Data Warehousing & Business Intelligence
Y3 S2

Assignment 1

Submitted to
Sri Lanka Institute of Information Technology
By
Fernando WTH – IT22313652
Weekend Batch

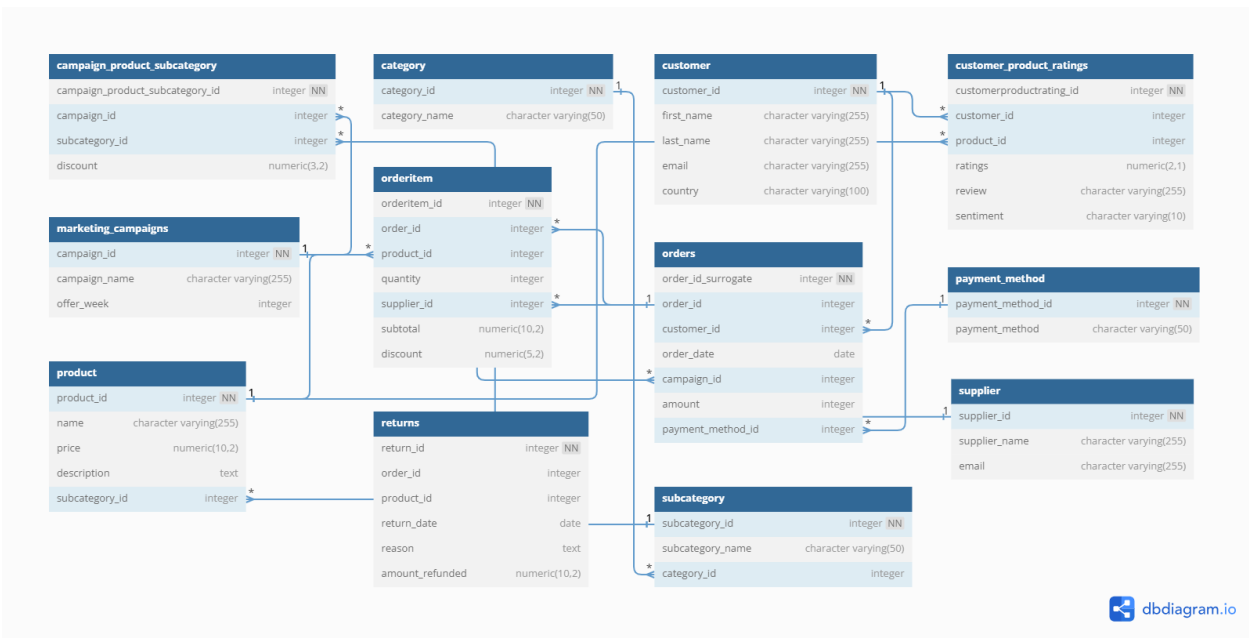
Step 1: Data Set Selection

The selected dataset is a publicly available dataset that simulates a real-world e-commerce online retail platform. It can be used to create a data warehouse solution for order lifecycle tracking and advanced customer behavior analytics. Each table represents either a business entity or a transaction which aligns with the OLTP characteristics.

Following the previously set guidelines for this assignment, the dataset consisted of around one year's worth of data and records and attributes. The contents of this dataset was only csv files but using them I created 3 different data sources (CSV, text file and database). As the data was sufficient to create a data warehouse, I was also able to perform ETL functions with this dataset. I am also able to identify the various hierarchies, dimensions and aggregates within this dataset and as this is a data collection I am also able to generate reports.

Do to this I selected this Ecommerce Dataset as my chosen dataset for this assignment.
Link: https://www.kaggle.com/datasets/sharangkulkarni/oltp-ecommerce-data?select=eCommerce_schema.sql

ER Diagram Schema



Step 02: Preparation of Data Sources

There were 12 CSV files available in the dataset. They were category.csv, customer.csv, customer_product_ratings.csv, marketing_campaigns.csv, orderitem.csv, orders.csv, payment_method.csv, product.csv, returns.csv, subcategory.csv, supplier.csv and campaign_product_subcategory.csv.

Using 9 of those CSV files I created a database an used that as one of my sources. For the remaining 3 files, category.csv, customer.csv and payment_method.csv, I decided to convert the customer file to a text file and use the remaining two as CSV files bringing my data source count to 3.

Data Source	Data Source Type	Description	Final Type
category.csv	csv	This file contains the basic information about the various product categories.	CSV
customer.csv	csv	This file contains the basic information about the various customers.	Text File(.txt)
customer_product_ratings.csv	csv	This file contains information about the customer product ratings.	Ecommerce_OLTP datablase
marketing_campaigns.csv	csv	This file contains the basic information about the marketing campaigns.	
orderitem.csv	csv	This file contains the basic information about the various ordered items.	
orders.csv	csv	This file contains information about the customer orders.	
payment_method.csv	csv	This file contains the payment methods.	CSV
product.csv	csv	This file contains the basic information about the various products.	Ecommerce_OLTP datablase
returns.csv	csv	This file contains the information regarding order returns.	
subcategory.csv	csv	This file contains the basic information about the subcategories of products.	

supplier.csv	csv	This file contains the basic information about the suppliers.	Ecommerce_OLTP datablase
campaign_product_subcategory .csv	csv	This file contains the information about the ampaign product subcategories. .	

Shown below are the files I imported to the database

A1 customerproductrating_id

	A	B	C	D	E	F
1	customerp	customer_	product_id	ratings	review	sentiment
2	5760	83	206	2.5	Avoid this p	bad
3	5761	356	76	4.5	Great qual	good
4	5762	92	344	4	Very satisf	good
5	5763	270	212	4.5	Outstandir	good
6	5764	270	439	3.5	Fantastic p	good
7	5765	412	121	1.5	Product di	bad
8	5766	28	109	4.5	Great qual	good
9	5767	323	294	4.5	Couldn't be	good
10	5768	173	27	4	Top-notch	good
11	5769	184	247	3.5	Fantastic p	good
12	5770	258	410	2	Overpricec	bad
13	5771	378	213	3	Waste of m	bad
14	5772	126	327	2.5	Terrible pr	bad
15	5773	422	412	4	Excellent p	good
16	5774	469	57	1	False adve	bad
17	5775	489	375	2	Cheaply m	bad
18	5776	322	135	2	Terrible pr	bad
19	5777	112	483	4.5	Fantastic p	good
20	5778	115	20	4.5	Impressed	good
21	5779	168	450	2	Overpricec	bad
22	5780	291	94	4	Delivered c	good
23	5781	450	258	3.5	Excellent p	good
24	5782	424	66	1	Waste of m	bad
25	5783	195	496	1.5	Avoid this p	bad
26	5784	449	180	2	Waste of m	bad

Figure1 customer_product_ratings.csv

A1 campaign_id

	A	B	C	D	E
1	campaign_	campaign_	offer_week		
2	1	NewYearS	1		
3	2	Valentines	6		
4	3	EasterSale	16		
5	4	Independe	27		
6	5	Halloween	44		
7	6	Thanksgivi	48		
8	7	Christmas	52		
9	8	LaborDayS	36		
10	9	MemorialD	21		
11	10	BlackFrida	48		
12	11	CyberMon	49		
13	12	Mother'sD	19		
14	13	Father'sDa	24		
15	14	BackToSch	34		
16	15	SummerSc	25		
17	16	Groundhog	5		

Figure 2 marketing_campaigns.csv

	A	B	C	D	E	F	G	H
1	orderitem_	order_id	product_id	quantity	supplier_id	subtotal	discount	
2	1	1	109	6	39	9.6	0	
3	2	1	281	8	25	1238.64	0	
4	3	1	358	8	27	5503.52	0	
5	4	1	74	10	43	530.6	0	
6	5	1	481	6	12	1069.2	0	
7	6	2	67	5	2	1773.5	0	
8	7	2	99	3	40	1808.76	0	
9	8	3	194	9	11	2814.12	0	
10	9	3	109	4	29	6.4	0	
11	10	3	474	9	27	1142.73	0	
12	11	3	76	5	4	4000.6	0	
13	12	4	487	8	15	2297.28	0	
14	13	4	270	1	8	162.68	0	
15	14	4	405	3	24	938.97	0	
16	15	4	82	7	42	4895.38	0	
17	16	4	358	5	5	3439.7	0	
18	17	4	354	7	15	3717.63	0	
19	18	4	47	5	45	2852.35	0	
20	19	4	293	2	13	1417.4	0	
21	20	4	288	4	17	3941.36	0	
22	21	4	252	8	28	5707.92	0	
23	22	5	324	8	43	5560.88	0.19	
24	23	5	428	1	30	867.33	0.23	
25	24	5	173	2	38	324.66	0.09	

Figure 3 orderitem.csv

	A	B	C	D	E	F	G	H	I
1	order_id_s	order_id	customer_	order_date	campaign_	amount	payment_method_id		
2	1	1	373	########		8352	4		
3	2	2	408	9/5/2021		3582	2		
4	3	3	101	6/2/2019		7964	5		
5	4	4	247	########		29371	1		
6	5	5	361	########	5	15620	5		
7	6	6	295	########		45001	5		
8	7	7	252	9/3/2018	8	10243	2		
9	8	8	488	########		12405	2		
10	9	9	227	########		21059	3		
11	10	10	279	########		8972	4		
12	11	11	484	4/8/2019		14947	3		
13	12	12	314	########	14	3553	4		
14	13	13	333	########	3	13101	3		
15	14	14	463	########		22147	1		
16	15	15	285	########		4589	2		
17	16	16	221	########		3330	2		
18	17	17	370	4/4/2020		4436	5		
19	18	18	240	########		5325	3		
20	19	19	497	########		15636	1		
21	20	20	178	2/6/2017	2	4337	1		
22	21	21	194	########	7	15270	5		
23	22	22	327	4/4/2020		30035	1		
24	23	23	159	########		12029	4		
25	24	24	78	########	15	12514	1		
26	25	25	20	########		12488	3		

Figure 4 orders.csv

product_id	name	price	descriptio	subcategory_id
1	Smartphor	545.54	This is a de	1
2	Smartphor	59.94	This is a de	1
3	Smartphor	216.59	This is a de	1
4	Smartphor	414.26	This is a de	1
5	Smartphor	431.23	This is a de	1
6	Laptops - F	736.51	This is a de	2
7	Laptops - F	673.74	This is a de	2
8	Laptops - F	689.21	This is a de	2
9	Laptops - F	917.56	This is a de	2
10	Laptops - F	505.61	This is a de	2
11	Headphon	235.61	This is a de	3
12	Headphon	869.98	This is a de	3
13	Headphon	528.03	This is a de	3
14	Headphon	993.84	This is a de	3
15	Headphon	766.82	This is a de	3
16	Cameras -	615.52	This is a de	4
17	Cameras -	942.05	This is a de	4
18	Cameras -	795.17	This is a de	4
19	Cameras -	628.59	This is a de	4
20	Cameras -	944.07	This is a de	4
21	Wearables	692.04	This is a de	5
22	Wearables	98.64	This is a de	5
23	Wearables	735.91	This is a de	5
24	Wearables	567.42	This is a de	5
25	Wearables	282.11	This is a de	5

Figure 5 product.csv

return_id	order_id	product_id	return_dat	reason	amount_refunded
1	28970	167	########	Received v	0
2	46978	433	2/8/2017	Color does	152.55
3	41845	60	########	Changed n	0
4	1026	225	8/8/2018	Item not as	0
5	16602	30	########	Received v	0
6	32669	367	########	Changed n	500.16
7	2555	227	########	Product di	0
8	31659	250	########	Size doesn	892.74
9	25315	473	########	Color does	191.92
10	42742	333	########	Defective p	0
11	21124	169	########	Defective p	1179.68
12	44203	22	########	Color does	0
13	27229	482	9/2/2018	Found a be	0
14	48178	356	########	Ordered by	88.78
15	6387	232	########	Changed n	7.04
16	10099	292	########	Found a be	0
17	27348	162	########	Product da	0
18	30144	216	########	Product di	0
19	13540	266	########	Size doesn	0
20	854	456	########	Received v	643.55
21	33985	207	3/3/2017	Defective p	0
22	44688	183	########	Changed n	809.05
23	8850	70	########	Changed n	0
24	30942	263	########	Received v	0
25	14471	355	########	Item not as	0

Figure 6 returns.csv

subcatego	subcatego	category_id
1	Smartphor	1
2	Laptops	1
3	Headphon	1
4	Cameras	1
5	Wearables	1
6	T-Shirts	2
7	Dresses	2
8	Jeans	2
9	Sweaters	2
10	Activewea	2
11	Furniture	3
12	Cookware	3
13	Bedding	3
14	Appliances	3
15	Decor	3
16	Fiction	4
17	Non-Fictio	4
18	Mystery	4
19	Science Fi	4
20	Biography	4
21	Outdoor Cl	5
22	Exercise E	5
23	Camping G	5
24	Sports Sho	5
25	Bicycles	5

Figure 7 subcategory.csv

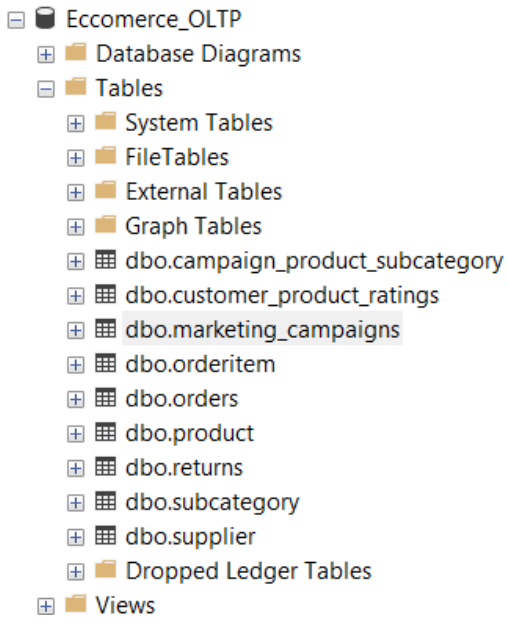
supplier_id	supplier_n	email
1	Alexander,	christopherpearson@example.net
2	Wu Ltd	crystal27@example.org
3	Dixon, Johr	xtapia@example.net
4	Martinez-V	amber91@example.com
5	Hernandez	ipatterson@example.org
6	Ingram-Wh	ortizchristopher@example.com
7	Page-Mcc	garrettdarrell@example.org
8	Best, Werr	dennisgarcia@example.net
9	James Inc	qmontoya@example.org
10	Freeman a	desireeballard@example.com
11	Fitzgerald,	ylee@example.org
12	Wilcox-Rai	andrewhurley@example.net
13	Davis, Und	melaniemejia@example.com
14	Lamb Grou	benjamin10@example.com
15	Reid, Morg	felicia71@example.org
16	Gutierrez-(	trevorbrown@example.net
17	Flores, Fie	wmiller@example.org
18	Gonzales (	jnguyen@example.com
19	House, Pri	nathan81@example.net
20	Crawford,	coopermatthew@example.com
21	Guzman-G	emily68@example.net
22	Bowers-Mi	xcurry@example.org
23	Baker-Jacl	tyoung@example.org
24	Howard, C	noblemonique@example.net
25	Robinson,	uprice@example.org

Figure 8 supplier.csv

campaign_	campaign_	subcatego	discount
1	1	1	0.16
2	1	2	0.11
3	1	3	0.25
4	1	4	0.11
5	1	5	0.07
6	1	6	0.17
7	1	7	0.24
8	1	8	0.07
9	1	9	0.12
10	1	10	0.2
11	1	11	0.17
12	1	12	0.05
13	1	13	0.06
14	1	14	0.22
15	1	15	0.1
16	1	16	0.07
17	1	17	0.18
18	1	18	0.2
19	1	19	0.17
20	1	20	0.15
21	1	21	0.09
22	1	22	0.24
23	1	23	0.19
24	1	24	0.14
25	1	25	0.15

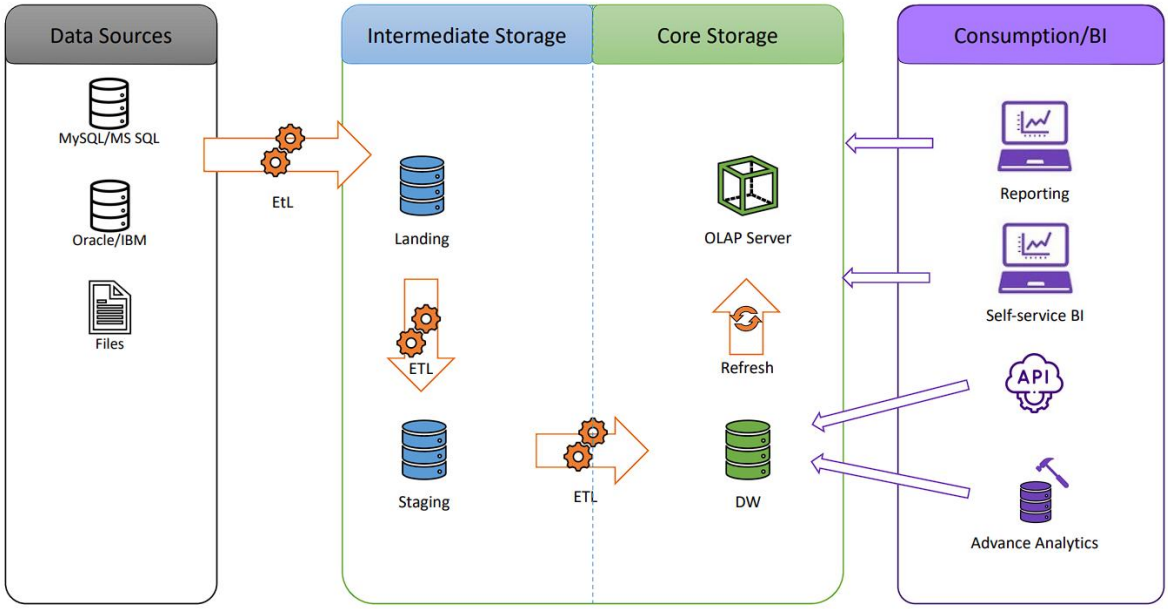
Figure 9 campaign_product_subcategory.csv

I loaded them all into a database as shown below.



I also created a data warehouse name Ecommerce_OLTP_DW where I created all my dimension tables and fact tables.

Step 3: Solution Architecture



1. **Source Systems**: Text, CSV, SQL tables
2. **ETL Layer (SSIS)**: Extract from sources, transform data (cleansing, surrogate keys, lookups), load into DW
3. **Data Warehouse (SQL Server)**: Snowflake schema containing dimensions and fact tables
4. **Reporting Tools**: Power BI

Step 4: Data warehouse design & development

For my Data Set the schema that I chose was the snowflake schema. In my data there are 8-dimension tables and 3 fact tables. The slowly changing dimension table is the Customer table as the country the customer is currently residing in can change from time to time as well as their email, so we have to maintain historical data.

Dimension Tables Created:

- DimCustomer (SCD)
- DimProduct
- DimSupplier
- DimCategory
- DimSubCategory
- DimCampaign
- DimCampaignSubcategory
- DimDate

Each dimension includes insert_date, modified_date, and in the case of DimCustomer, also includes start_date, end_date.

Logical foreign key relationships are maintained through surrogate keys but not enforced physically for ETL performance.

Dimension Table Creation

Before going forward with the other dimension tables I first created a date dimension table.

HP\SECOND.Eccom...W - dbo.DimDate | SQLQuery10.sql -...TP_DW (HP\HP (55))*

Column Name	Data Type	Allow Nulls
DateKey	int	☐
Date	datetime	☑
FullDateUK	char(10)	☑
FullDateUSA	char(10)	☑
DayOfMonth	varchar(2)	☑
DaySuffix	varchar(4)	☑
DayName	varchar(9)	☑
DayOfWeekUSA	char(1)	☑
DayOfWeekUK	char(1)	☑
DayOfWeekInMonth	varchar(2)	☑
DayOfWeekInYear	varchar(2)	☑
DayOfQuarter	varchar(3)	☑
DayOfYear	varchar(3)	☑
WeekOfMonth	varchar(1)	☑
WeekOfQuarter	varchar(2)	☑
WeekOfYear	varchar(2)	☑
Month	varchar(2)	☑
MonthName	varchar(9)	☑
MonthOfQuarter	varchar(2)	☑
Quarter	char(1)	☑
QuarterName	varchar(9)	☑
Year	char(4)	☑
YearName	char(7)	☑
MonthYear	char(10)	☑
MMYYYY	char(6)	☑
FirstDayOfMonth	date	☑
LastDayOfMonth	date	☑

Then following that I created a total of 7 other dimension tables.

Column Name	Data Type	Allow Nulls
campaign_key	int	☐
campaign_id	tinyint	☑
campaign_name	nvarchar(50)	☑
offer_week	tinyint	☑
start_date	date	☑
end_date	date	☑
insert_date	datetime	☑
modified_date	datetime	☑
		☐

DimCampaign

HP\SECOND.Eccom...paignSubcategory | SQLQuery10.sql -...TP_DW (HP\HP (55))*

Column Name	Data Type	Allow Nulls
campaign_subcategory_key	int	☐
campaign_id	tinyint	☑
subcategory_id	tinyint	☑
discount	nvarchar(16)	☑
insert_date	datetime	☑
modified_date	datetime	☑

DimCampaignSubCategory

HP\SECOND.Eccome...dbo.DimCategory

Column Name	Data Type	Allow Nulls
category_key	int	☐
category_id	int	☑
category_name	nvarchar(100)	☑
insert_date	datetime	☑
modified_date	datetime	☑
		☐

DimCategory

HP\SECOND.Eccom...dbo.DimCustomer

Column Name	Data Type	Allow Nulls
customer_key	int	☐
customer_id	int	☑
first_name	nvarchar(200)	☑
last_name	nvarchar(200)	☑
email	nvarchar(200)	☑
country	nvarchar(200)	☑
start_date	datetime	☑
end_date	datetime	☑
insert_date	datetime	☑
modified_date	datetime	☑
		☐

DimCustomer

HP\SECOND.Eccome...- dbo.DimProduct

Column Name	Data Type	Allow Nulls
product_key	int	☐
product_id	int	☑
name	nvarchar(50)	☑
price	float	☑
description	nvarchar(100)	☑
subcategory_id	tinyint	☑
insert_date	datetime	☑
modified_date	datetime	☑
		☐

DimProduct

HP\SECOND.Eccom...o.DimSubCategory

Column Name	Data Type	Allow Nulls
subcategory_key	int	☐
subcategory_id	tinyint	☑
subcategory_name	nvarchar(100)	☑
category_key	int	☑
insert_date	datetime	☑
modified_date	datetime	☑
		☐

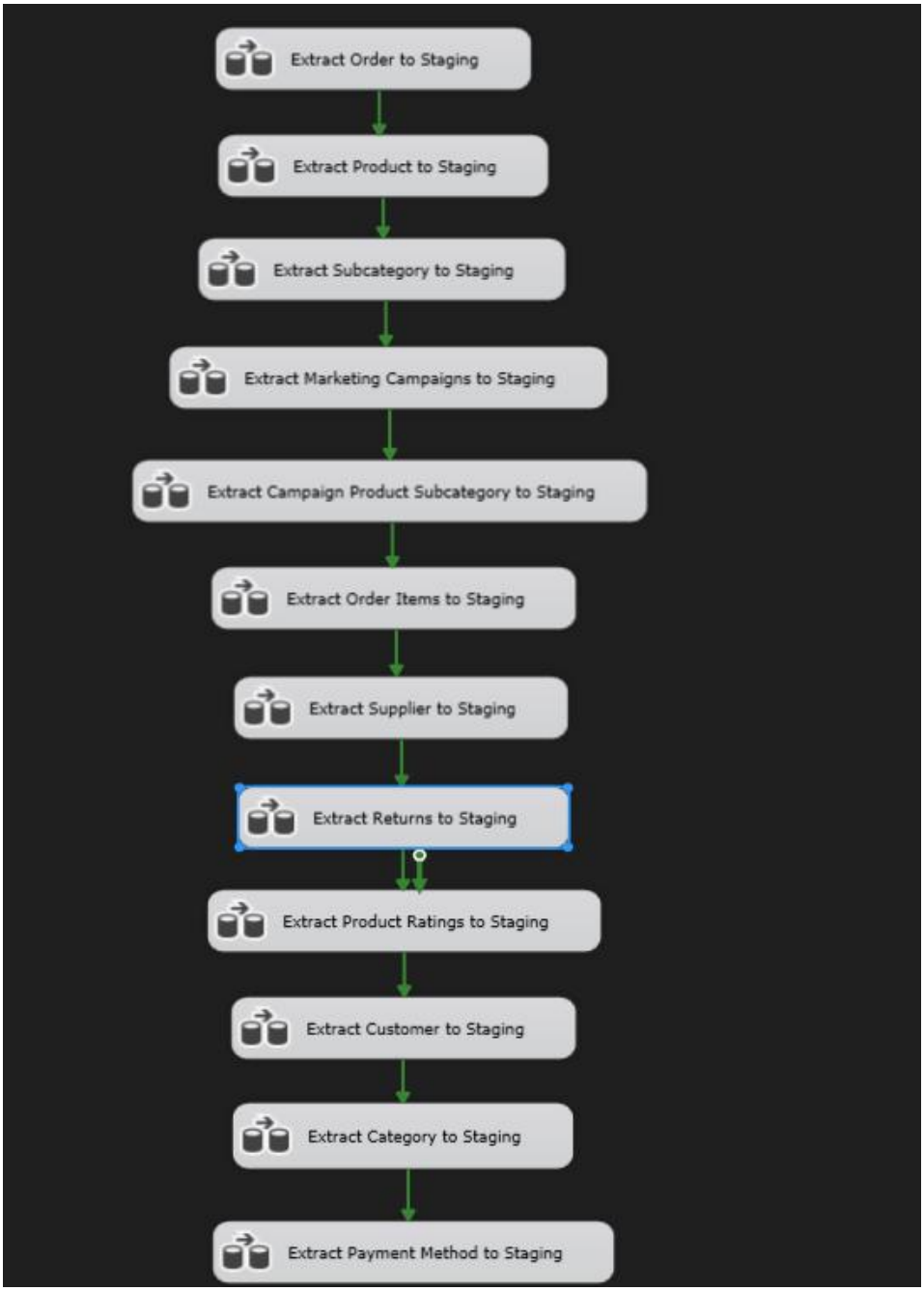
DimSubCategory

HP\SECOND.Eccome...dbo.DimSupplier

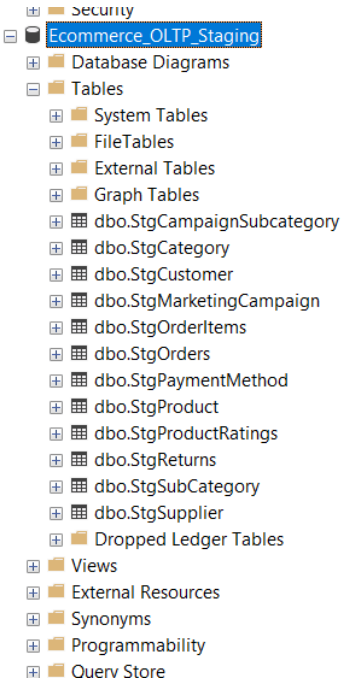
Column Name	Data Type	Allow Nulls
supplier_key	int	☐
supplier_id	tinyint	☑
supplier_name	nvarchar(50)	☑
supplier_email	nvarchar(50)	☑
insert_date	datetime	☑
modified_date	datetime	☑
		☐

DimSupplier

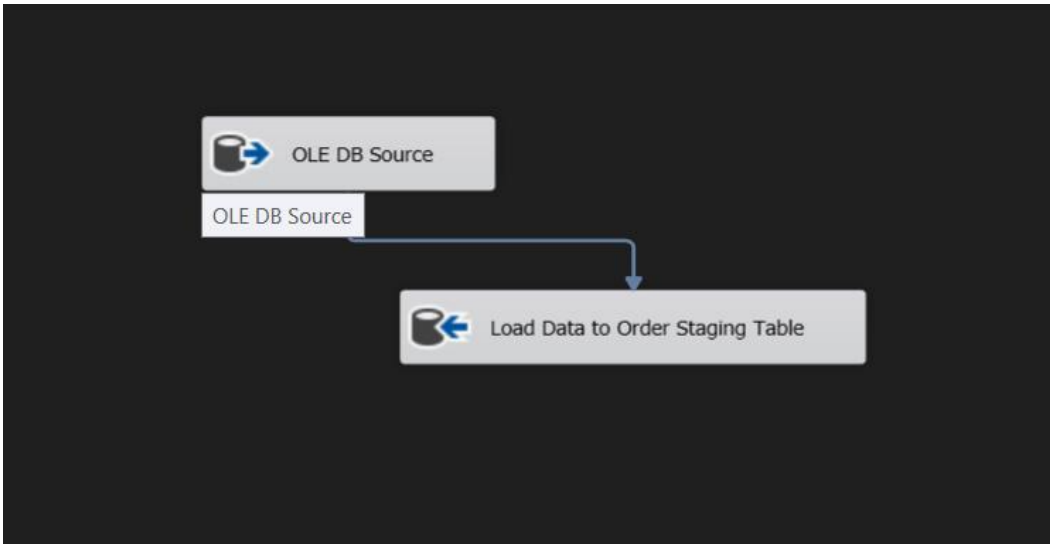
Step 5: ETL Development



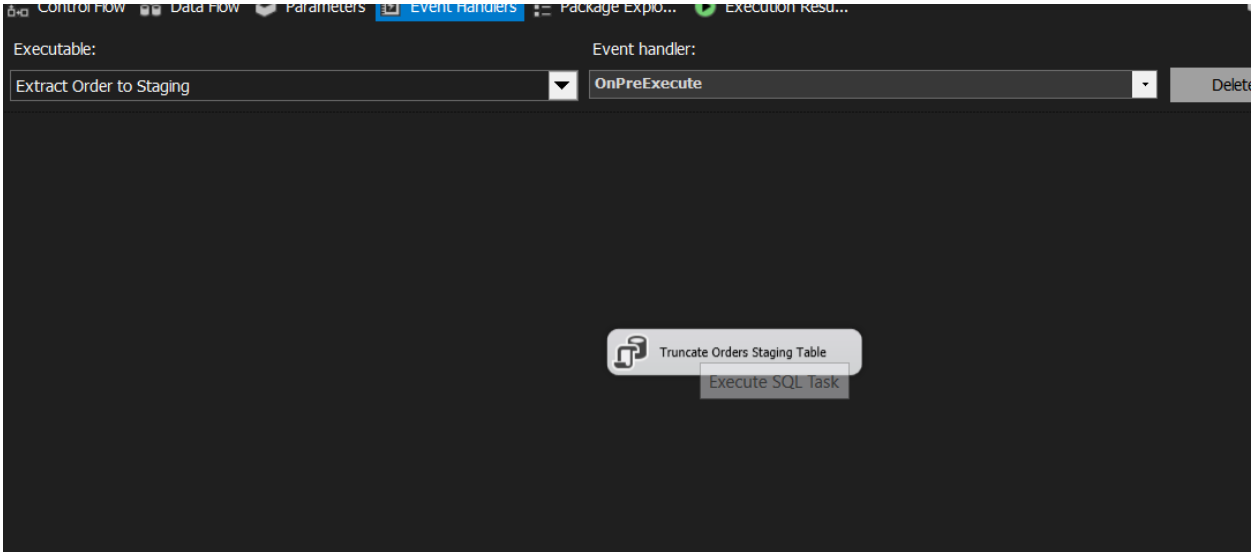
Using the SQL Server Integration Service available in Visual Studio, I extracted all the data from the tables that were in the source database and the 2 separate CSV files and the text file to a separate staging DB called Ecommerce_OLTP_Staging.



Extract Order Data to Staging



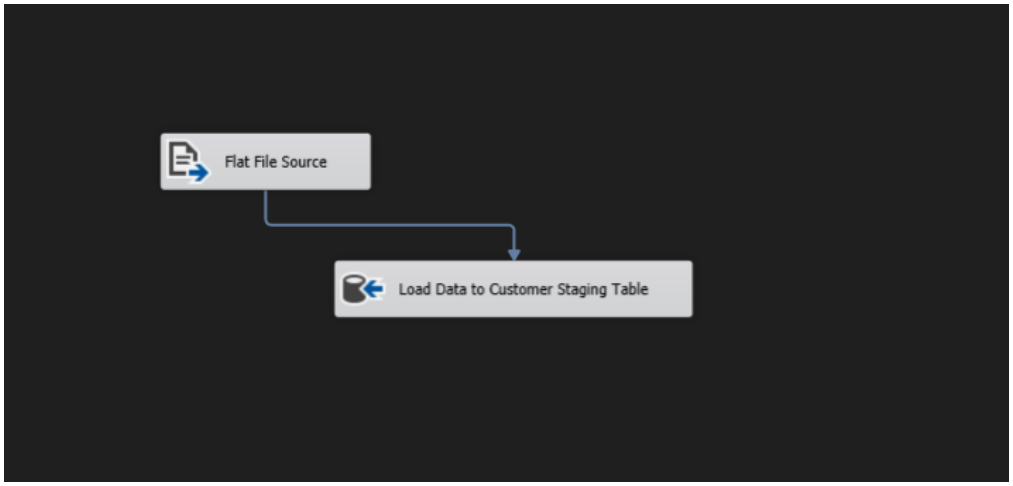
Used the OLE DB Source to connect to the source table dbo.orders table in the Ecommerce_OLTP source database and loaded that data to a staging table called dbo.StgOrders in the Ecommerce_OLTP_Staging database using a OLE DB Destination.



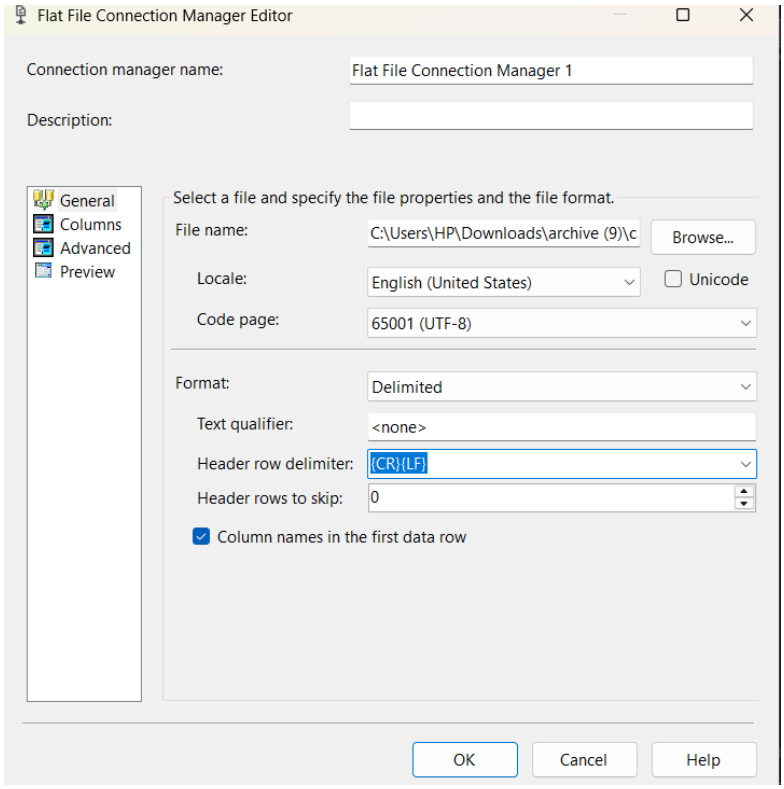
Used an Execute SQL task tool in SSIS to truncate the orders table for each time it's loaded to avoid duplicates.

This process was repeated for all the tables in the source database.

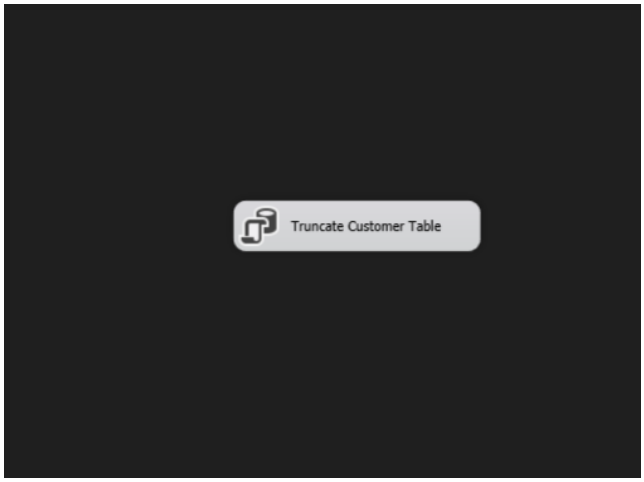
Extract Customer Data to Staging



Used a flat file source to extract data from the text file, which is considered as a flat file along with csv files.



Used the flat file connection manager to properly identify the columns and data types of the flat file data.

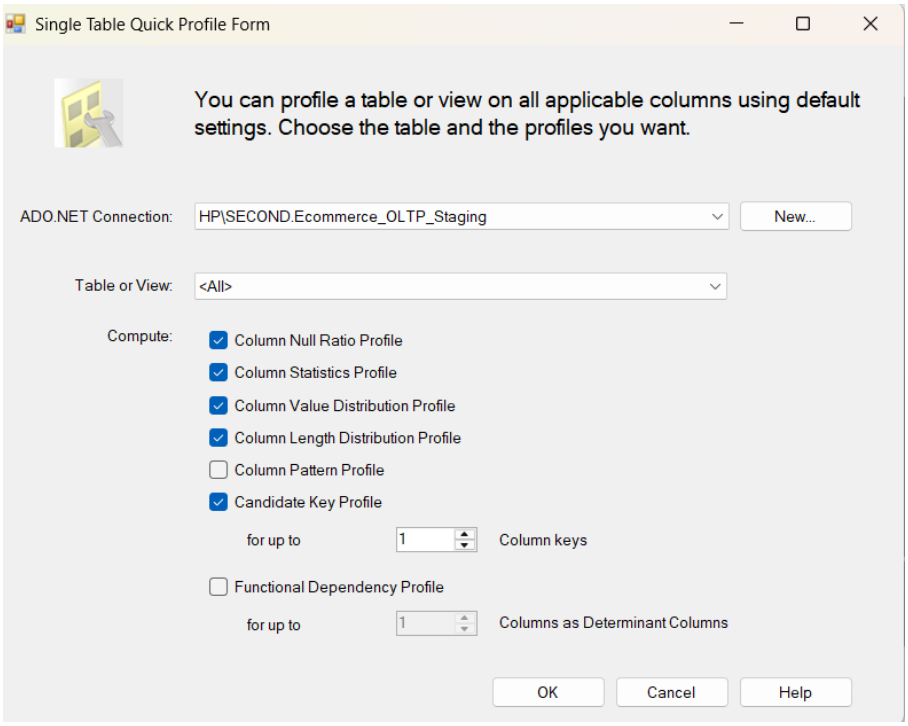


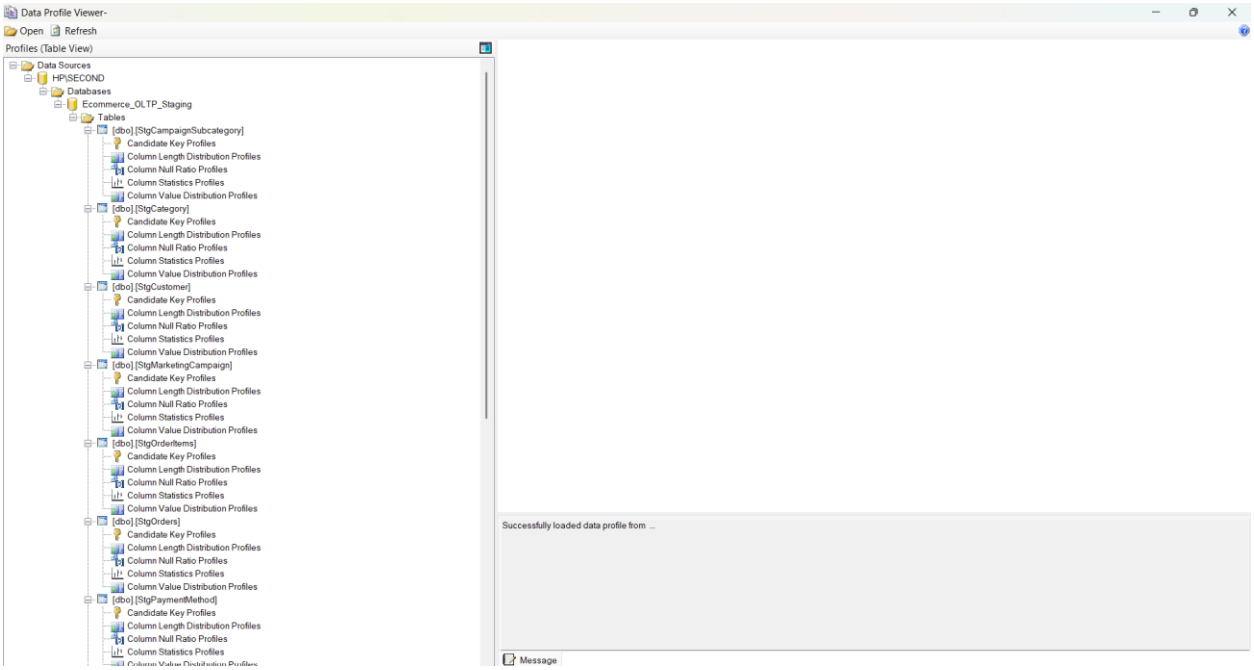
Then used an execute sql task to truncate the data from the file before loading to the table.

This method was then applied to the remaining two flat files as well until all the data was loaded into separate staging tables.

Data Profiling

I used the staging table data to do some data profiling to analyze the data and to determine and understand what transformations need to be done.



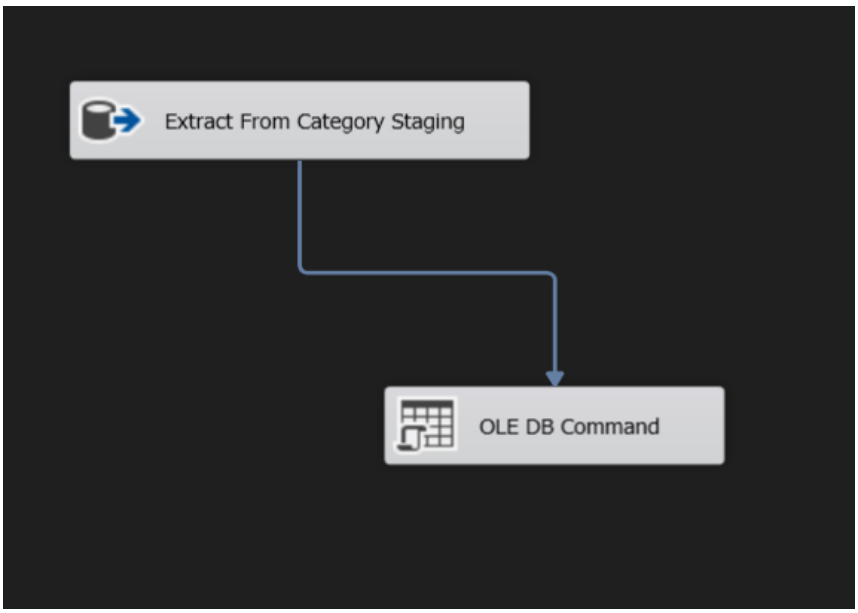


Data Transformation

To begin the Data Transformation section I first created a new package called Ecommerce_Load_DW. Considering the orders and hierarchy I first loaded the category data followed by the subcategory and product tables.

Transform and Load Category Data

I created a new Data Flow Task within the newly created package. Dragged and dropped a OLE DB Source to extract data from the Category Staging into the Dimension Table using an OLE DB Command that contained the sql command for executing the created procedure.

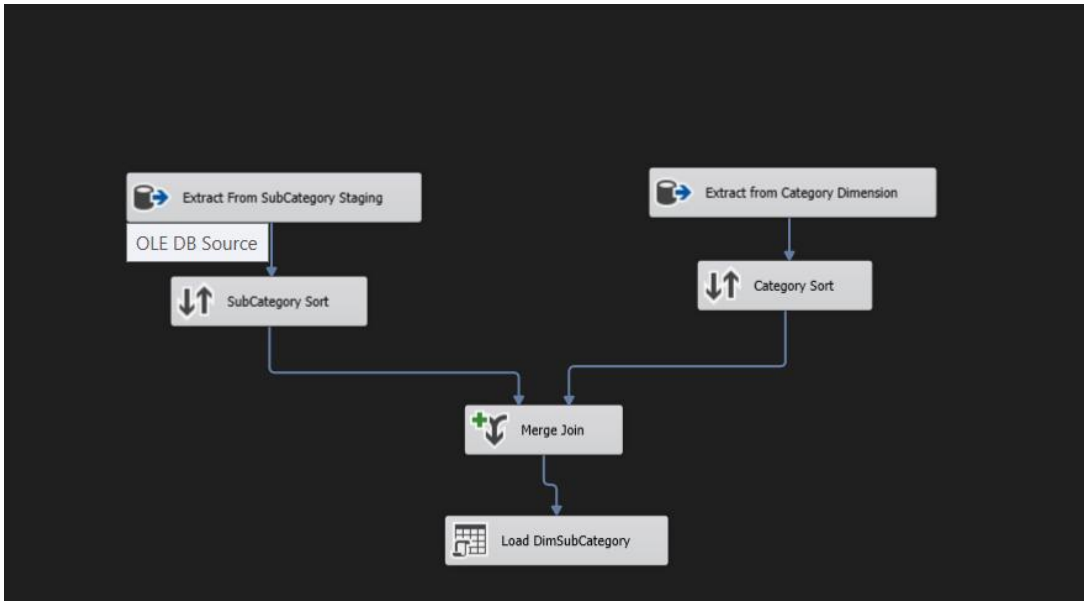


The procedure was created in the SSMS in the data warehouse database.

```sql
CREATE PROCEDURE dbo.UpdateDimCategory
    @CategoryID INT,
    @category_name NVARCHAR(100)
AS
BEGIN
    IF NOT EXISTS (
        SELECT category_key FROM dbo.DimCategory WHERE category_id = @CategoryID
    )
    BEGIN
        INSERT INTO dbo.DimCategory
        (category_id, category_name, insert_date, modified_date)
        VALUES
        (@CategoryID, @category_name, GETDATE(), GETDATE());
    END;

    IF EXISTS (
        SELECT category_key FROM dbo.DimCategory WHERE category_id = @CategoryID
    )
    BEGIN
        UPDATE dbo.DimCategory
        SET category_name = category_name,
            modified_date = GETDATE()
        WHERE category_id = @CategoryID;
    END;
END;
```

Transform and Load SubCategory Data

I created the DimSubCategory Table by sorting and merging the subcategory staging with the category dimension table as it uses the category id as well.

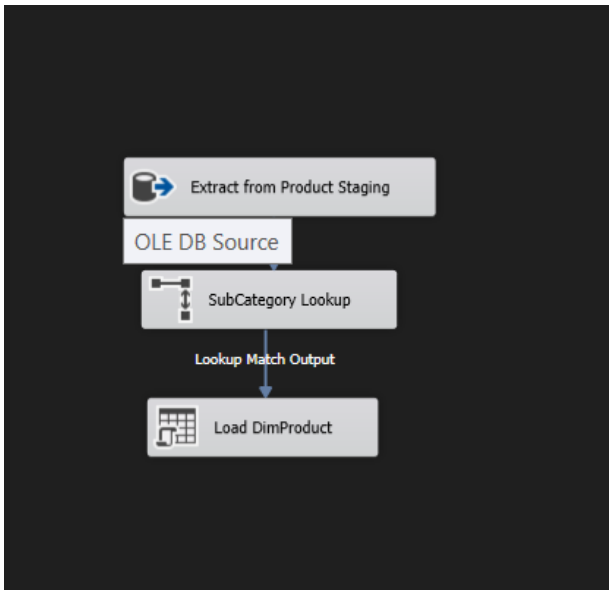


The procedure was written in the SSMS and the command to execute it was entered into the OLE DB Command.

```sql
CREATE PROCEDURE dbo.UpdateDimSubCategory
    @SubCategoryID TINYINT,
    @category_key INT,
    @SubCategoryName NVARCHAR(100)

AS
BEGIN
    IF NOT EXISTS (
        SELECT subcategory_key FROM dbo.DimSubCategory WHERE subcategory_id = @SubCategoryID
    )
    BEGIN
        INSERT INTO dbo.DimSubCategory
        (subcategory_id, category_key, subcategory_name, insert_date, modified_date)
        VALUES
        (@SubCategoryID, @category_key, @SubCategoryName, GETDATE(), GETDATE());
    END;

    IF EXISTS (
        SELECT category_key FROM dbo.DimSubCategory WHERE subcategory_id = @SubCategoryID
    )
    BEGIN
        UPDATE dbo.DimSubCategory
        SET subcategory_name = subcategory_name,
            modified_date = GETDATE()
        WHERE subcategory_id = @SubCategoryID;
    END;
END;
```

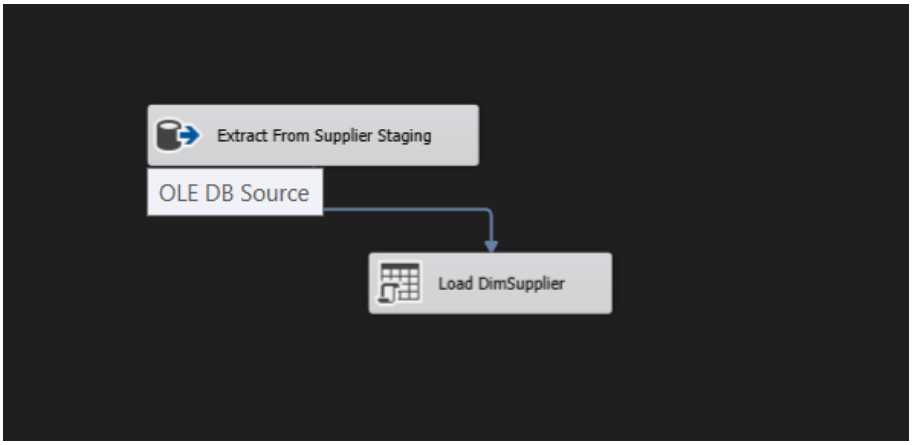
Transform and Load ProductData

For the Product table we used a lookup as the product table includes the subcategory ID and it is a much easier and simpler method rather than using the merge and sort components. We repeated this process for the other dimension tables.



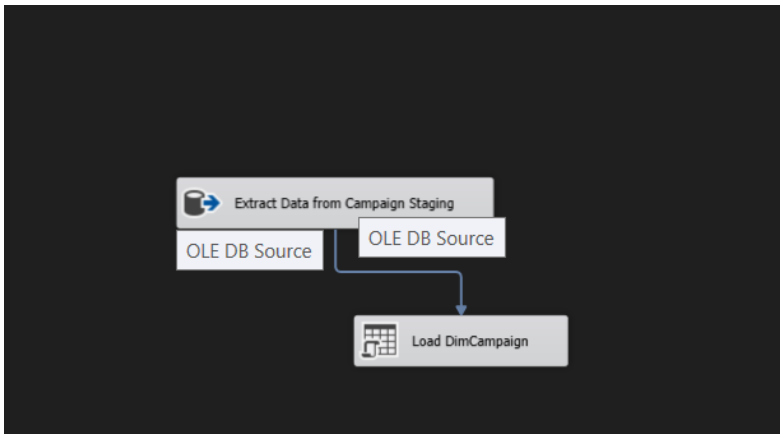

```sql
CREATE PROCEDURE dbo.UpdateDimProduct
    @ProductID INT,
    @Name NVARCHAR(50),
    @Price FLOAT,
    @Description NVARCHAR(100),
    @SubCategoryID TINYINT
AS
BEGIN
    IF NOT EXISTS (
        SELECT product_key FROM dbo.DimProduct WHERE product_id = @ProductID
    )
    BEGIN
        INSERT INTO dbo.DimProduct
        (product_id, name, price, description, subcategory_id, insert_date, modified_date)
        VALUES
        (@ProductID, @Name, @Price, @Description, @SubCategoryID, GETDATE(), GETDATE());
    END;

    IF EXISTS (
        SELECT product_key FROM dbo.DimProduct WHERE product_id = @ProductID
    )
    BEGIN
        UPDATE dbo.DimProduct
        SET name = @Name,
            price = @Price,
            description = @Description,
            subcategory_id = @SubCategoryID,
            modified_date = GETDATE()
        WHERE product_id = @ProductID;
    END;
END;
```

Transform and Load Supplier Data



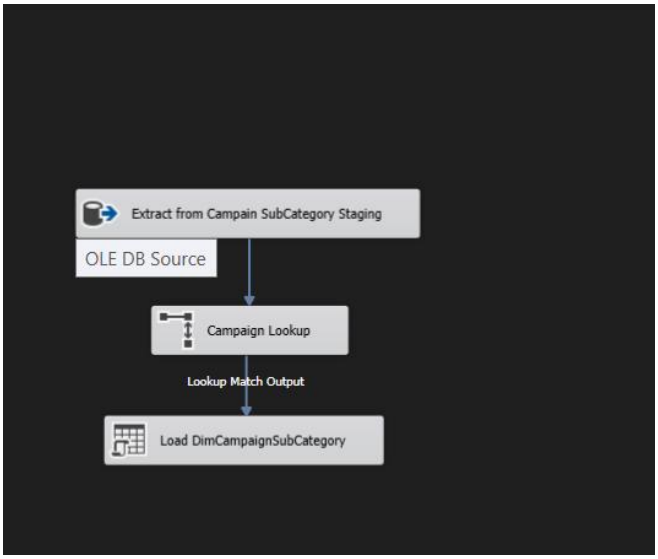

```sql
CREATE PROCEDURE dbo.UpdateDimSupplier
    @SupplierID INT,
    @SupplierName NVARCHAR(100),
    @SupplierEmail NVARCHAR(100)
AS
BEGIN
    IF NOT EXISTS (
        SELECT supplier_key FROM dbo.DimSupplier WHERE supplier_id = @SupplierID
    )
    BEGIN
        INSERT INTO dbo.DimSupplier
        (supplier_id, supplier_name, supplier_email, insert_date, modified_date)
        VALUES
        (@SupplierID, @SupplierName, @SupplierEmail, GETDATE(), GETDATE());
    END;

    IF EXISTS (
        SELECT supplier_key FROM dbo.DimSupplier WHERE supplier_id = @SupplierID
    )
    BEGIN
        UPDATE dbo.DimSupplier
        SET supplier_name = @SupplierName,
            supplier_email = @SupplierEmail,
            modified_date = GETDATE()
        WHERE supplier_id = @SupplierID;
    END;
END;
```

Transform and Load Campaign Data




```sql
CREATE PROCEDURE dbo.UpdateDimCampaign
    @CampaignID INT,
    @CampaignName NVARCHAR(100),
    @OfferWeek TINYINT
AS
BEGIN
    IF NOT EXISTS (
        SELECT campaign_key FROM dbo.DimCampaign WHERE campaign_id = @CampaignID
    )
    BEGIN
        INSERT INTO dbo.DimCampaign
        (campaign_id, campaign_name, offer_week, insert_date, modified_date)
        VALUES
        (@CampaignID, @CampaignName, @OfferWeek, GETDATE(), GETDATE());
    END;

    IF EXISTS (
        SELECT campaign_key FROM dbo.DimCampaign WHERE campaign_id = @CampaignID
    )
    BEGIN
        UPDATE dbo.DimCampaign
        SET campaign_name = @CampaignName,
            offer_week = @OfferWeek,
            modified_date = GETDATE()
        WHERE campaign_id = @CampaignID;
    END;
END;
```

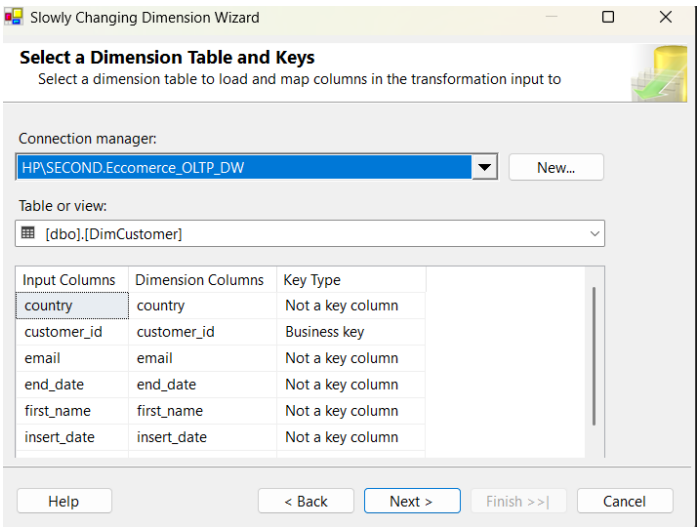
Transform and Load Campaign SubCategory Data

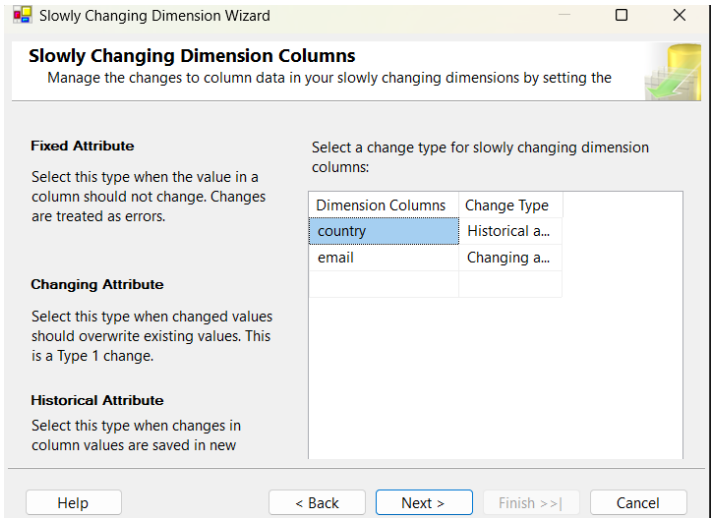


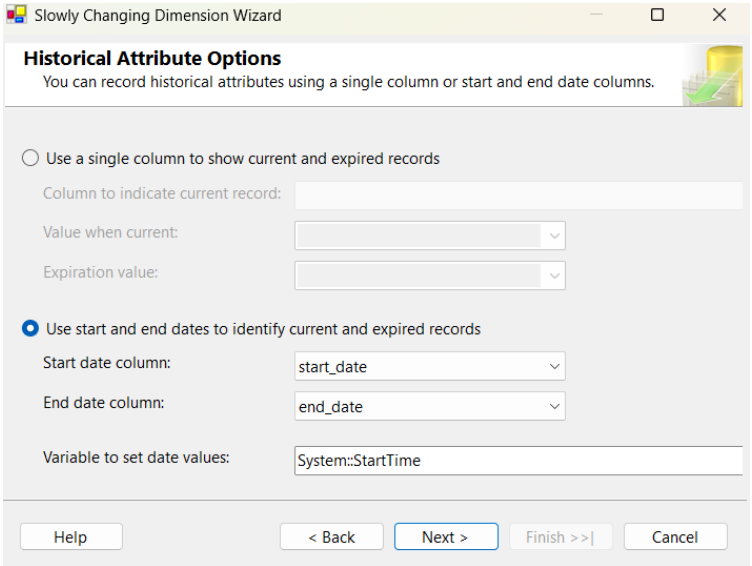

```sql
CREATE PROCEDURE dbo.UpdateDimCampaignSubcategory
    @CampaignID INT,
    @SubCategoryID INT,
    @Discount NVARCHAR(16)
AS
BEGIN
    IF NOT EXISTS (
        SELECT campaign_subcategory_key FROM dbo.DimCampaignSubcategory
        WHERE campaign_id = @CampaignID AND subcategory_id = @SubCategoryID
    )
    BEGIN
        INSERT INTO dbo.DimCampaignSubcategory
        (campaign_id, subcategory_id, discount, insert_date, modified_date)
        VALUES
        (@CampaignID, @SubCategoryID, @Discount, GETDATE(), GETDATE());
    END;

    IF EXISTS (
        SELECT campaign_subcategory_key FROM dbo.DimCampaignSubcategory
        WHERE campaign_id = @CampaignID AND subcategory_id = @SubCategoryID
    )
    BEGIN
        UPDATE dbo.DimCampaignSubcategory
        SET discount = @Discount,
            modified_date = GETDATE()
        WHERE campaign_id = @CampaignID AND subcategory_id = @SubCategoryID;
    END;
END;
```

Transform and Load Customer Data(SCD)

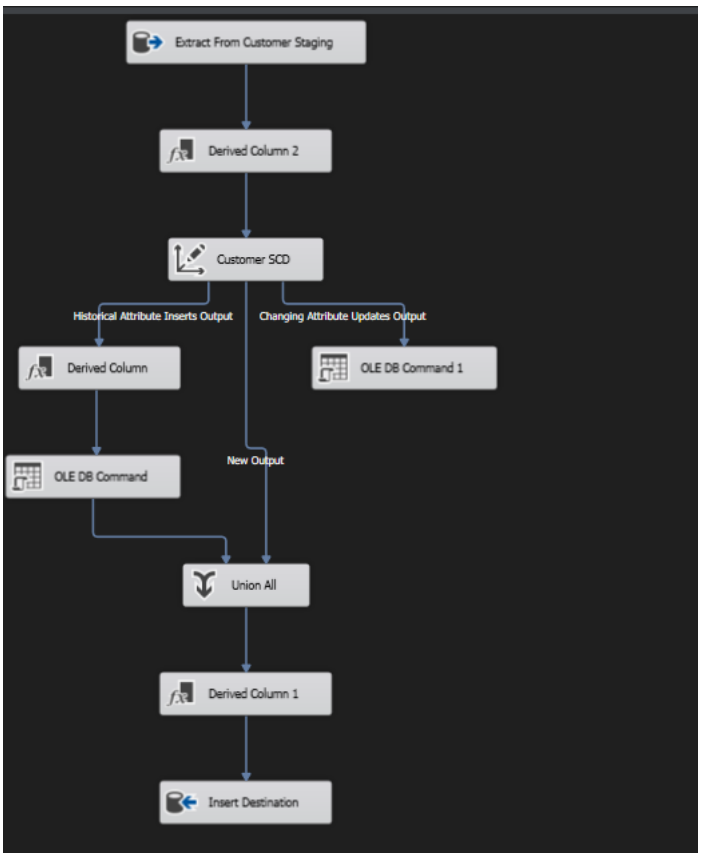
After extracting from the customer table and using a derived column to get the insert date and start date I dragged and dropped a SCD. I then set the following configurations.







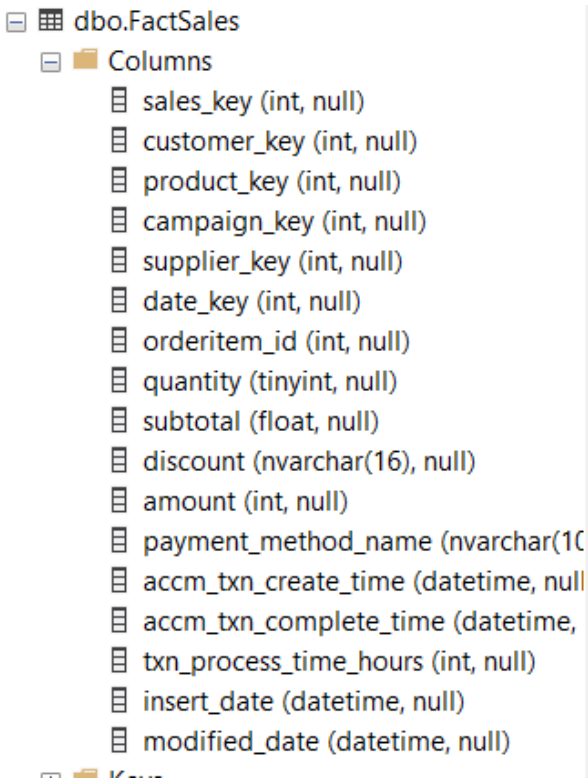
Once it has all been configured the rest will be automatically be generated as seen below.


Extract From Customer Staging
Derived Column 2
Customer SCD
Historical Attribute Inserts Output
Changing Attribute Updates Output
Derived Column
OLE DB Command 1
OLE DB Command
New Output
Union All
Derived Column 1
Insert Destination

STEP 06: ETL Development -Accumulating Fact Table

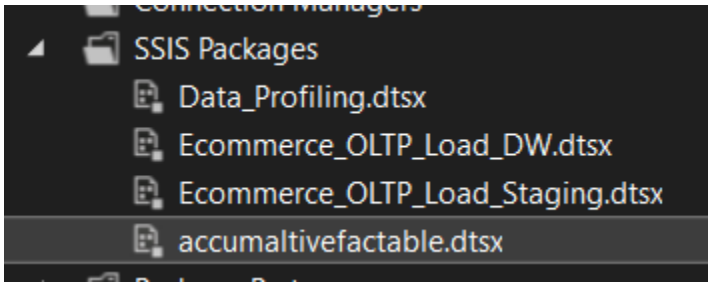
First, I extended my sales fact table with following 03 columns.

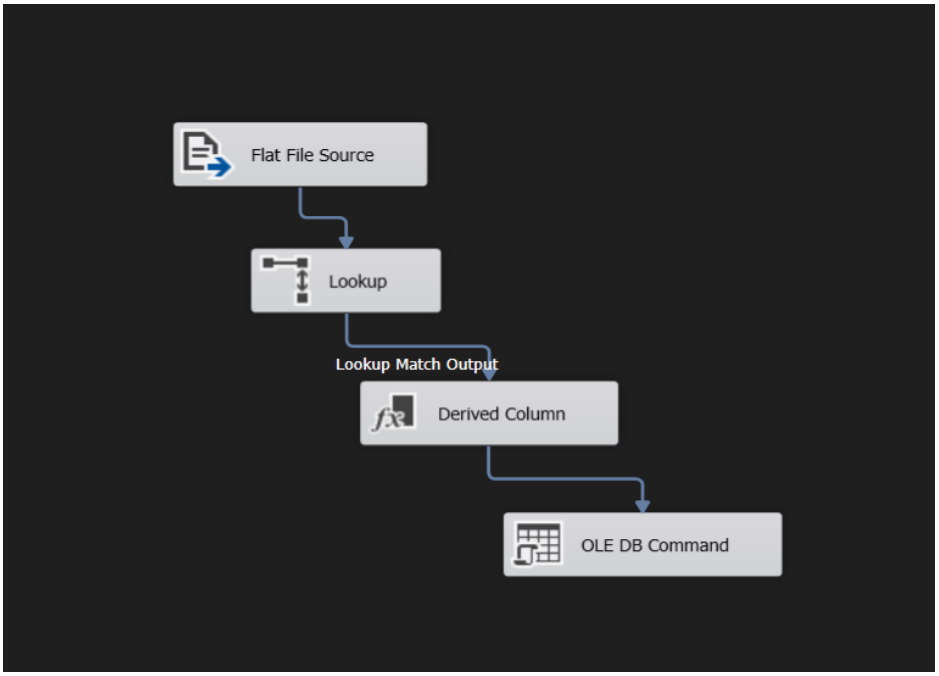
accm_txn_create_time
accm_txn_complete_time
txn_process_time_hours



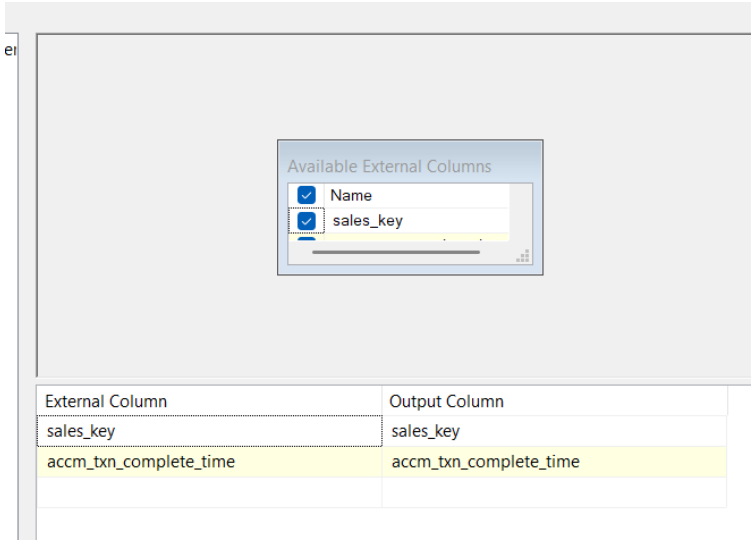
I then prepared a separate dataset for the complete time.

After that I created a new package in which I created a new dataflow task. This will be to receive updates and update the aam_txn_complete_time accordingly.

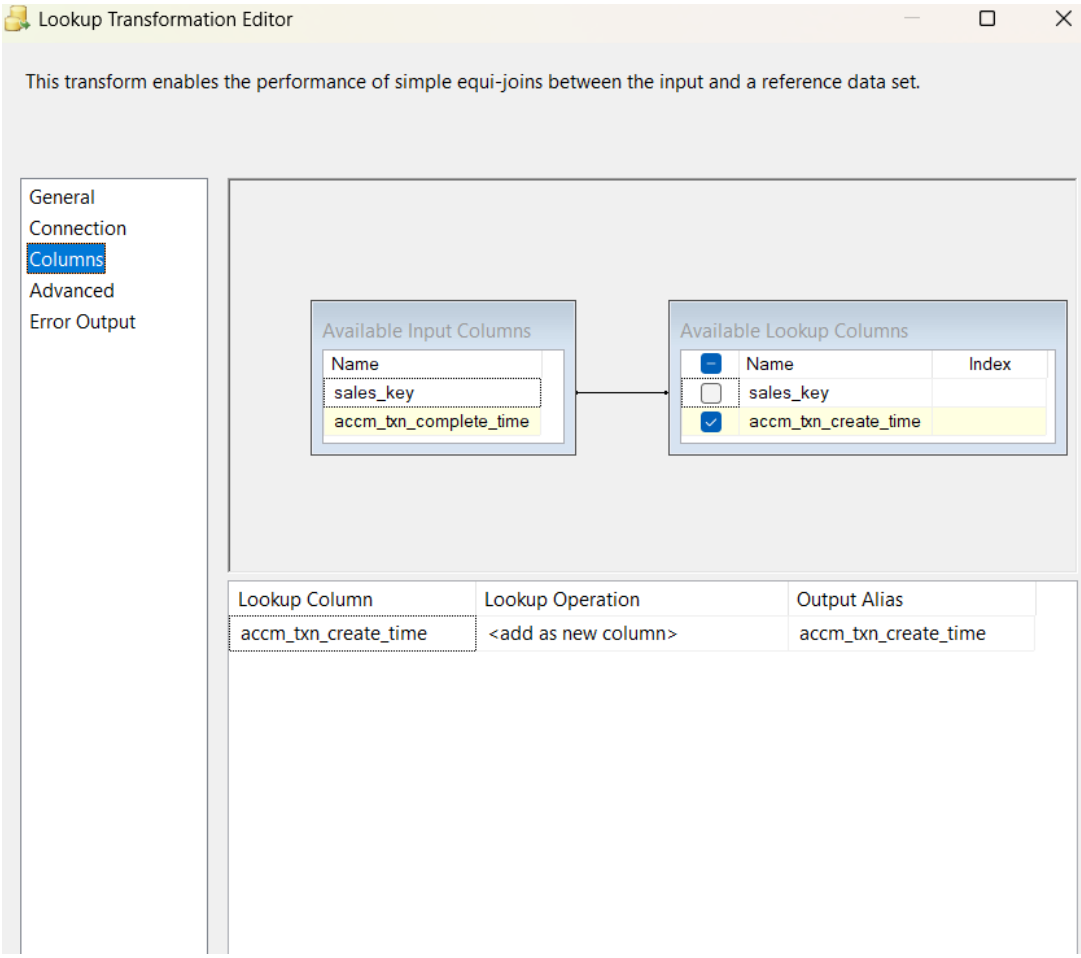




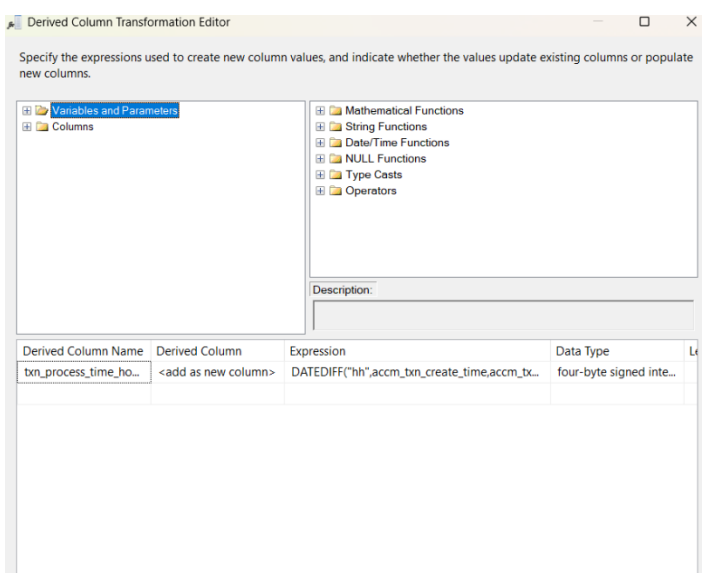
I extracted the data from the flat file



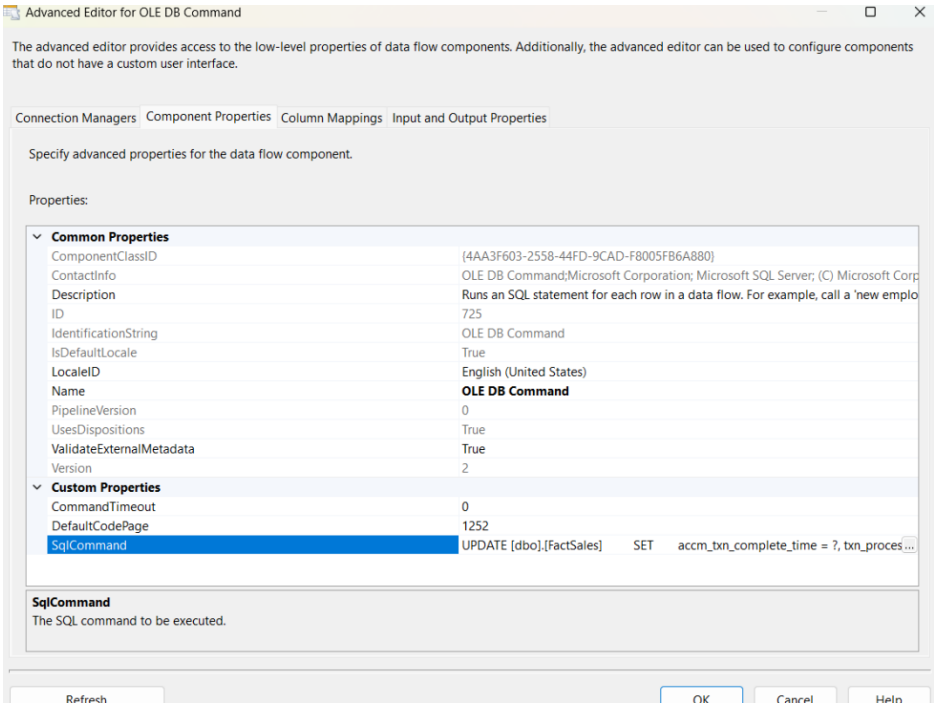
Used the lookup to match with the factSales table using the sales_key and to retrieve the accm_txn_create_time value which we need for the calculation of txn_process_time_hours.

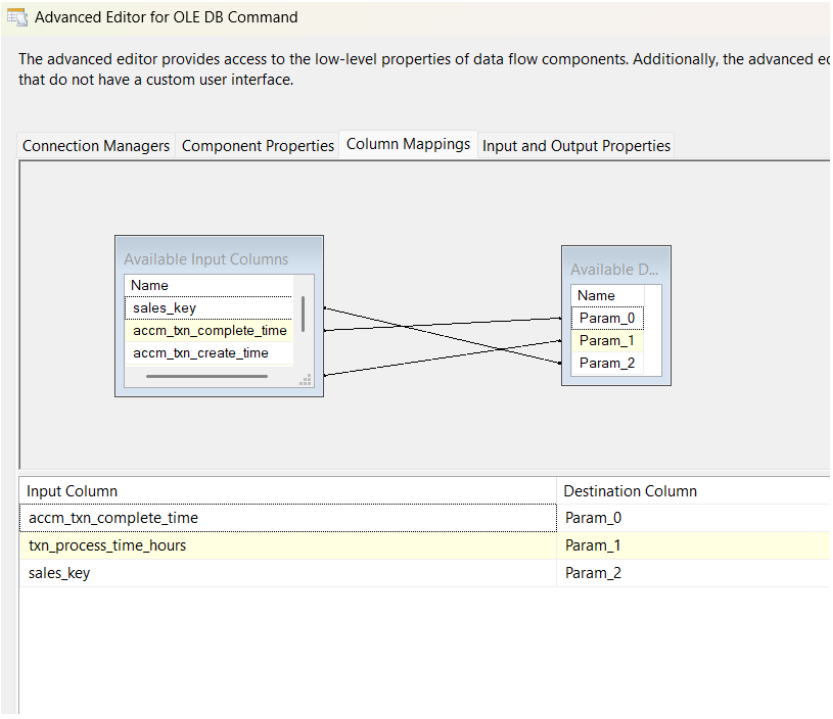


I then used the derived column to generate the value for the txn_process_time_hours and to add the column.



Finally in the OLE DB Command we connected it to the Data Warehouse database and gave an sql command to update the factsales table columns accordingly and mapped the necessary parameters.





The results

121 %

Results | Messages

	sales_key	customer_key	product_key	campaign_key	supplier_key	date_key	orderitem_id	quantity	subtotal	discount	amount	payment_method_name	accm_txn_create_time	accm_txn_complete_time	txn_process_time_hours	insert_date
12...	36700	125	430	4	48	20210708	201644	7	3082.65991210938	00:21:00.0000000	16268	credit card	2025-05-01 18:07:07.327	2025-05-03 23:48:22.000	53	2025-05-01 1
12...	36700	125	271	4	14	20210708	201645	7	2324.48999023438	00:06:00.0000000	16268	credit card	2025-05-01 18:07:07.327	2025-05-03 23:48:22.000	53	2025-05-01 1
12...	36700	125	248	4	41	20210708	201646	4	3159.15991210938	00:24:00.0000000	16268	credit card	2025-05-01 18:07:07.327	2025-05-03 23:48:22.000	53	2025-05-01 1
12...	36700	125	69	4	17	20210708	201647	9	2150.01000976563	00:11:00.0000000	16268	credit card	2025-05-01 18:07:07.327	2025-05-03 23:48:22.000	53	2025-05-01 1
12...	36700	125	386	4	43	20210708	201649	7	2919.2099609375	00:10:00.0000000	16268	credit card	2025-05-01 18:07:07.327	2025-05-03 23:48:22.000	53	2025-05-01 1
12...	36701	195	349	NULL	37	20210123	201653	1	155.839996337891	00:00:00.0000000	6698	credit card	2025-05-01 18:07:07.327	2025-05-07 07:39:22.000	133	2025-05-01 1
12...	36701	195	28	NULL	39	20210123	201652	2	84.1800003051758	00:00:00.0000000	6698	credit card	2025-05-01 18:07:07.327	2025-05-07 07:39:22.000	133	2025-05-01 1
12...	36701	195	244	NULL	31	20210123	201651	7	5904.919921875	00:00:00.0000000	6698	credit card	2025-05-01 18:07:07.327	2025-05-07 07:39:22.000	133	2025-05-01 1
12...	36701	195	86	NULL	18	20210123	201650	9	553.140014648438	00:00:00.0000000	6698	credit card	2025-05-01 18:07:07.327	2025-05-07 07:39:22.000	133	2025-05-01 1
12...	36702	370	22	NULL	35	20220331	201654	10	986.400024414063	00:00:00.0000000	986	credit card	2025-05-01 18:07:07.327	2025-05-05 12:37:22.000	90	2025-05-01 1
12...	36703	168	373	15	15	20180619	201662	1	653.440002441406	00:11:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1
12...	36703	168	430	15	2	20180619	201661	1	440.380004882813	00:11:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1
12...	36703	168	421	15	5	20180619	201660	8	3736.39990234375	00:15:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1
12...	36703	168	395	15	47	20180619	201659	1	185.800003051758	00:18:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1
12...	36703	168	222	15	46	20180619	201657	9	7986.06005859375	00:15:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1
12...	36703	168	275	15	39	20180619	201656	7	4303.18017578125	00:25:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1
12...	36703	168	172	15	11	20180619	201655	7	3898.51000976563	00:17:00.0000000	22396	cash	2025-05-01 18:07:07.327	2025-05-06 01:37:22.000	103	2025-05-01 1

Final Control flow of the Data Warehouse