be obtained as results after further modifications.

4) Data Warehouse Design and Development



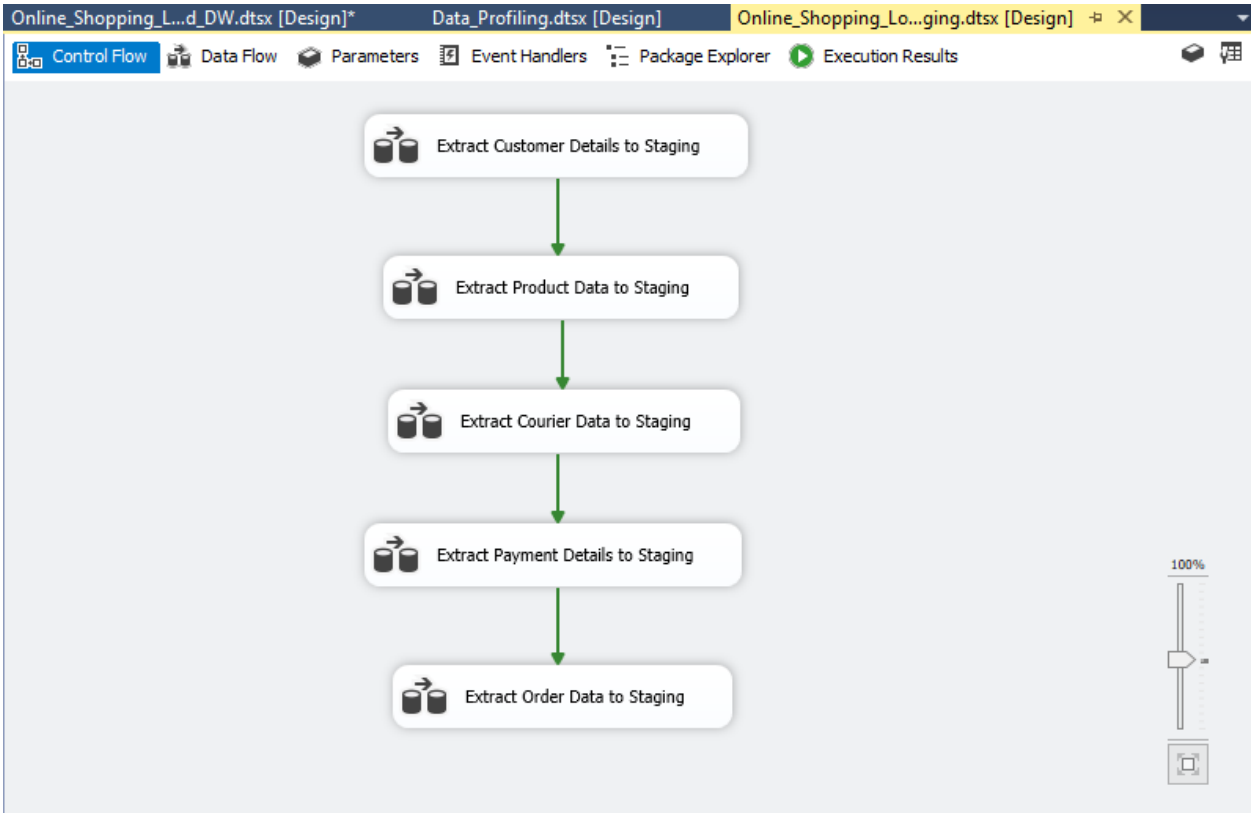
Snowflake schema is used to design the Datawarehouse design. There is one fact table as payments and 4 dimensions.

Assumptions.

Customer Details were considered as a slowly changing dimension.

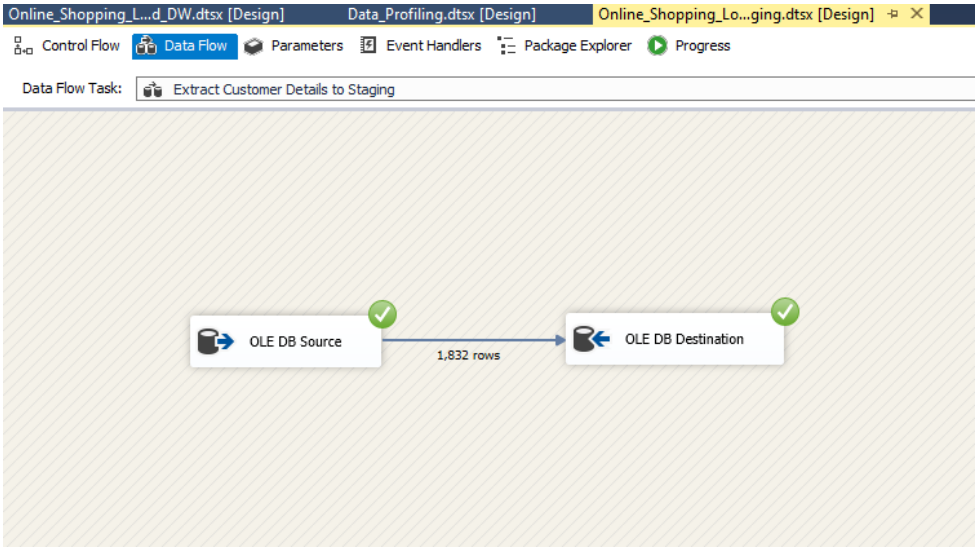
5) ETL Development

The first step of the SSIS ETL process is the extraction of data from source systems. For every extraction, data flow task was used, and data was extracted from the sources to the staging table. All the data flow tasks were joined as shown below at the end:



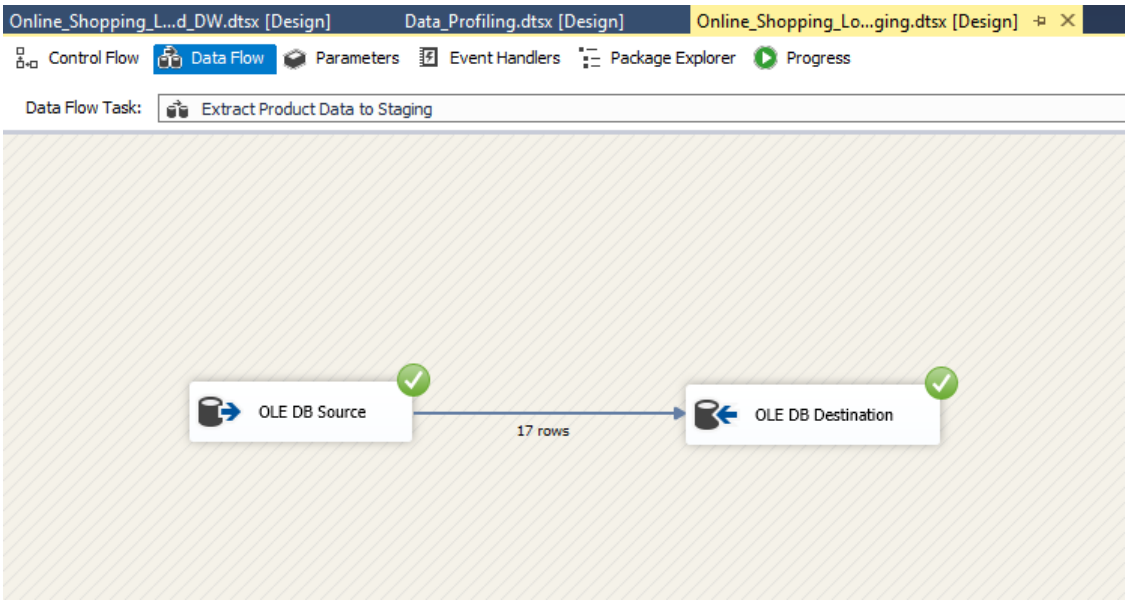
Screenshots of all the data sources that were extracted to staging are attached below:

1) Staging Customer Details



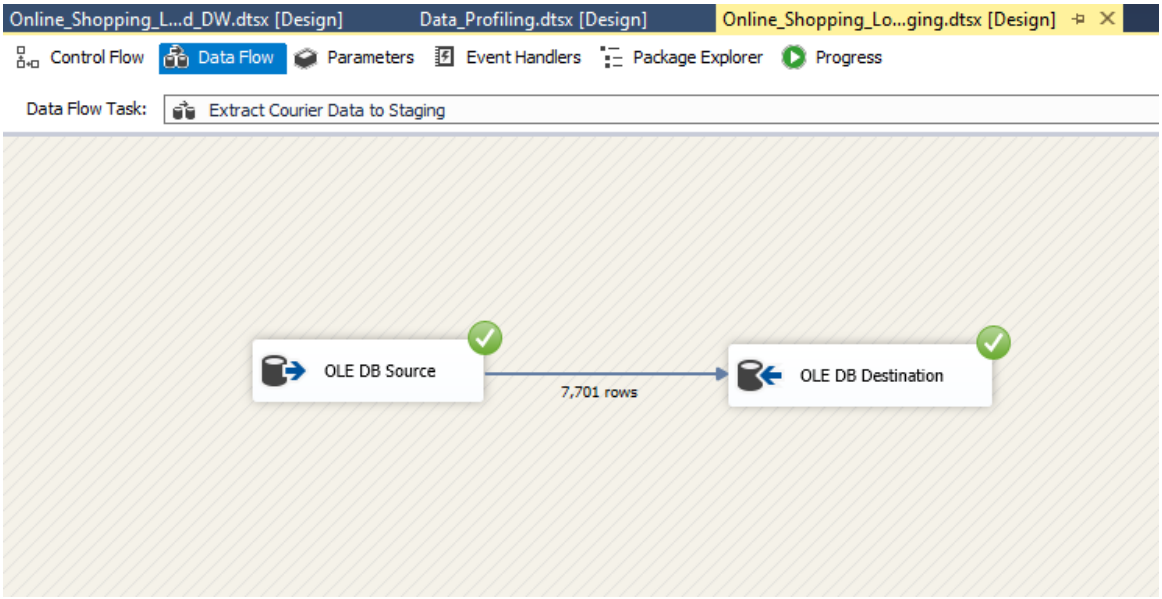
Customer_details – data is extracted from the customer details table in the source database and inserted to the customer details staging table.

2) Staging Product Data



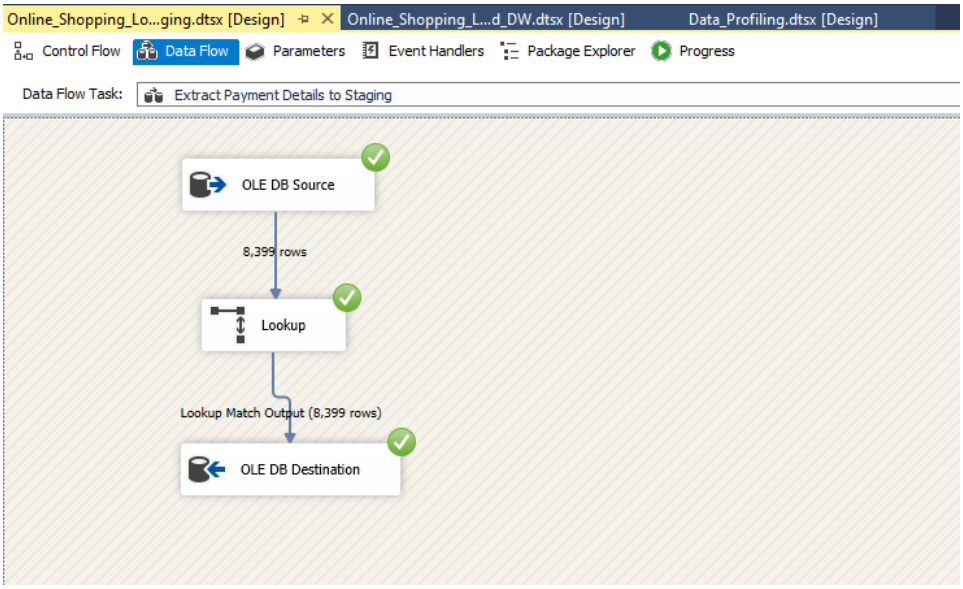
Product_data – data is extracted from the product data table in the source database and inserted to the product data staging table.

3) Staging Courier Data



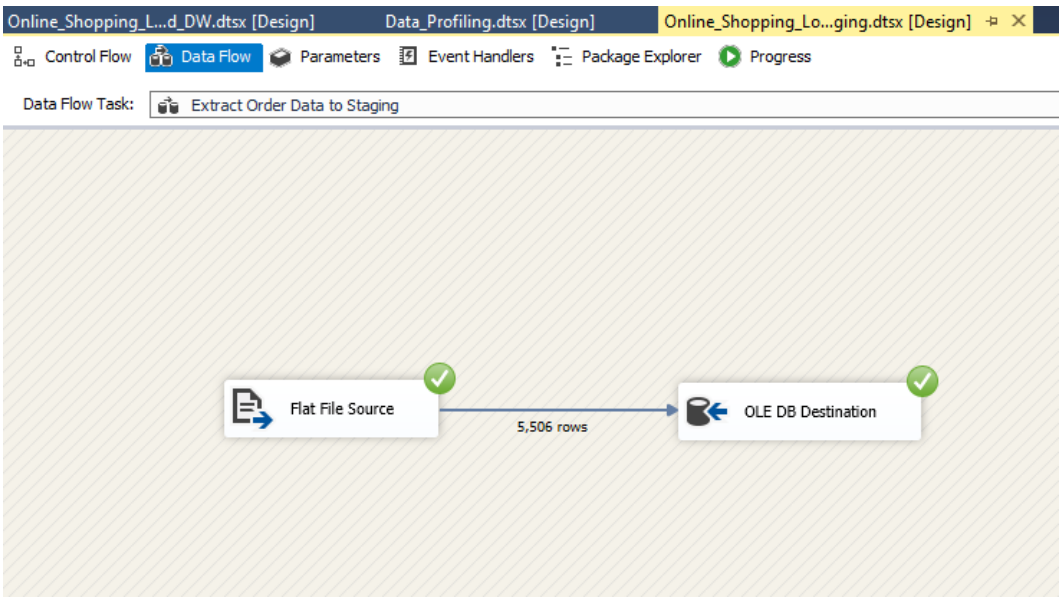
Courier_data – data is extracted from the courier data table in the source database and inserted to the courier data staging table.

4) Staging Payment Details



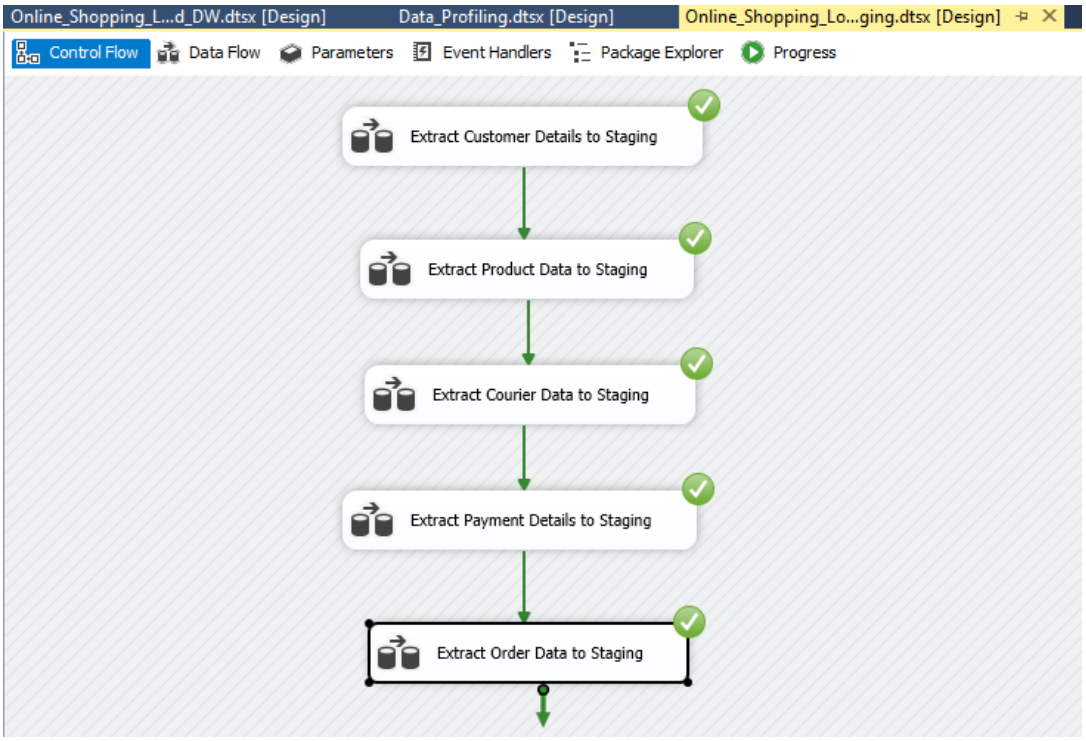
payment_details – data is extracted from the payment details table in the source database and inserted to the payment details staging table.

5) Staging Order Data

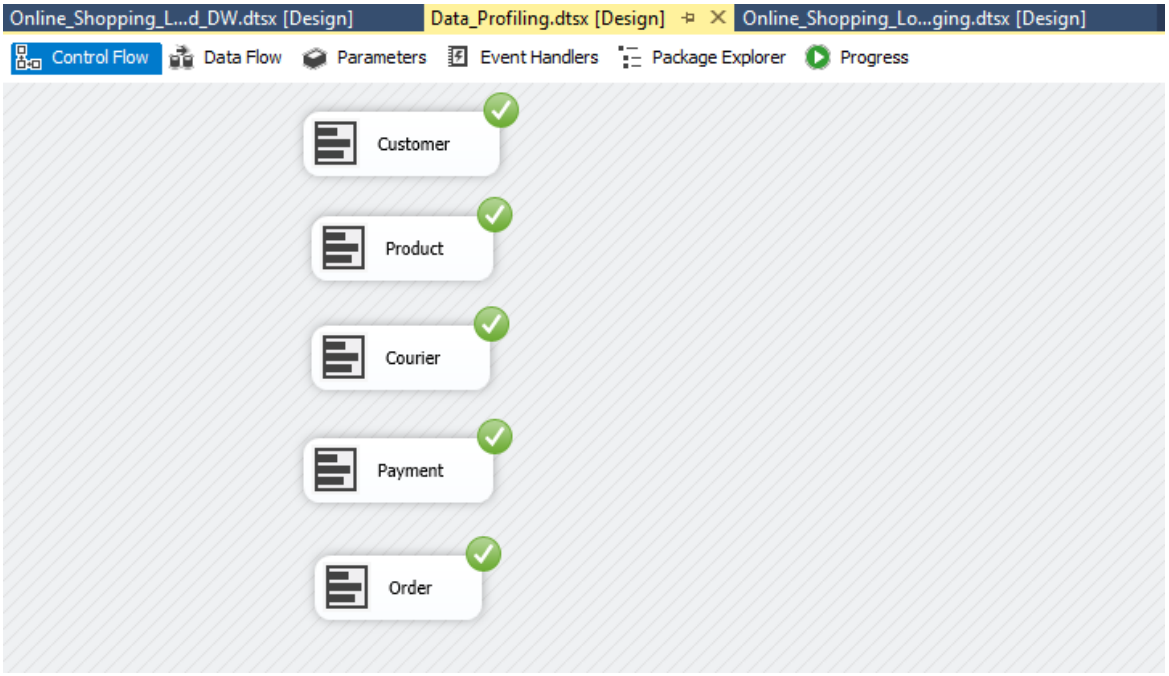


Order_data – data is extracted from the order data table in the source database and inserted to the order data staging table.

After following the above steps and executing:

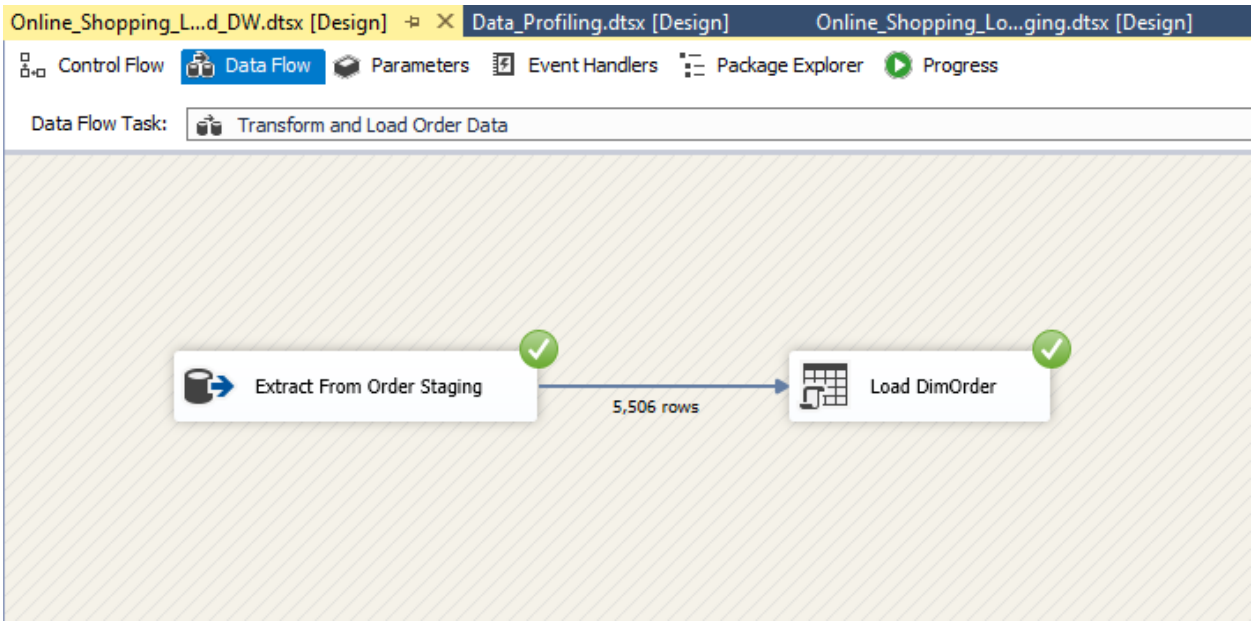


After completing data profiling step:



Next step focuses on how to perform data transformations on the staging database and load them into the data warehouse.

1) Order Data Transformation and Loading

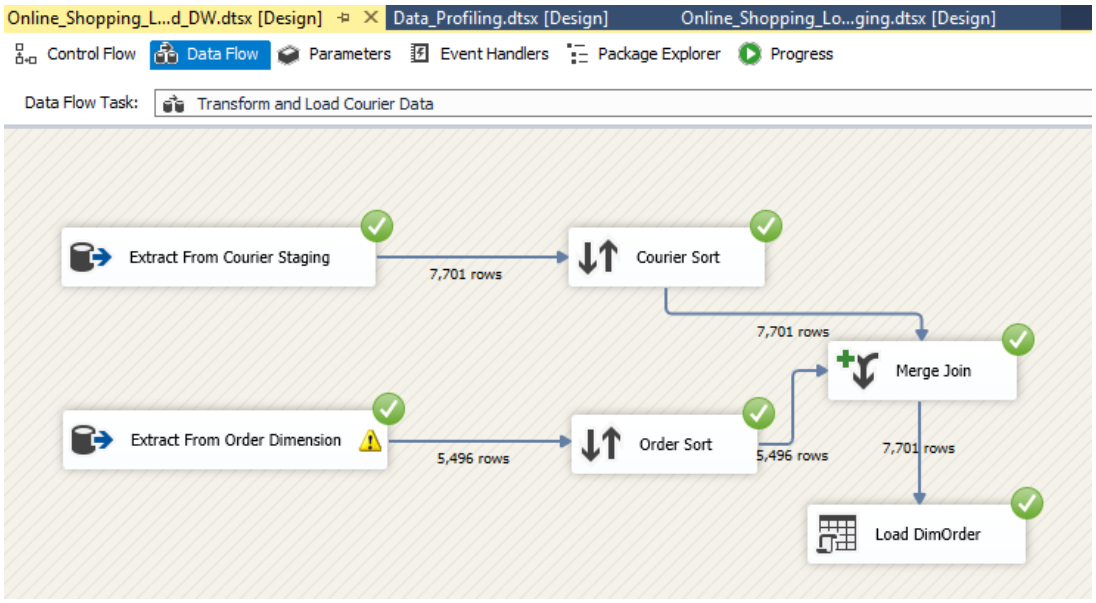


Data was extracted from the staging table and loaded into the order dimension.

The update procedure used to update order data is attached below:

```
CREATE PROCEDURE UpdateDimOrder
@Order_ID int,
@Order_Date datetime,
@Order_Priority nvarchar(50),
@Ord_id nvarchar(50)
AS
BEGIN
if not exists (select Order_sk
from dbo.DimOrder
where AlternateOrderID = @Order_ID)
BEGIN
insert into dbo.DimOrder
(AlternateOrderID,OrderDate,OrderPriority,Ordid,InsertDate, ModifiedDate)
values
(@Order_ID, @Order_Date,@Order_Priority,@Ord_id,GETDATE(), GETDATE())
END;
if exists (select Order_sk
from dbo.DimOrder
where AlternateOrderID = @Order_ID)
BEGIN
update dbo.DimOrder
set OrderDate = @Order_Date,
OrderPriority = @Order_Priority,
OrdId = @Ord_id,
ModifiedDate = GETDATE()
where AlternateOrderID = @Order_ID
END;
END;
```

2) Courier Data Transforming and Loading

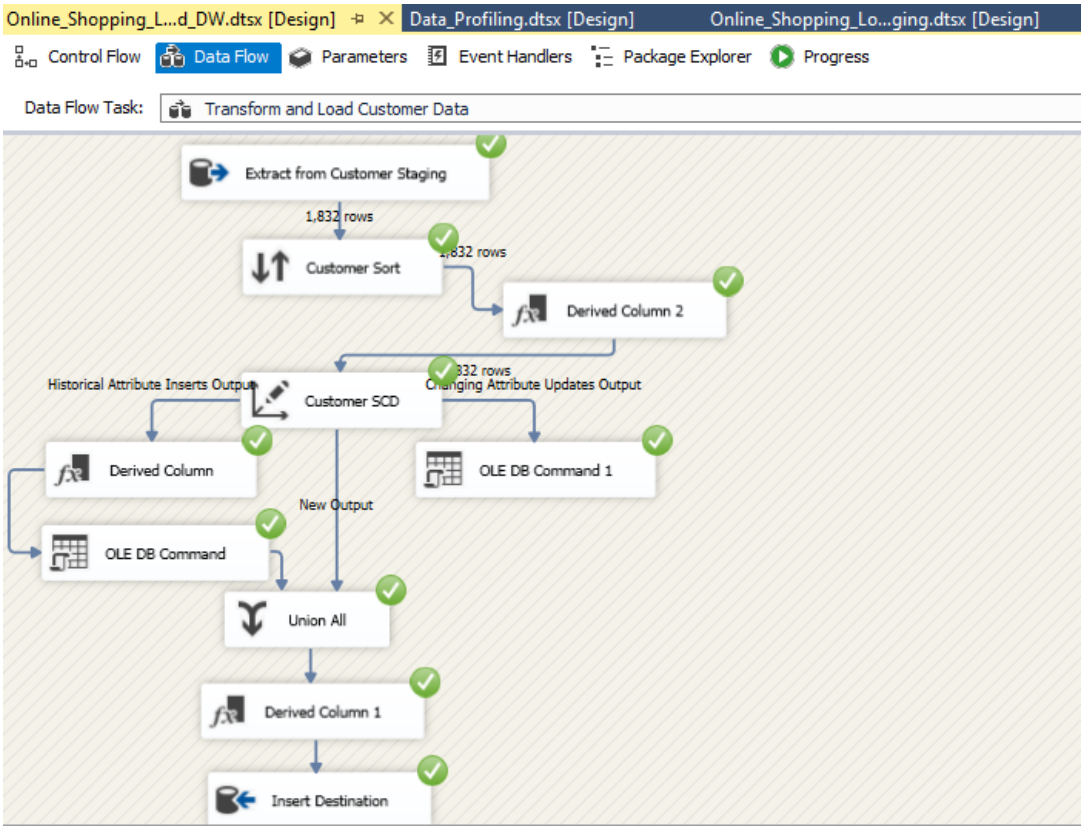


Data was extracted from the staging table and loaded into the courier dimension.

The update procedure used to update courier data is attached below:

```
drop procedure UpdateDimCourier;
create procedure UpdateDimCourier
@Order_ID int,
@Courier_Mode nvarchar(50),
@Courier_Date datetime,
@Courier_id nvarchar(50)
AS
BEGIN
if not exists (select Courier_sk
from dbo.DimCourier
where OrderID = @Order_ID)
BEGIN
insert into dbo.DimCourier
(OrderID, Courierid, CourierDate, CourierMode,InsertDate, ModifiedDate)
values
(@Order_ID, @Courier_id, @Courier_Date, @Courier_Mode,GETDATE(), GETDATE() )
END;
if exists (select Courier_sk
from dbo.DimCourier
where OrderID = @Order_ID)
BEGIN
update dbo.DimCourier
set OrderID = @Order_ID,
Courierid = @Courier_id,
CourierDate = @Courier_Date,
CourierMode = @Courier_Mode,
ModifiedDate = GETDATE()
where OrderID = @Order_ID
END;
END;
```

3) Customer Details Transforming and Loading

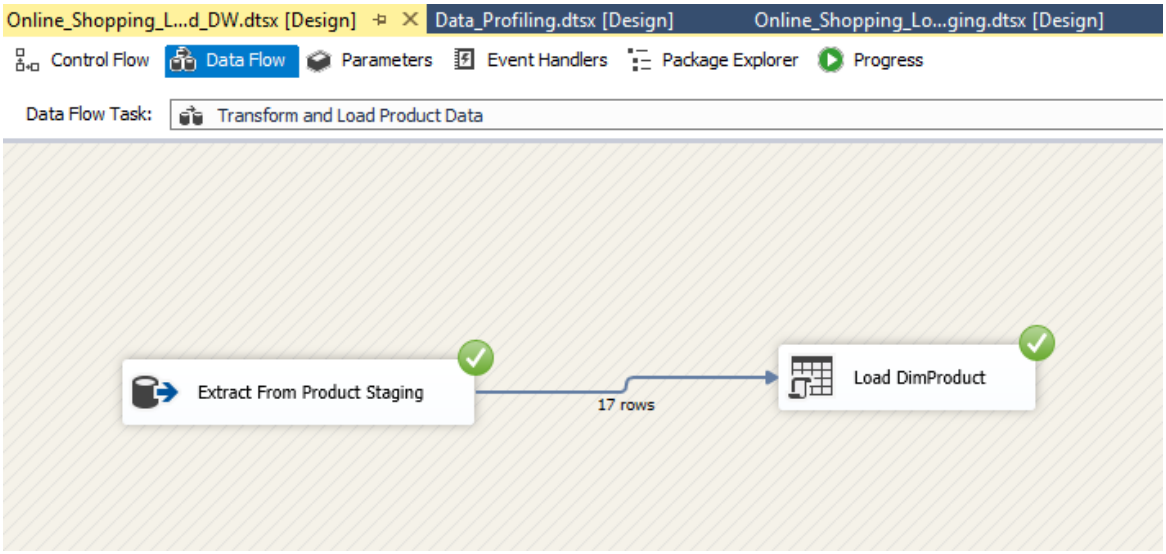


Customer details were considered as slowly changing dimensions.

Province and the customer segments tables are created as changing dimensions.

After extracting data from the Customer staging table, it was sorted according to the customer id and as it was identified as a slowly changing dimension, it was connected as shown above and loaded data to the Customer dimension table.

4) Product Data Transforming and Loading



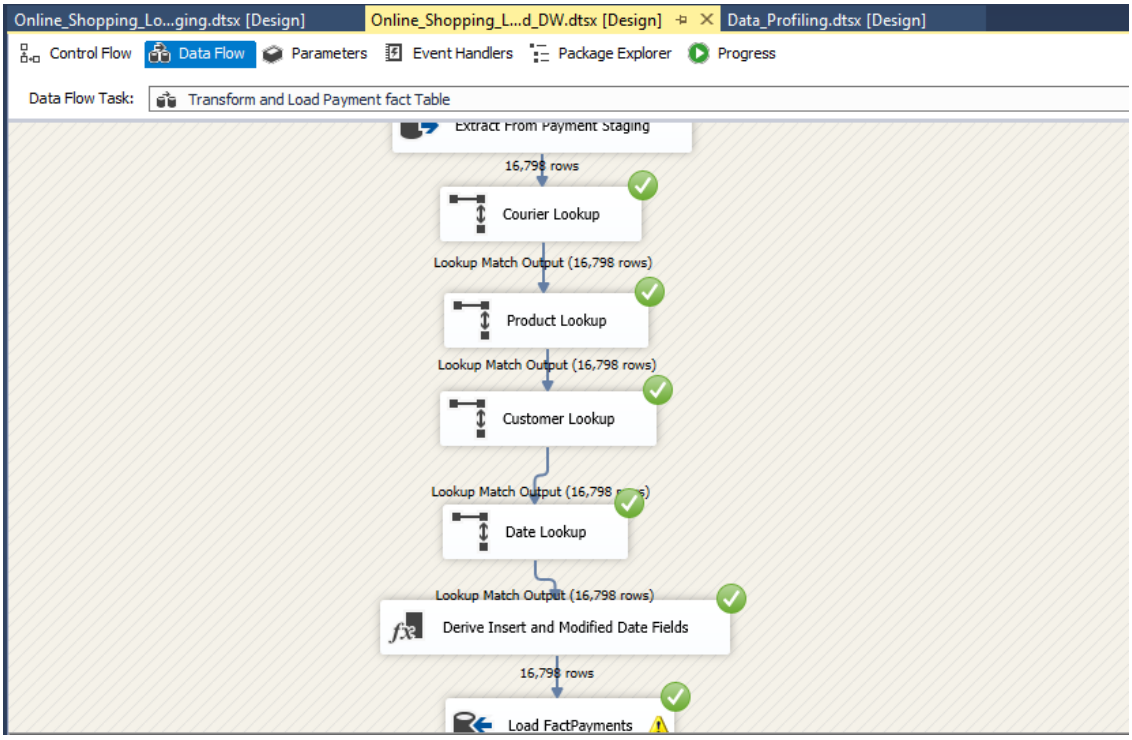
Data was extracted from the staging table and loaded into the product dimension.

The update procedure used to update product data is attached below:

```
CREATE PROCEDURE UpdateDimProduct
@Product_id nvarchar(50),
@Product_Category nvarchar(50),
@Product_Sub_Category nvarchar(50)

AS
BEGIN
if not exists (select product_sk
from dbo.DimProduct
where AlternateProductId = @Product_id)
BEGIN
insert into dbo.DimProduct
(AlternateProductId,ProductCategory,ProductSubCategory,InsertDate, ModifiedDate)
values
(@Product_id, @Product_Category,@Product_Sub_Category,GETDATE(), GETDATE())
END;
if exists (select product_sk
from dbo.DimProduct
where AlternateProductId = @Product_id)
BEGIN
update dbo.DimProduct
set ProductCategory = @Product_Category,
ProductSubCategory = @Product_Sub_Category,
ModifiedDate = GETDATE()
where AlternateProductId = @Product_id
END;
END;
```

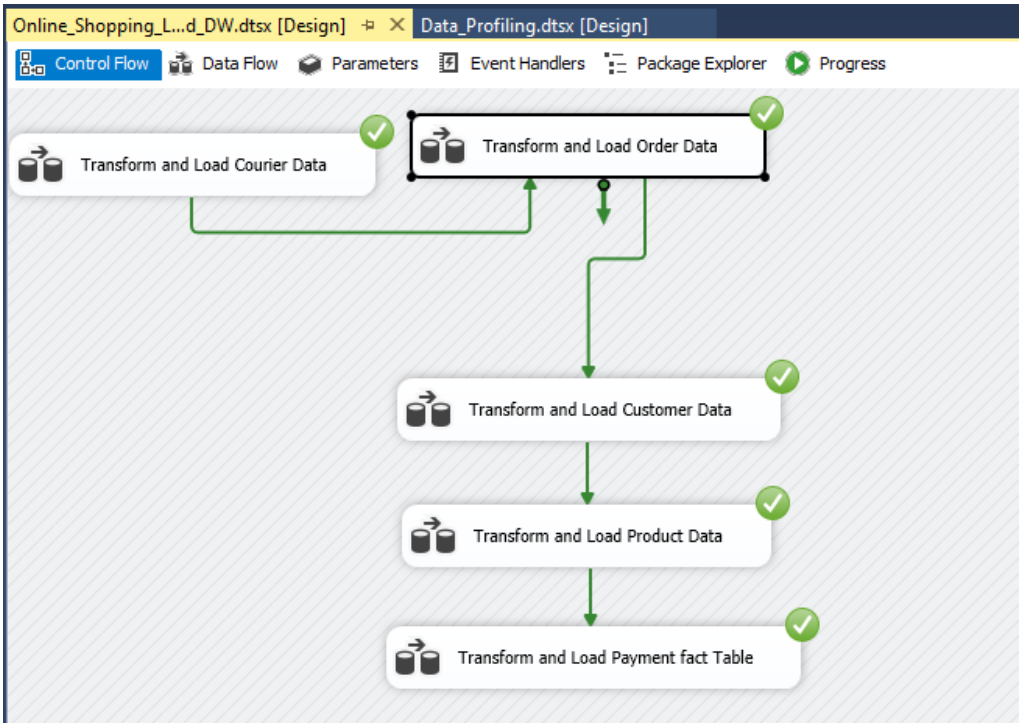
0

5) Transform and Load Payment Fact Table



1. Data was extracted from the Payments staging table.
2. Next steps are to join relevant dimension table to obtain corresponding surrogate keys.
3. The courier dimension table was joined using Courier Lookup component.
4. Product dimension table was joined using Product Lookup component.
5. Customer dimension table was joined using Customer Lookup component.
6. Insert data to the Payment fact table using Derived column component.
7. Extracted tables are loaded to the fact table using OLE DB destination editor.

After executing above tasks, the ETL will be like below.



The query used to create the date dimension is mentioned below.

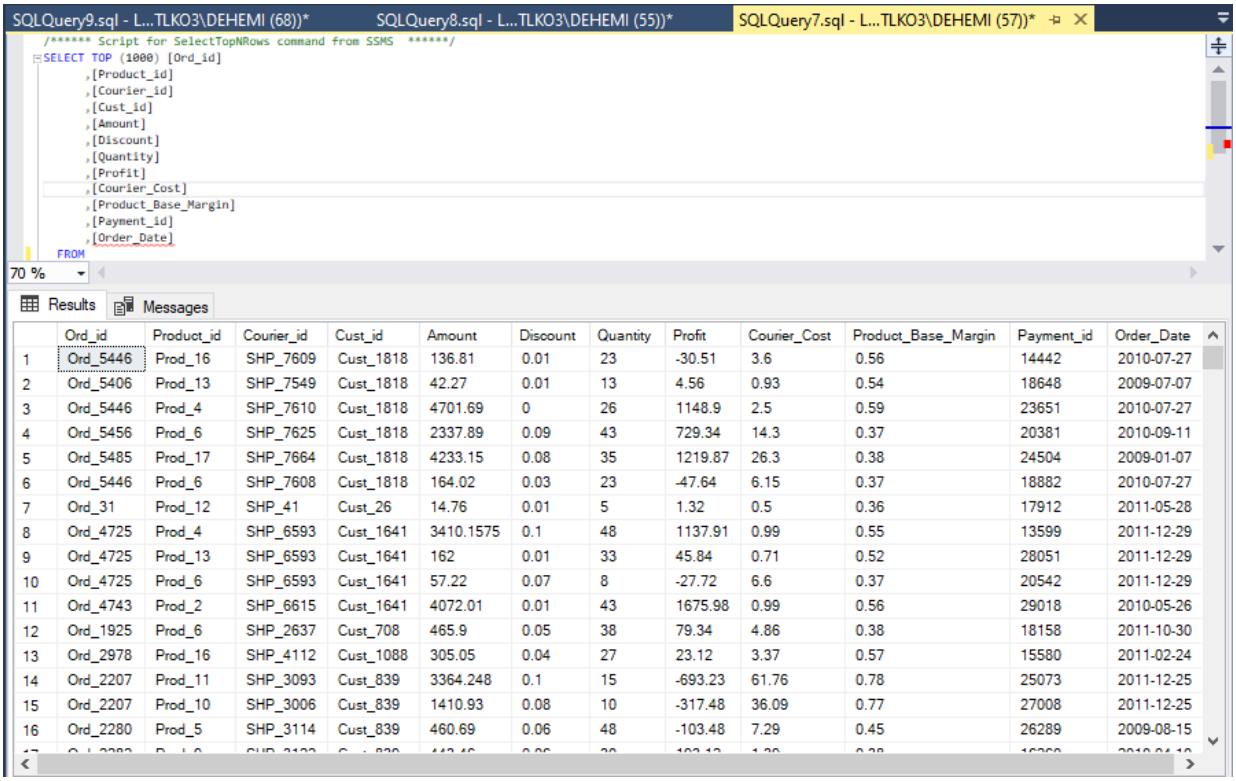
```
BEGIN TRY
        DROP TABLE [dbo].[DimDate]
END TRY

BEGIN CATCH
        /*No Action*/
END CATCH

/***********************************************************************************/
drop table if exists DimDate;
CREATE TABLE    [dbo].[DimDate]
        (       [DateKey] INT primary key,
                [Date] DATETIME,
                [FullDateUK] CHAR(10), -- Date in dd-MM-yyyy format
                [FullDateUSA] CHAR(10),-- Date in MM-dd-yyyy format
                [DayOfMonth] VARCHAR(2), -- Field will hold day number of Month
                [DaySuffix] VARCHAR(4), -- Apply suffix as 1st, 2nd ,3rd etc
                [DayName] VARCHAR(9), -- Contains name of the day, Sunday, Monday
                [DayOfWeekUSA] CHAR(1),-- First Day Sunday=1 and Saturday=7
                [DayOfWeekUK] CHAR(1),-- First Day Monday=1 and Sunday=7
                [DayOfWeekInMonth] VARCHAR(2), --1st Monday or 2nd Monday in Month
                [DayOfWeekInYear] VARCHAR(2),
                [DayOfQuarter] VARCHAR(3),
                [DayOfYear] VARCHAR(3),
                [WeekOfMonth] VARCHAR(1),-- Week Number of Month
                [WeekOfQuarter] VARCHAR(2), --Week Number of the Quarter
                [WeekOfYear] VARCHAR(2),--Week Number of the Year
                [Month] VARCHAR(2), --Number of the Month 1 to 12
                [MonthName] VARCHAR(9),--January, February etc
                [MonthOfQuarter] VARCHAR(2),-- Month Number belongs to Quarter
                [Quarter] CHAR(1),
                [QuarterName] VARCHAR(9),--First,Second..
                [Year] CHAR(4),-- Year value of Date stored in Row
                [YearName] CHAR(7), --CY 2012,CY 2013
                [MonthYear] CHAR(10), --Jan-2013,Feb-2013
                [MMYYYY] CHAR(6),
                [FirstDayOfMonth] DATE,
                [LastDayOfMonth] DATE,
                [FirstDayOfQuarter] DATE,
                [LastDayOfQuarter] DATE,
                [FirstDayOfYear] DATE,
                [LastDayOfYear] DATE,
                [IsHolidaySL] BIT,-- Flag 1=National Holiday, 0-No National Holiday
                [IsWeekday] BIT,-- 0=Week End ,1=Week Day
                [HolidaySL] VARCHAR(50),--Name of Holiday in US
                [isCurrentDay] int, -- Current day=1 else = 0
                [isDataAvailable] int, -- data available for the day = 1, no data available for the day =
0
                [isLatestDataAvailable] int
        )
GO


/*********************************************************************************************/
--Specify Start Date and End date here
--Value of Start Date Must be Less than Your End Date

DECLARE @StartDate DATETIME = '01/01/1990' --Starting value of Date Range
DECLARE @EndDate DATETIME = '01/01/2099' --End Value of Date Range

--Temporary Variables To Hold the Values During Processing of Each Date of Year
DECLARE
        @DayOfWeekInMonth INT,
        @DayOfWeekInYear INT,
        @DayOfQuarter INT,
        @WeekOfMonth INT,
```

```
        @CurrentYear INT,
        @CurrentMonth INT,
        @CurrentQuarter INT

/*Table Data type to store the day of week count for the month and year*/
DECLARE @DayOfWeek TABLE (DOW INT, MonthCount INT, QuarterCount INT, YearCount INT)

INSERT INTO @DayOfWeek VALUES (1, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (2, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (3, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (4, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (5, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (6, 0, 0, 0)
INSERT INTO @DayOfWeek VALUES (7, 0, 0, 0)

--Extract and assign various parts of Values from Current Date to Variable

DECLARE @CurrentDate AS DATETIME = @StartDate
SET @CurrentMonth = DATEPART(MM, @CurrentDate)
SET @CurrentYear = DATEPART(YY, @CurrentDate)
SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)

/********************************************************************************************/
--Proceed only if Start Date(Current date ) is less than End date you specified above

WHILE @CurrentDate < @EndDate
BEGIN

/*Begin day of week logic*/

         /*Check for Change in Month of the Current date if Month changed then
          Change variable value*/
        IF @CurrentMonth != DATEPART(MM, @CurrentDate)
        BEGIN
                UPDATE @DayOfWeek
                SET MonthCount = 0
                SET @CurrentMonth = DATEPART(MM, @CurrentDate)
        END

        /* Check for Change in Quarter of the Current date if Quarter changed then change
         Variable value*/

        IF @CurrentQuarter != DATEPART(QQ, @CurrentDate)
        BEGIN
                UPDATE @DayOfWeek
                SET QuarterCount = 0
                SET @CurrentQuarter = DATEPART(QQ, @CurrentDate)
        END

        /* Check for Change in Year of the Current date if Year changed then change
         Variable value*/


        IF @CurrentYear != DATEPART(YY, @CurrentDate)
        BEGIN
                UPDATE @DayOfWeek
                SET YearCount = 0
                SET @CurrentYear = DATEPART(YY, @CurrentDate)
        END

        -- Set values in table data type created above from variables

        UPDATE @DayOfWeek
        SET
                MonthCount = MonthCount + 1,
                QuarterCount = QuarterCount + 1,
                YearCount = YearCount + 1
        WHERE DOW = DATEPART(DW, @CurrentDate)
```

```
        SELECT
                @DayOfWeekInMonth = MonthCount,
                @DayOfQuarter = QuarterCount,
                @DayOfWeekInYear = YearCount
        FROM @DayOfWeek
        WHERE DOW = DATEPART(DW, @CurrentDate)

/*End day of week logic*/


/* Populate Your Dimension Table with values*/

        INSERT INTO [dbo].[DimDate]
        SELECT

                CONVERT (char(8),@CurrentDate,112) as DateKey,
                @CurrentDate AS Date,
                CONVERT (char(10),@CurrentDate,103) as FullDateUK,
                CONVERT (char(10),@CurrentDate,101) as FullDateUSA,
                DATEPART(DD, @CurrentDate) AS DayOfMonth,
                --Apply Suffix values like 1st, 2nd 3rd etc..
                CASE
                        WHEN DATEPART(DD,@CurrentDate) IN (11,12,13)
                        THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
                        WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 1
                        THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'st'
                        WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 2
                        THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'nd'
                        WHEN RIGHT(DATEPART(DD,@CurrentDate),1) = 3
                        THEN CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'rd'
                        ELSE CAST(DATEPART(DD,@CurrentDate) AS VARCHAR) + 'th'
                        END AS DaySuffix,

                DATENAME(DW, @CurrentDate) AS DayName,
                DATEPART(DW, @CurrentDate) AS DayOfWeekUSA,


-- check for day of week as Per US and change it as per UK format
                CASE DATEPART(DW, @CurrentDate)
                        WHEN 1 THEN 7
                        WHEN 2 THEN 1
                        WHEN 3 THEN 2
                        WHEN 4 THEN 3
                        WHEN 5 THEN 4
                        WHEN 6 THEN 5
                        WHEN 7 THEN 6
                        END
                        AS DayOfWeekUK,

                @DayOfWeekInMonth AS DayOfWeekInMonth,
                @DayOfWeekInYear AS DayOfWeekInYear,
                @DayOfQuarter AS DayOfQuarter,
                DATEPART(DY, @CurrentDate) AS DayOfYear,
                DATEPART(WW, @CurrentDate) + 1 - DATEPART(WW, CONVERT(VARCHAR,
                DATEPART(MM, @CurrentDate)) + '/1/' + CONVERT(VARCHAR,
                DATEPART(YY, @CurrentDate))) AS WeekOfMonth,
                (DATEDIFF(DD, DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0),
                @CurrentDate) / 7) + 1 AS WeekOfQuarter,
                DATEPART(WW, @CurrentDate) AS WeekOfYear,
                DATEPART(MM, @CurrentDate) AS Month,
                DATENAME(MM, @CurrentDate) AS MonthName,
                CASE
                        WHEN DATEPART(MM, @CurrentDate) IN (1, 4, 7, 10) THEN 1
                        WHEN DATEPART(MM, @CurrentDate) IN (2, 5, 8, 11) THEN 2
                        WHEN DATEPART(MM, @CurrentDate) IN (3, 6, 9, 12) THEN 3
                        END AS MonthOfQuarter,
                DATEPART(QQ, @CurrentDate) AS Quarter,
                CASE DATEPART(QQ, @CurrentDate)
```

```
                        WHEN 1 THEN 'First'
                        WHEN 2 THEN 'Second'
                        WHEN 3 THEN 'Third'
                        WHEN 4 THEN 'Fourth'
                        END AS QuarterName,
                DATEPART(YEAR, @CurrentDate) AS Year,
                'CY ' + CONVERT(VARCHAR, DATEPART(YEAR, @CurrentDate)) AS YearName,
                LEFT(DATENAME(MM, @CurrentDate), 3) + '-' + CONVERT(VARCHAR,
                DATEPART(YY, @CurrentDate)) AS MonthYear,
                RIGHT('0' + CONVERT(VARCHAR, DATEPART(MM, @CurrentDate)),2) +
                CONVERT(VARCHAR, DATEPART(YY, @CurrentDate)) AS MMYYYY,
                CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD,
                @CurrentDate) - 1), @CurrentDate))) AS FirstDayOfMonth,
                CONVERT(DATETIME, CONVERT(DATE, DATEADD(DD, - (DATEPART(DD,
                (DATEADD(MM, 1, @CurrentDate)))), DATEADD(MM, 1,
                @CurrentDate)))) AS LastDayOfMonth,
                DATEADD(QQ, DATEDIFF(QQ, 0, @CurrentDate), 0) AS FirstDayOfQuarter,
                DATEADD(QQ, DATEDIFF(QQ, -1, @CurrentDate), -1) AS LastDayOfQuarter,
                CONVERT(DATETIME, '01/01/' + CONVERT(VARCHAR, DATEPART(YY,
                @CurrentDate))) AS FirstDayOfYear,
                CONVERT(DATETIME, '12/31/' + CONVERT(VARCHAR, DATEPART(YY,
                @CurrentDate))) AS LastDayOfYear,
                NULL AS IsHolidaySL,
                CASE DATEPART(DW, @CurrentDate)
                        WHEN 1 THEN 0
                        WHEN 2 THEN 1
                        WHEN 3 THEN 1
                        WHEN 4 THEN 1
                        WHEN 5 THEN 1
                        WHEN 6 THEN 1
                        WHEN 7 THEN 0
                        END AS IsWeekday,
                NULL AS HolidaySL, (case when @CurrentDate = convert(date, sysdatetime()) then 1 else 0
end), 0, 0

        SET @CurrentDate = DATEADD(DD, 1, @CurrentDate)
END

/*****************************************************************************************/

/*****************************************************************************************/

SELECT * FROM [dbo].[DimDate]
```

Fact Table



	Ord_id	Product_id	Courier_id	Cust_id	Amount	Discount	Quantity	Profit	Courier_Cost	Product_Base_Margin	Payment_id	Order_Date
1	Ord_5446	Prod_16	SHP_7609	Cust_1818	136.81	0.01	23	-30.51	3.6	0.56	14442	2010-07-27
2	Ord_5406	Prod_13	SHP_7549	Cust_1818	42.27	0.01	13	4.56	0.93	0.54	18648	2009-07-07
3	Ord_5446	Prod_4	SHP_7610	Cust_1818	4701.69	0	26	1148.9	2.5	0.59	23651	2010-07-27
4	Ord_5456	Prod_6	SHP_7625	Cust_1818	2337.89	0.09	43	729.34	14.3	0.37	20381	2010-09-11
5	Ord_5485	Prod_17	SHP_7664	Cust_1818	4233.15	0.08	35	1219.87	26.3	0.38	24504	2009-01-07
6	Ord_5446	Prod_6	SHP_7608	Cust_1818	164.02	0.03	23	-47.64	6.15	0.37	18882	2010-07-27
7	Ord_31	Prod_12	SHP_41	Cust_26	14.76	0.01	5	1.32	0.5	0.36	17912	2011-05-28
8	Ord_4725	Prod_4	SHP_6593	Cust_1641	3410.1575	0.1	48	1137.91	0.99	0.55	13599	2011-12-29
9	Ord_4725	Prod_13	SHP_6593	Cust_1641	162	0.01	33	45.84	0.71	0.52	28051	2011-12-29
10	Ord_4725	Prod_6	SHP_6593	Cust_1641	57.22	0.07	8	-27.72	6.6	0.37	20542	2011-12-29
11	Ord_4743	Prod_2	SHP_6615	Cust_1641	4072.01	0.01	43	1675.98	0.99	0.56	29018	2010-05-26
12	Ord_1925	Prod_6	SHP_2637	Cust_708	465.9	0.05	38	79.34	4.86	0.38	18158	2011-10-30
13	Ord_2978	Prod_16	SHP_4112	Cust_1088	305.05	0.04	27	23.12	3.37	0.57	15580	2011-02-24
14	Ord_2207	Prod_11	SHP_3093	Cust_839	3364.248	0.1	15	-693.23	61.76	0.78	25073	2011-12-25
15	Ord_2207	Prod_10	SHP_3006	Cust_839	1410.93	0.08	10	-317.48	36.09	0.77	27008	2011-12-25
16	Ord_2280	Prod_5	SHP_3114	Cust_839	460.69	0.06	48	-103.48	7.29	0.45	26289	2009-08-15

6) ETL development – Accumulating fact tables

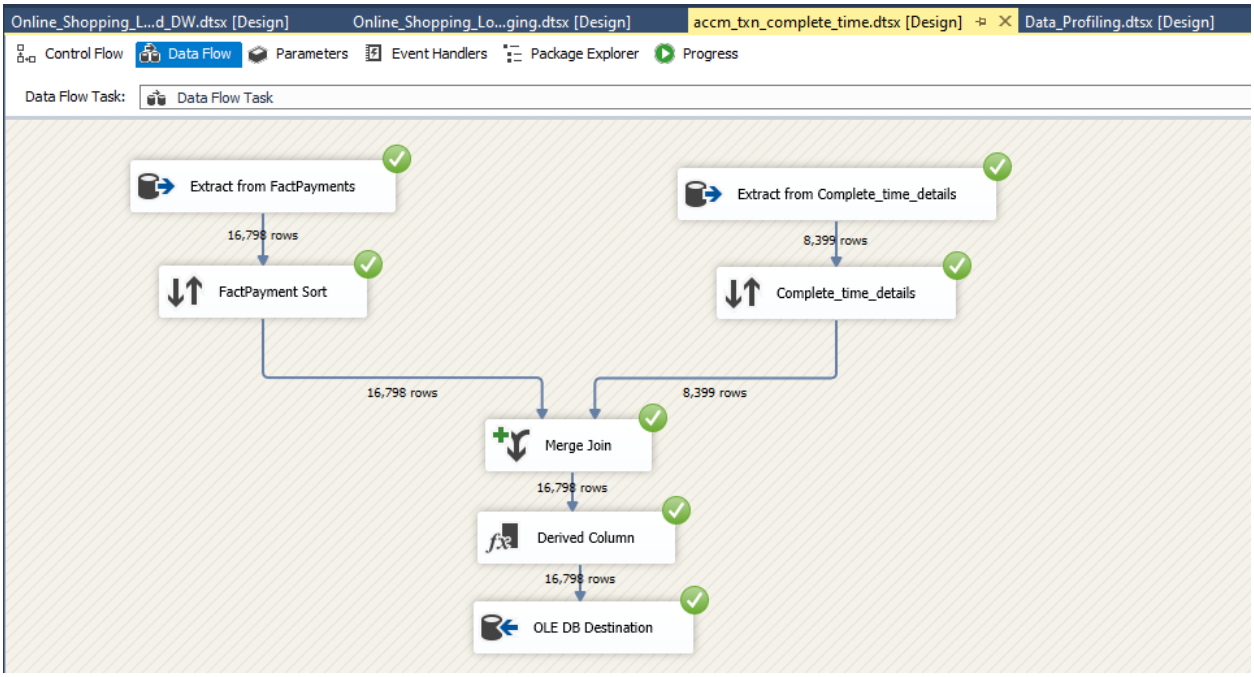
Fact table was extended by below three columns.

accm_txn_create_time

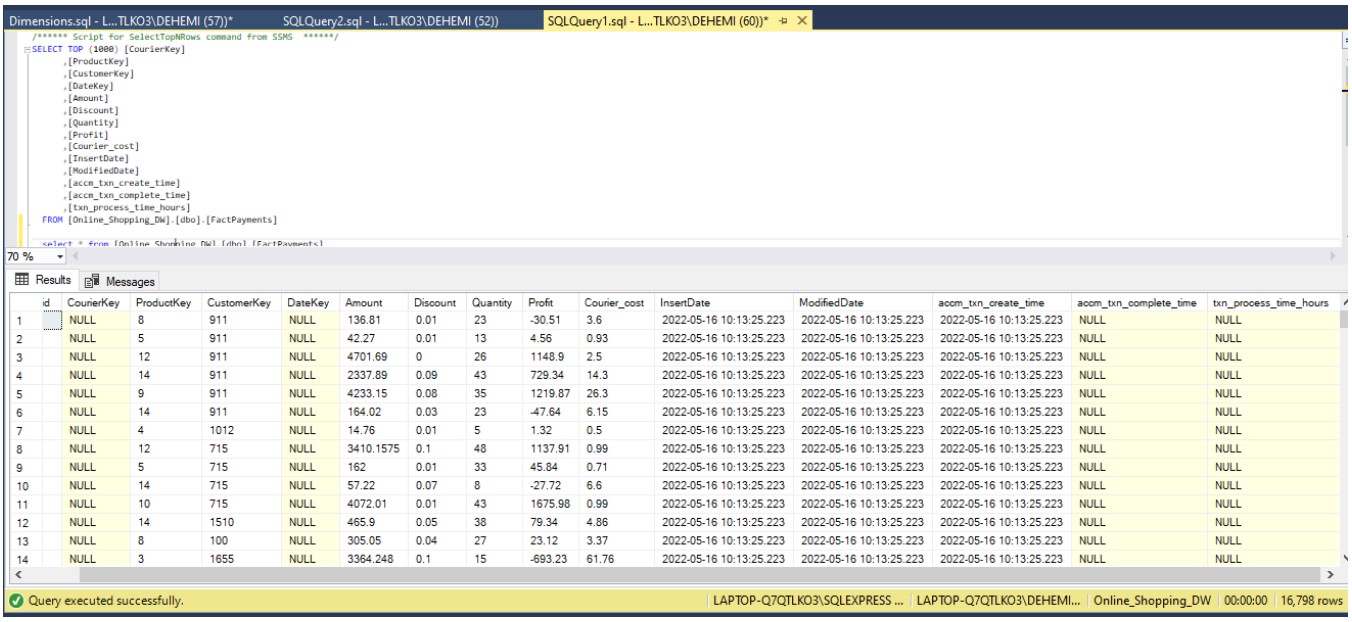
accm_txn_complete_time

txn_process_time_hours

following is the ETL ssis package which reads data from the created source file and update the corresponding accm_txn_complete_time in DW fact table



Following is the created fact table.


Dimensions.sql - L...TLKO3\DEHEMI (57))*
SQLQuery2.sql - L...TLKO3\DEHEMI (52))
SQLQuery1.sql - L...TLKO3\DEHEMI (60))*
/****** Script for SelectTopNRows command from SSMS ******/
SELECT TOP (1000) [CourierKey]
,[ProductKey]
,[CustomerKey]
,[DateKey]
,[Amount]
,[Discount]
,[Quantity]
,[Profit]
,[Courier_cost]
,[InsertDate]
,[ModifiedDate]
,[accm_txn_create_time]
,[accm_txn_complete_time]
,[txn_process_time_hours]
FROM [Online_Shopping_DW].[dbo].[FactPayments]
70 %
Results
Messages
id CourierKey ProductKey CustomerKey DateKey Amount Discount Quantity Profit Courier_cost InsertDate ModifiedDate accm_txn_create_time accm_txn_complete_time txn_process_time_hours
1 NULL 8 911 NULL 136.81 0.01 23 -30.51 3.6 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
2 NULL 5 911 NULL 42.27 0.01 13 4.56 0.93 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
3 NULL 12 911 NULL 4701.69 0 26 1148.9 2.5 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
4 NULL 14 911 NULL 2337.89 0.09 43 729.34 14.3 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
5 NULL 9 911 NULL 4233.15 0.08 35 1219.87 26.3 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
6 NULL 14 911 NULL 164.02 0.03 23 -47.64 6.15 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
7 NULL 4 1012 NULL 14.76 0.01 5 1.32 0.5 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
8 NULL 12 715 NULL 3410.1575 0.1 48 1137.91 0.99 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
9 NULL 5 715 NULL 162 0.01 33 45.84 0.71 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
10 NULL 14 715 NULL 57.22 0.07 8 -27.72 6.6 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
11 NULL 10 715 NULL 4072.01 0.01 43 1675.98 0.99 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
12 NULL 14 1510 NULL 465.9 0.05 38 79.34 4.86 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
13 NULL 8 100 NULL 305.05 0.04 27 23.12 3.37 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
14 NULL 3 1655 NULL 3364.248 0.1 15 -693.23 61.76 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 2022-05-16 10:13:25.223 NULL NULL
Query executed successfully.
LAPTOP-Q7QTLKO3\SQLEXPRESS ...
LAPTOP-Q7QTLKO3\DEHEMI...
Online_Shopping_DW
00:00:00
16,798 rows