Sri Lanka Institute of Information Technology

IT3021 - Data Warehouse and Business Intelligence

Assignment 1

Name : H.M.H.D. Premarathne
Reg-No : IT20006334
Batch : Y3.S1.WD.DS.05.01

Contents

1. DATA SET SELECTION

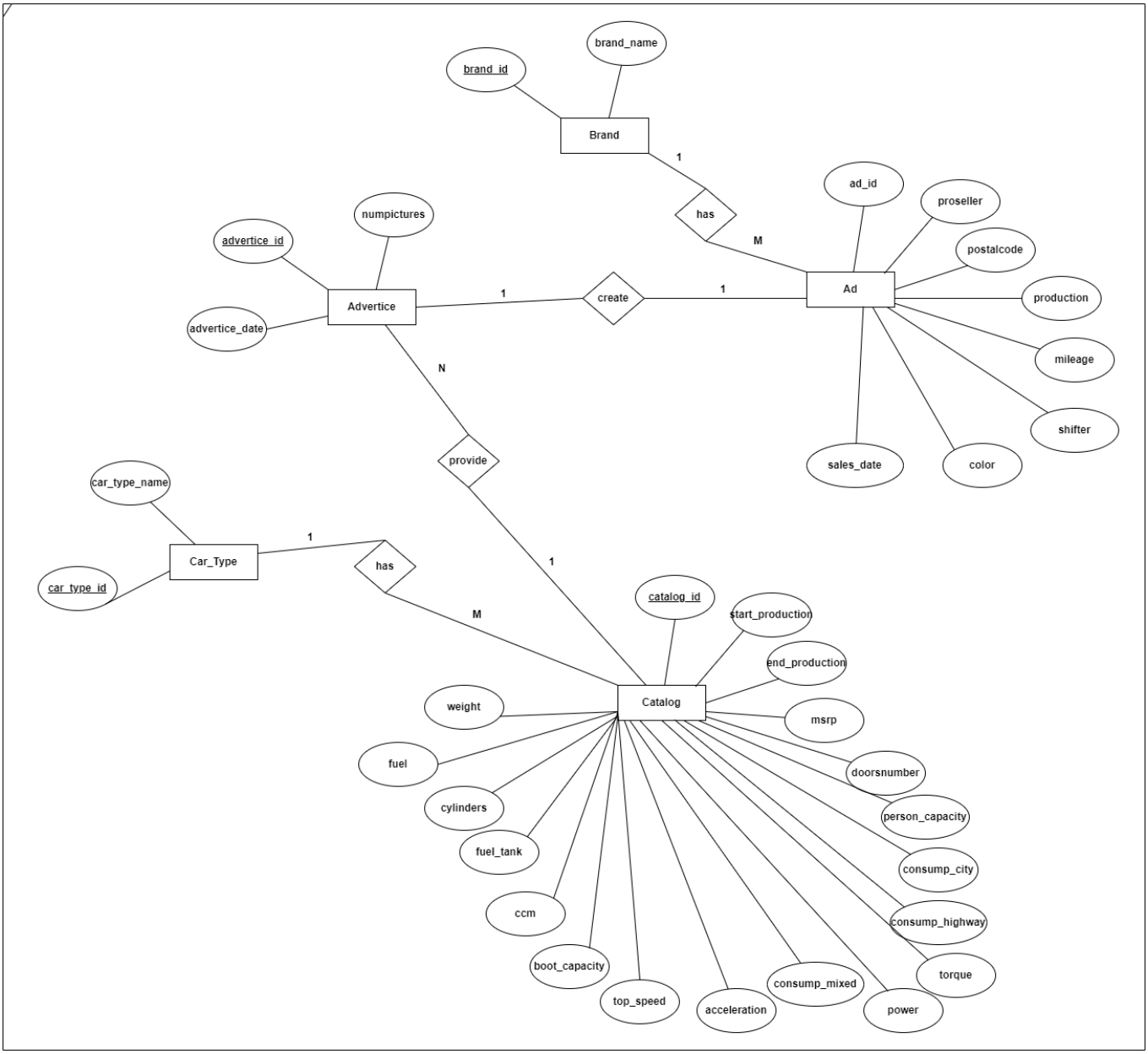
Data Set Name	:	Secondhand Car Market Hungary
Provided by	:	www.kaggle.com
Source link	:	https://www.kaggle.com/datasets/attilakiss/secondhand-car-market-data-parsing-dataset-v1

About Dataset

The selected data source is a collection of transactional data. The dataset originates from the biggest used car advertisement site of Hungary. There are two main tables in this dataset. They are Advertisements (Ad) and Catalogs (Catalog). Advertisements table contain advertisement details such as mileage, vehicle color, sales date, production date likewise. Catalog table contain catalog details such as start and end production dates, no of cylinders, top speed, torque, power, weight likewise. Advertisement table connect to Brands (Brand) table. That table contain all vehicle brands. And also catalog table connect to Car Type (CarType) table. That table contain all vehicle types in the world. Advertisement table and Catalog table both connect to Advertise table. Advertise table contain advertising details such as adverting date, number of pictures of vehicle, advertisement price likewise. Some modifications were done accordingly to the data set derived from the source.

- Adverticement.csv : Contains the essential information about the car advertisements
- Catalog.csv : Contain the information about car catalogs
- Brand.csv : Contain all car brands in the car market
- Car Type.csv : Contain all car types in the car market

ER - Diagram



- This diagram shows the relationships between entities in this dataset
- Assumptions :-
    - One car should have only one catalog
    - Catalog and Advertisement can be connected with only Advertise

2. PREPARATION OF DATA SOURCES

Although there are multiple csv files in the data set originally, due to high percentage of duplication of data I took Advertisement.csv and Catalog.csv as my base data sets and I separately create another data set called Advertice.csv with doing some necessary changes (Adding some columns on my own) and converted them to xls (Microsoft Excel 97 – 2003 Workbook) format for the purpose of creating a database file as a source.

Final State of Preparation of the source data formats before Transforming data

- Car Market (Source Database) Tables
  - ✓ dbo.Ad
  - ✓ dbo.Advertice
  - ✓ dbo.Catalog

- Car Type table that has been taken as a separate dataset
  - ✓ CarType.csv

- Text file that has been taken as a separate source type
  - ✓ Brand.txt

Description of the data set

1. Source Type : Car_Market_DB
   Table Name : Ad

Column	Data Type	Description
ad_id	int	ID of an advertisement (PK)
proseller	nchar(10)	Professional Seller or not
postalcode	int	Zip code of city
production	datetime	Production year of the car
mileage	int	Milage of the car
shifter	nchar(10)	Gear shifter type
color	int	Color code
brand_id	int	Brand ID of the car (FK – Brand table)
sales_date	datetime	Car sold date
is_sold	nchar(10)	Is sold or not

2. Source Type : Car_Market_DB
   Table Name : Catalog

Column	Data Type	Description
catalog_id	int	ID of a Catalog (PK)
car_type_id	int	Car Type ID of the car (FK – CarType table)
start_production	datetime	First year of the production
end_production	datetime	Last year of the production
msrp	int	Price as new
doorsnumber	int	Number of doors
person_capacity	int	Number of passengers with driver
weight	int	Weight of the car
fuel_tank	int	Fuel tank capacity
boot_capacity	int	Trunk capacity
fuel	nvarchar(50)	Fuel type
cylinders	int	Number of cylinders
ccm	int	Engine size
consump_city	real	City fuel consumption
consump_highway	real	Highway fuel consumption
consump_mixed	real	Mixed fuel consumption
top_speed	int	Top speed
acceleration	real	Acceleration to 100 km/h
torque	int	Torque in NM
power	int	Horsepower

3. Source Type : Car_Market_DB
   Table Name : Advertice

Column Name	Data Type	Description
advertice_id	int	ID of a Advertise (PK)
ad_id	int	Advertisement ID (FK – Ad table)
ad_price	int	Price of the advertisement
numpictures	int	Number of pictures in advertisement
catalog_id	int	Catalog ID (FK – Catalog table)
advertice_date	datetime	Advertised date

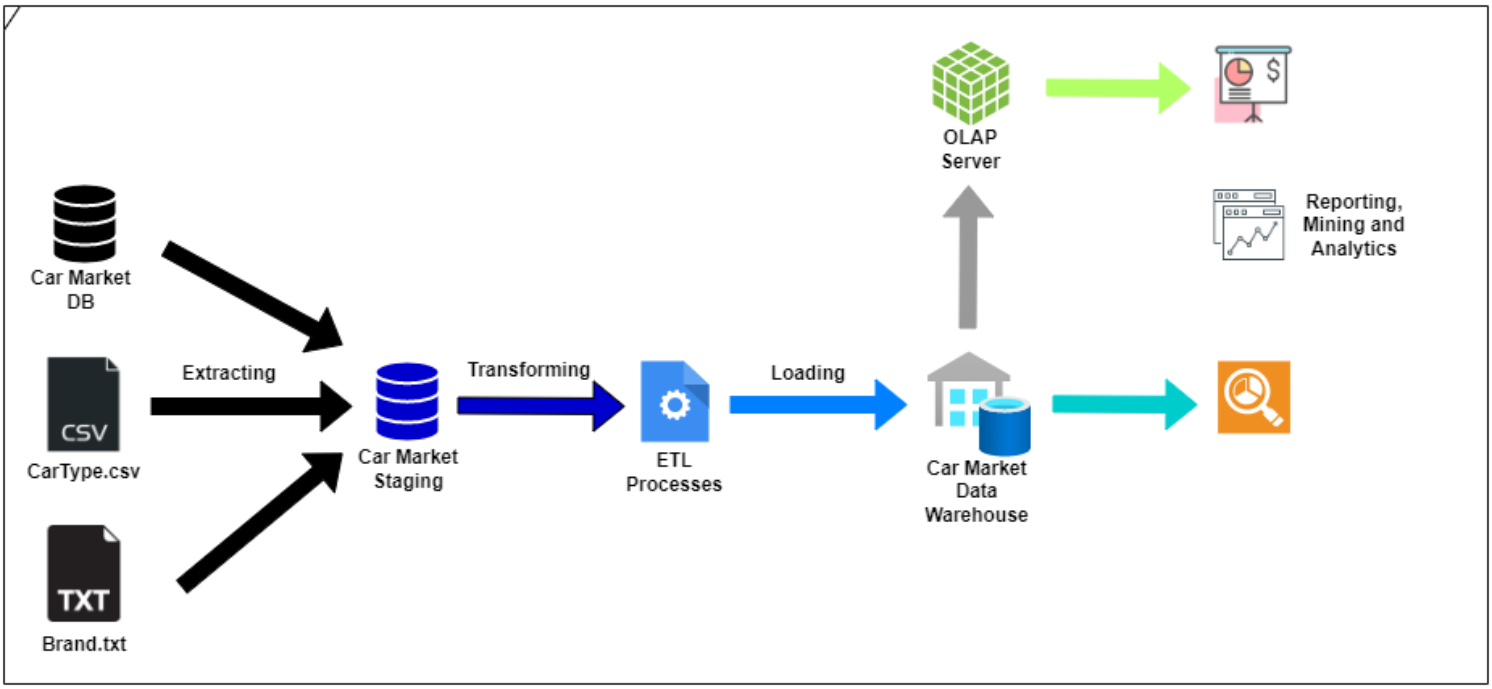
4. Source Type : Brand.txt
   Table Name : Brand

Column Name	Data Type	Description
brand_id	int	ID of a Brand (PK)
brand_name	nvarchar(50)	Car brand name

5. Source Type : CarType.csv
   Table Name : CarType

Column Name	Data Type	Description
car_type _id	int	ID of a Car Type (PK)
car_type_name	nvarchar(50)	Car type name

3. SOLUTION ARCHITECTURE

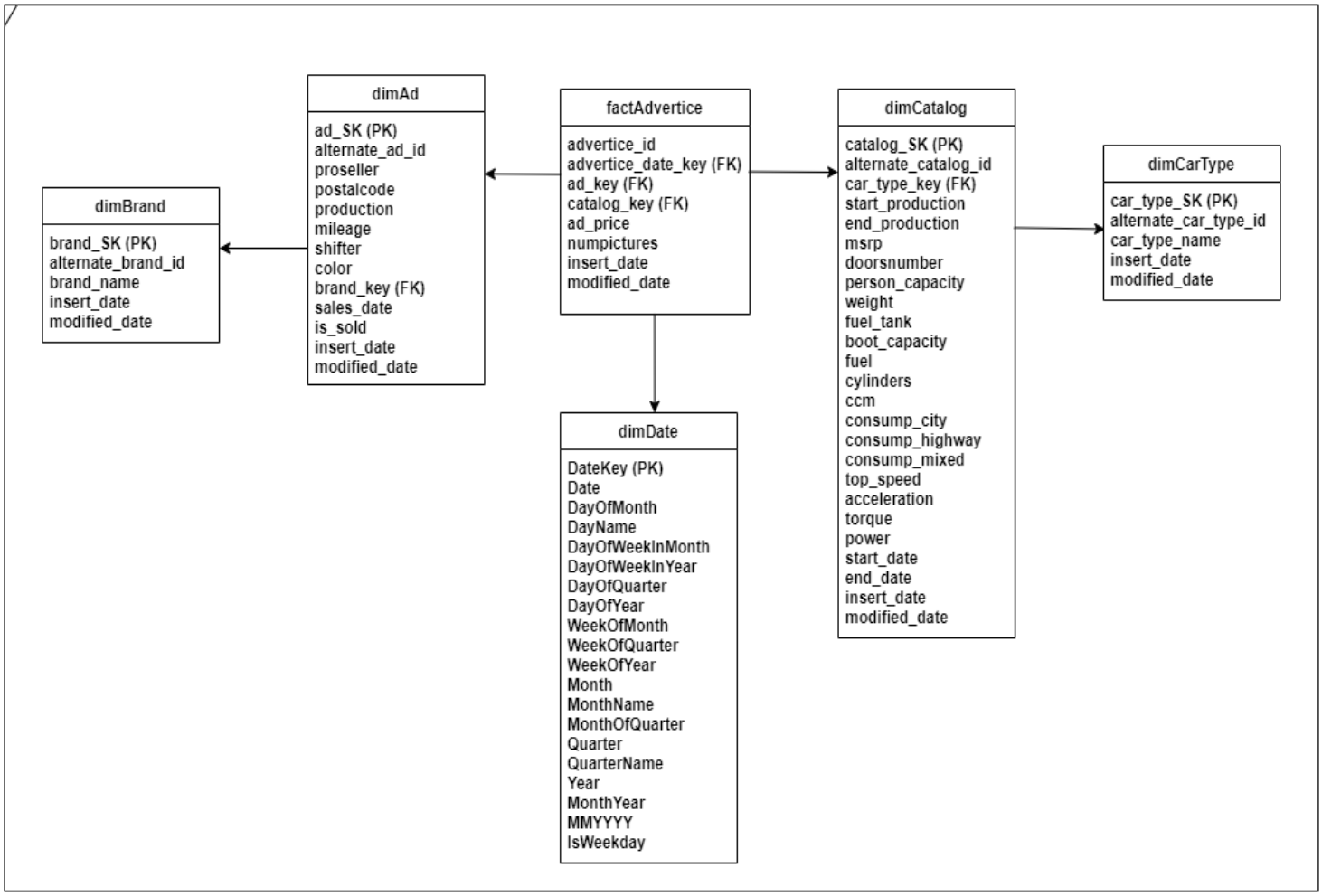


- As this architecture shows for the ETL processing, first **Car_Market_DB** (Source database) and then **CarType.csv** and **Brand.txt** has been used for the data extraction to the staging process. After that staged in **Car_Market_Staging** (Staging database) data are transforming and loading to **Car_Market_DW** (Data warehouse) and that data can be used for reporting, mining and analyzing purposes.

4. DATA WAREHOUSE DESIGN & DEVELOPMENT

I. Design

The Car_Market_DW (Data Warehouse) is designed according to a Snowflake schema as shown in this following figure with one fact table (dbo.factAdvertice) and five dimension tables including the Date dimension.



- Hierarchies

  dimDate is consisted with the hierarchy of dates which includes DayofMonth, Month, Year.

II. **Assumptions**

- dbo.dimDate is added to the Data Warehouse for better performance.
- dbo.factAdvertice is used in creating the fact table.

III. Slowly changing dimensions

Catalog Details are considered as a slowly changing dimension (dimCatalog)

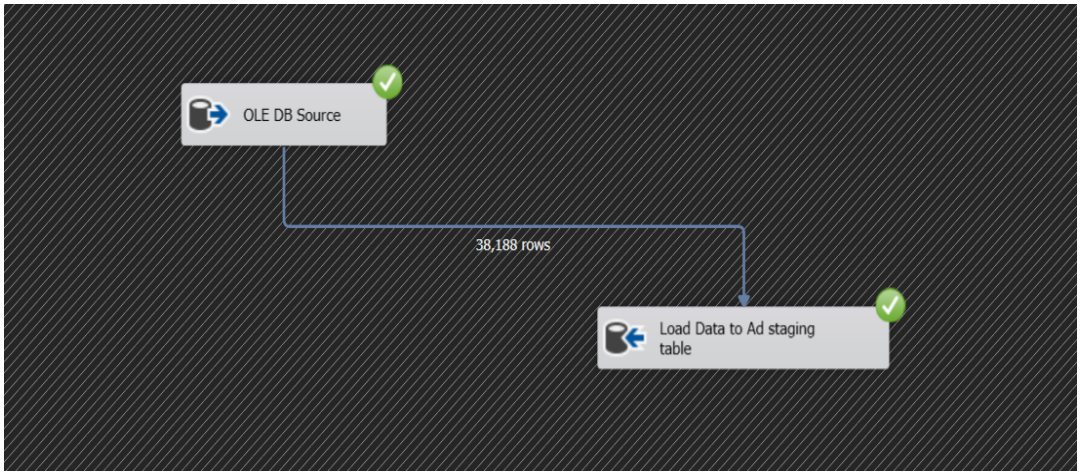
Dimension Table	Attributes
dbo.dimCatalog	msrp(Changing attribute) weight (Historical attribute) fuel_tank (Historical attribute) fuel (Historical attribute) boot_capacity (Historical attribute) cylinders (Historical attribute) ccm (Historical attribute) consump_city (Historical attribute) consump_highway (Historical attribute) consump_mixed (Historical attribute) top_speed (Historical attribute) acceleration (Historical attribute) torque (Historical attribute) power (Historical attribute) doorsnumber (Historical attribute) person_capacity (Historical attribute)

5. ETL DEVELOPMENT

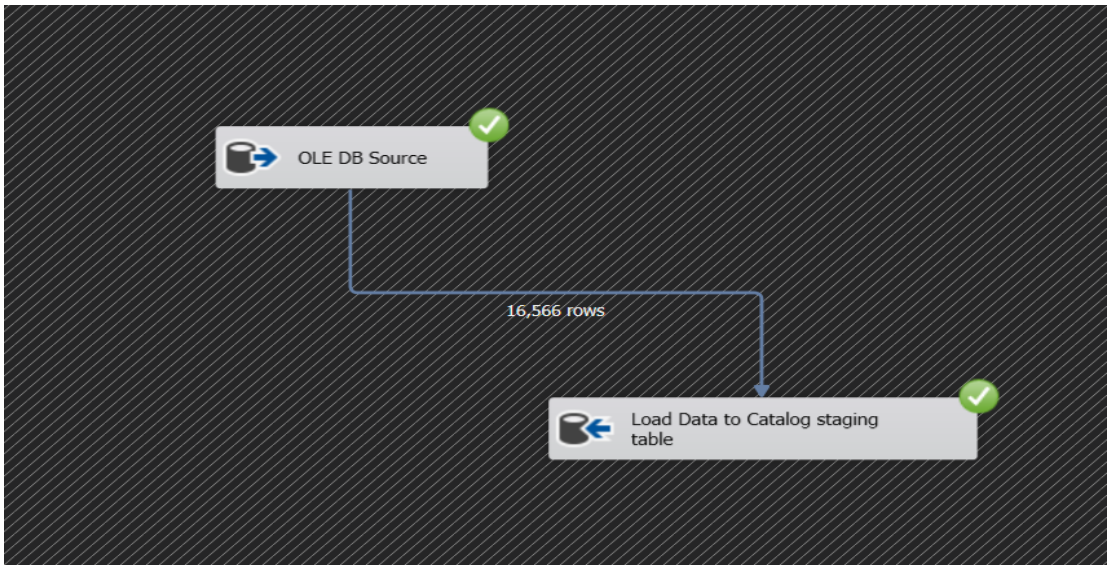
I. Data Extraction & Load into Staging tables

- Data Extraction is done by using the provided data sources mentioned above in Visual Studio 2019 (Data Tool) development environment. The text file , csv file and the source database were used here.
- Initially, **OLE DB SOURCE** (for source database) or **FLAT FILE SOURCE** (for txt file and csv file) is used to extract data for the Staging criteria. In this step developer can select the columns what would be included in the Staging from available data columns. As the next step of Staging, **OLE DB DESTINATION** has applied here to storing data in the Staging tables of **Car_Market_Staging**

❖ Advertisement data is extracted from Ad table in the data source database and inserted to the stgAd table.

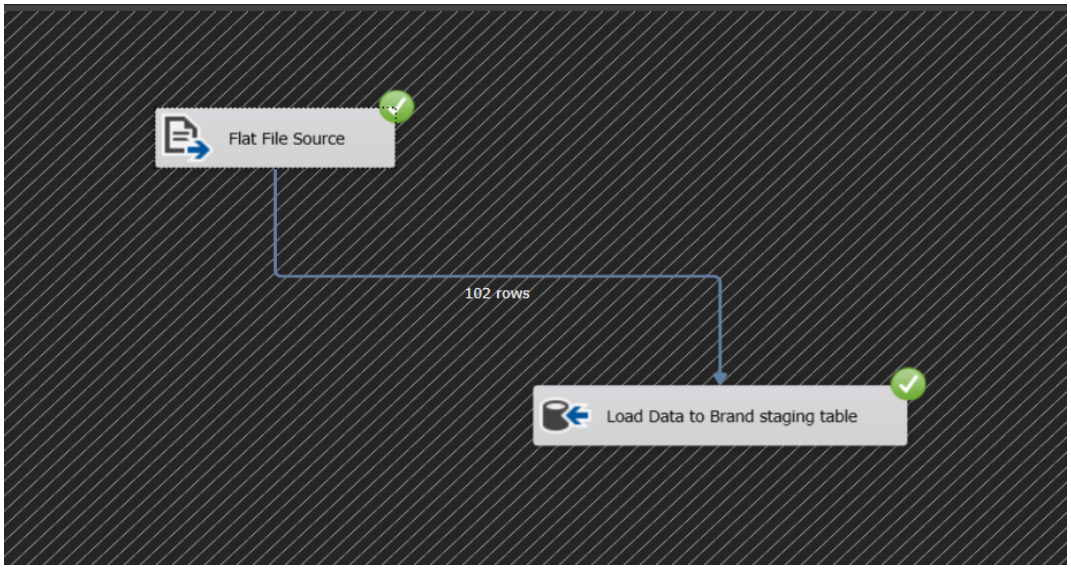


❖ Catalog data is extracted from Catalog table in the data source database and inserted to the stgCatalog table.

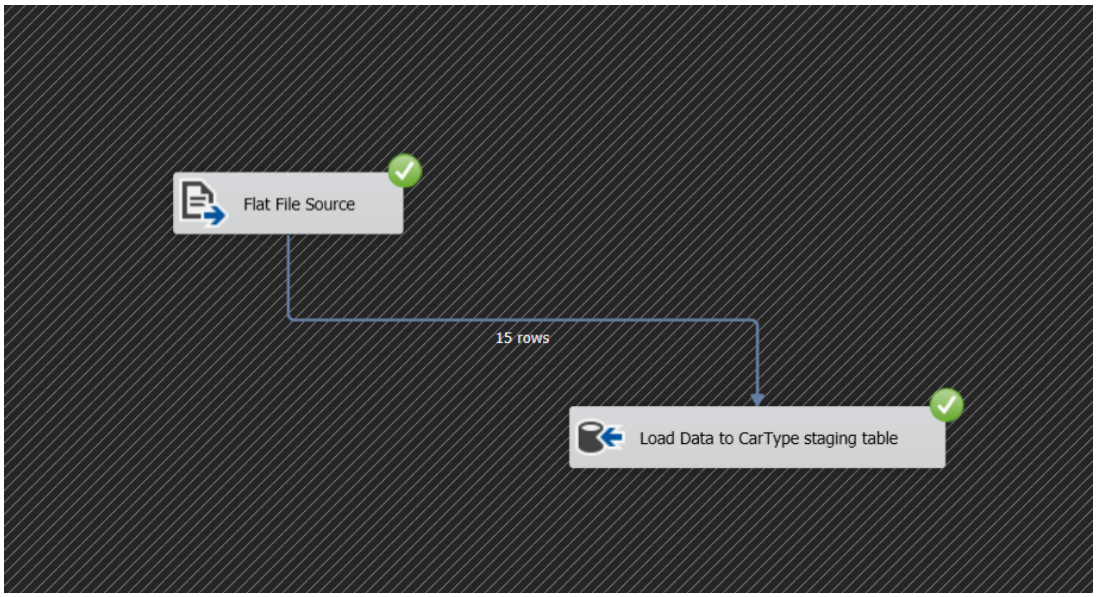


❖ Advertise data is extracted from Advertise table in the data source database and inserted to the stgAdvertise table.

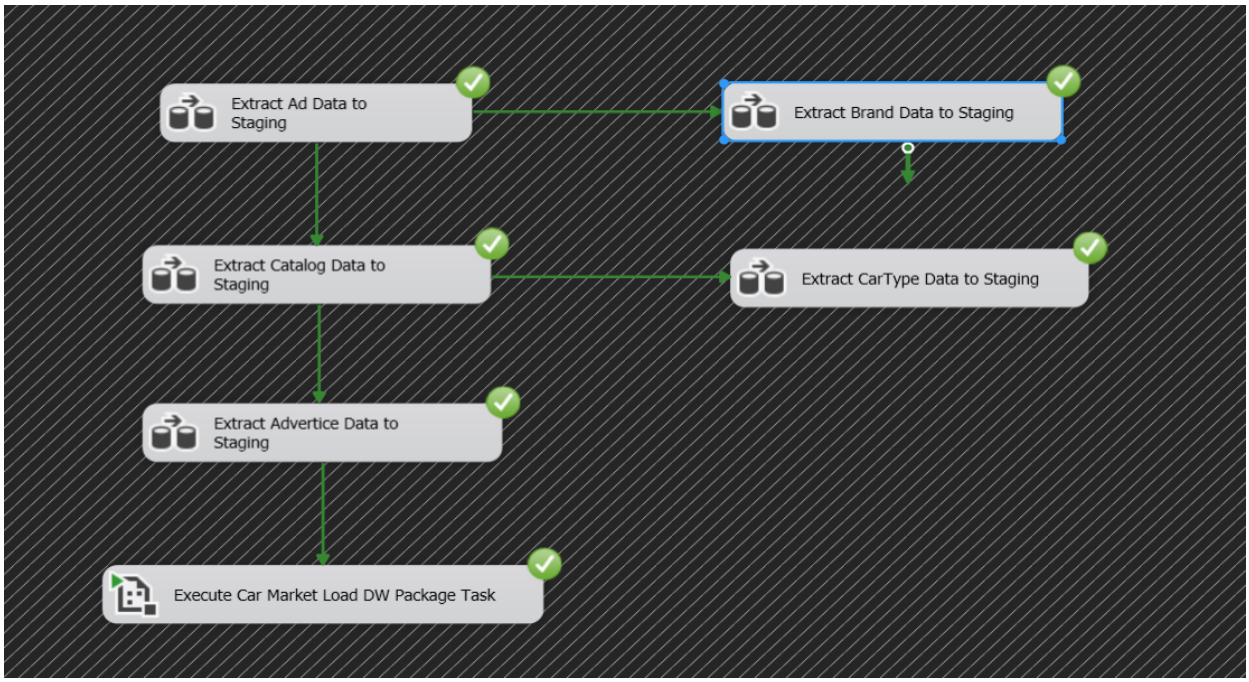
❖ Brand data is extracted from the text file of Brand.txt and inserted to the stgBrand table.



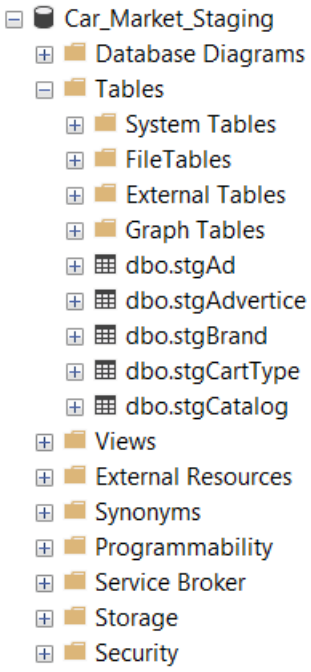
❖ Brand data is extracted from the csv file of CarType.csv and inserted to the stgCarType table.



❖ The Control Flow of Extract Data and Load into Staging can be shown in this figure.



- In here, I have used a Execute package task called 'Execute Car Market Load DW Package Task to execute both staging and data warehouse packages at the same time.



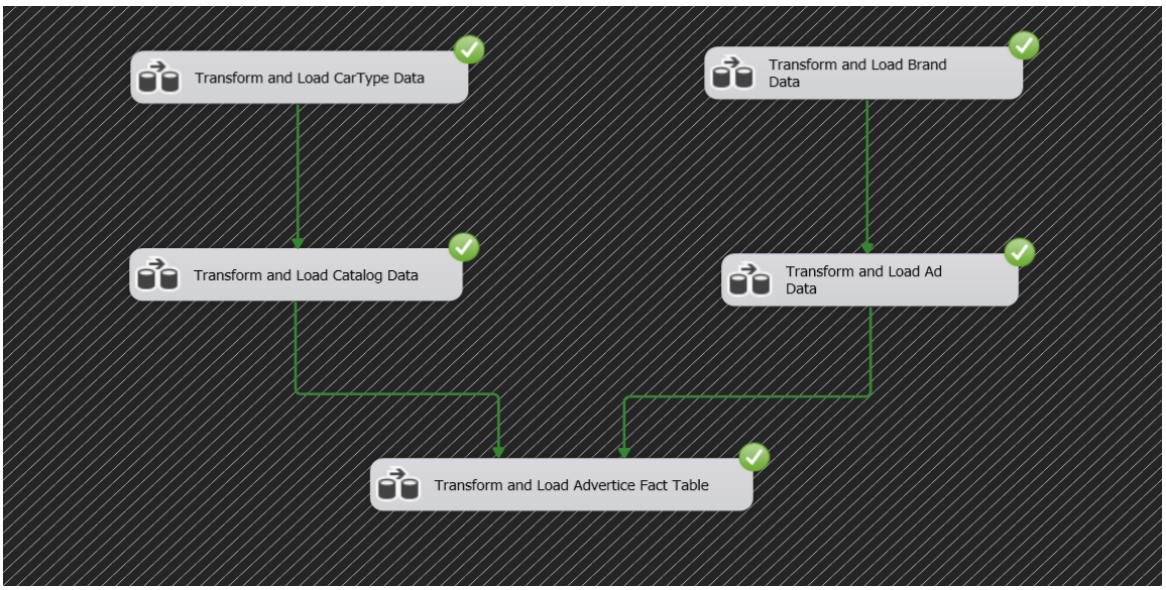
II. Data Profiling

Data Profiling provides the means of analyzing large amount of data using different kind of processes. In this step, null values, repeated values, and quality of the data is checked.

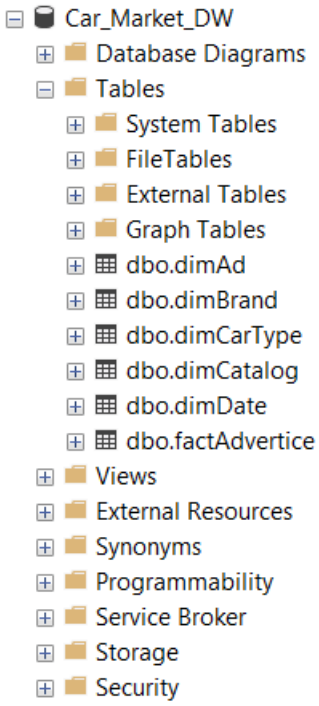
- Every staging table is profiled and saved in a selected location.
- After the Staging step doing this task shows the things what the developer must consider about the data which are stored in staging table and the developer is able to analyze and identify the issues with staging data by data profiling.

III. Data Transformation and Loading

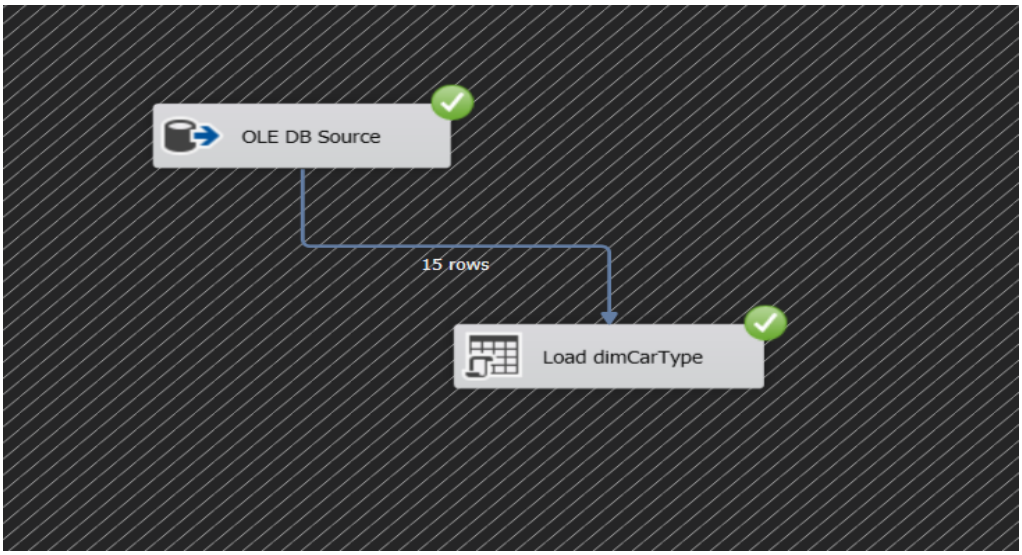
Data Transformation is developed according to the dimensional modeling designed above.



- In this step, the Dimension Tables created in Car_Market_DW are loaded with the data of relevant staging tables.

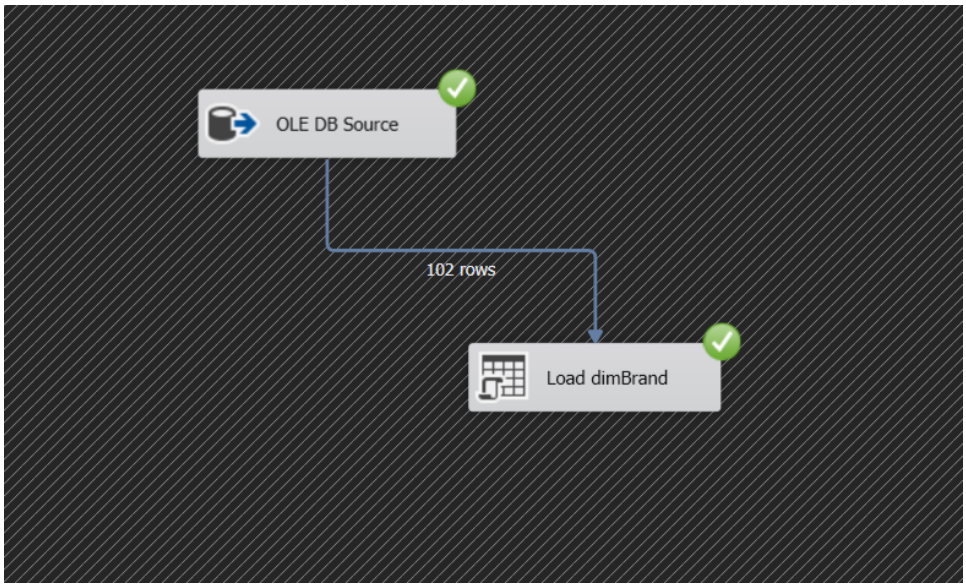


- Car Type data are loaded to dimCarType table from stgCarType
- UpdateDimCarType procedure is used to check whether the data is inserted or not.



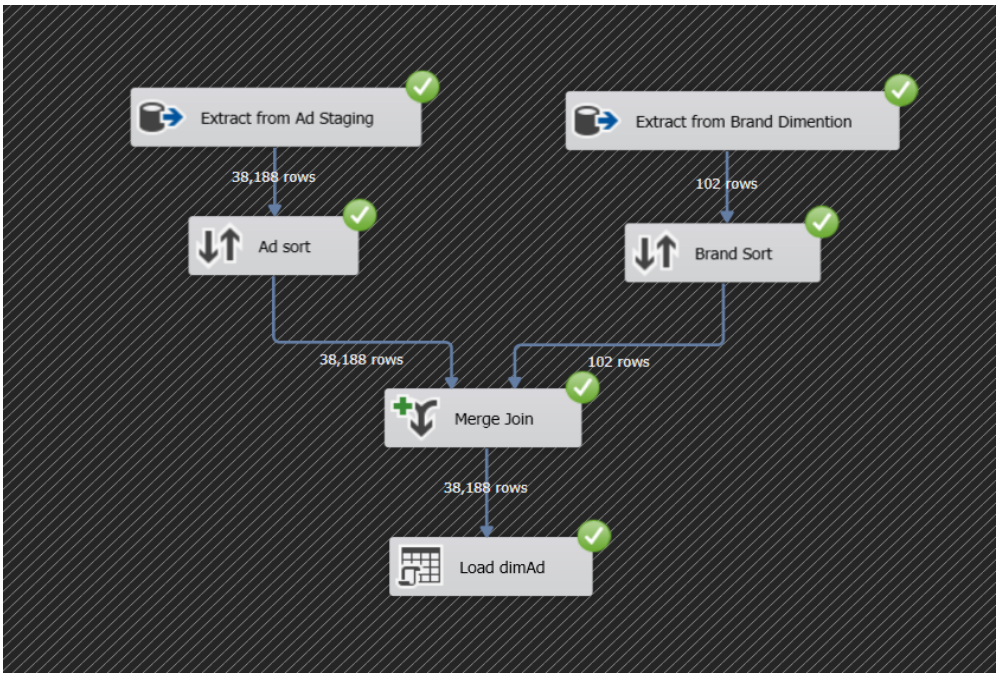

```
ALTER PROCEDURE [dbo].[UpdateDimCarType]
@car_type_id int,
@car_type_name nvarchar(50)
AS
BEGIN
if not exists (select car_type_SK
from dbo.dimCarType
where alternate_car_type_id = @car_type_id)
BEGIN
insert into dbo.dimCarType
(alternate_car_type_id, car_type_name, insert_date, modified_date)
values
(@car_type_id, @car_type_name, GETDATE(), GETDATE())
END;
if exists (select car_type_SK
from dbo.dimCarType
where alternate_car_type_id = @car_type_id)
BEGIN
update dbo.dimCarType
set car_type_name = @car_type_name,
modified_date = GETDATE()
where alternate_car_type_id = @car_type_id
END;
END;
```

- Brand data are loaded to dimBrand table from stgBrand
- UpdateDimBrand procedure is used to check whether the data is inserted or not.




```
ALTER PROCEDURE [dbo].[UpdateDimBrand]
@brand_id int,
@brand_name nvarchar(50)
AS
BEGIN
if not exists (select brand_SK
from dbo.dimBrand
where alternate_brand_id = @brand_id)
BEGIN
insert into dbo.dimBrand
(alternate_brand_id, brand_name, insert_date, modified_date)
values
(@brand_id, @brand_name, GETDATE(), GETDATE())
END;
if exists (select brand_SK
from dbo.dimBrand
where alternate_brand_id = @brand_id)
BEGIN
update dbo.dimBrand
set brand_name = @brand_name,
modified_date = GETDATE()
where alternate_brand_id = @brand_id
END;
END;
```

- Advertisement data are loaded to dimAd table from stgAd
- UpdateDimAd procedure is used to check whether the data is inserted or not.




```
ALTER PROCEDURE [dbo].[UpdateDimAd]
@ad_id int,
@proseller nchar(10),
@postalcode int,
@production datetime,
@mileage int,
@shifter nchar(10),
@color int,
@sales_date datetime,
@is_sold nchar(10),
@brand_key int
AS
BEGIN
if not exists (select ad_SK from dbo.dimAd where alternate_ad_id = @ad_id)
BEGIN
insert into dbo.dimAd (alternate_ad_id, proseller, postalcode, production, mileage, shifter,
 color, sales_date, is_sold,brand_key , insert_date, modified_date)

values(@ad_id, @proseller, @postalcode, @production, @mileage, @shifter,
 @color, @sales_date, @is_sold, @brand_key, GETDATE(), GETDATE())
END;
if exists (select ad_SK from dbo.dimAd where alternate_ad_id = @ad_id)
BEGIN
update dbo.dimAd
set proseller = @proseller, postalcode = @postalcode, production = @production, mileage = @mileage, shifter = @shifte
 color = @color, sales_date = @sales_date, is_sold = @is_sold,
 brand_key = @brand_key, modified_date = GETDATE()
where alternate_ad_id = @ad_id
END;
END;
```

Loading Slowly Changing Dimension

- dimCatalog is the slowly changing dimension in this dimensional modeling.
- In order to load data to dimension table, the slowly changing dimensions (historical) have two specific columns as start_date & end_date to ensure that the data is valid at the moment.
- Slowly changing dimension wizard let the developer to select the dimension table, business keys of the dimension and what would be the slowly changing attributes.

- As mentioned earlier under assumptions, Catalog details were considered as slowly changing details.

Changing attributes

- msrp

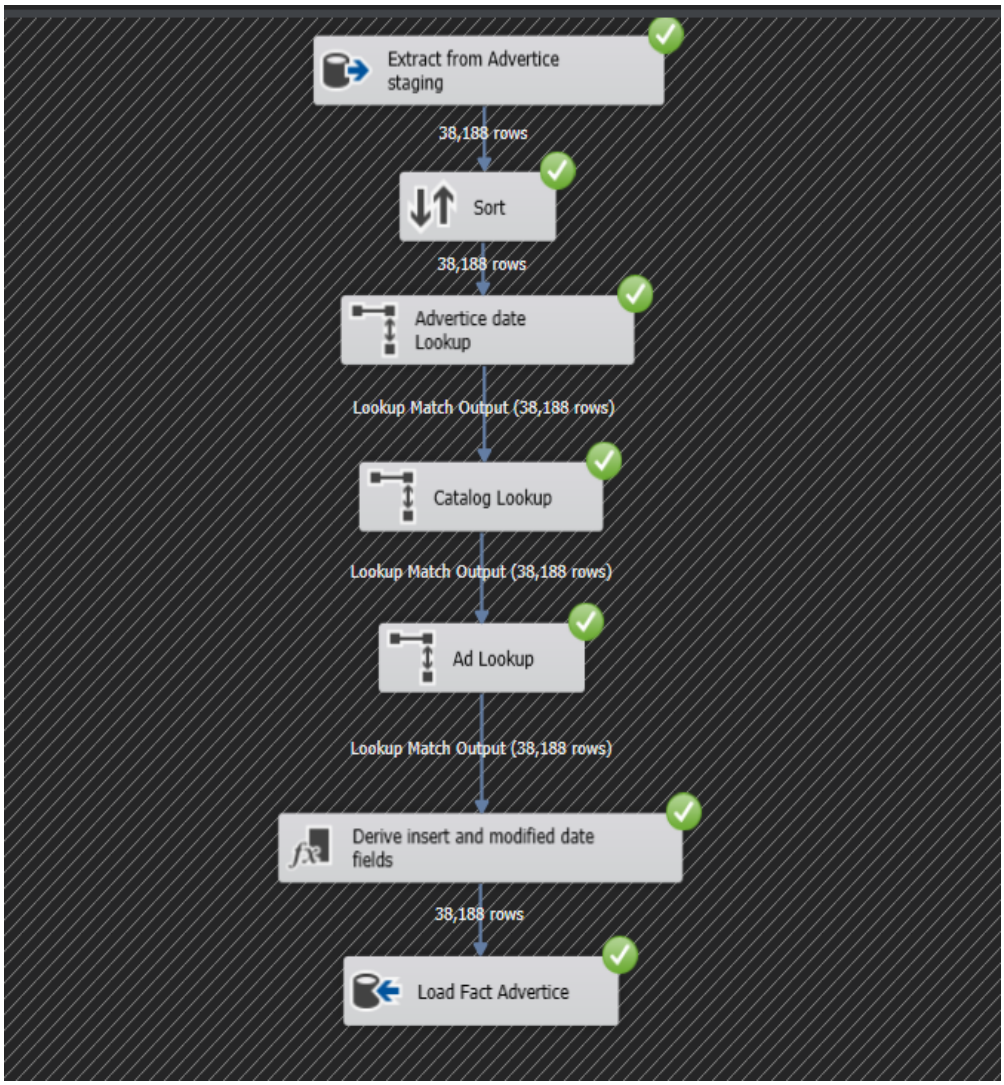
Historical attributes

- weight
- fuel_tank
- fuel
- boot_capacity
- cylinders
- ccm
- consump_city
- consump_highway
- consump_mixed
- top_speed
- acceleration
- torque
- power
- doorsnumber
- person_capacity

6. ETL Development – Accumulating fact table

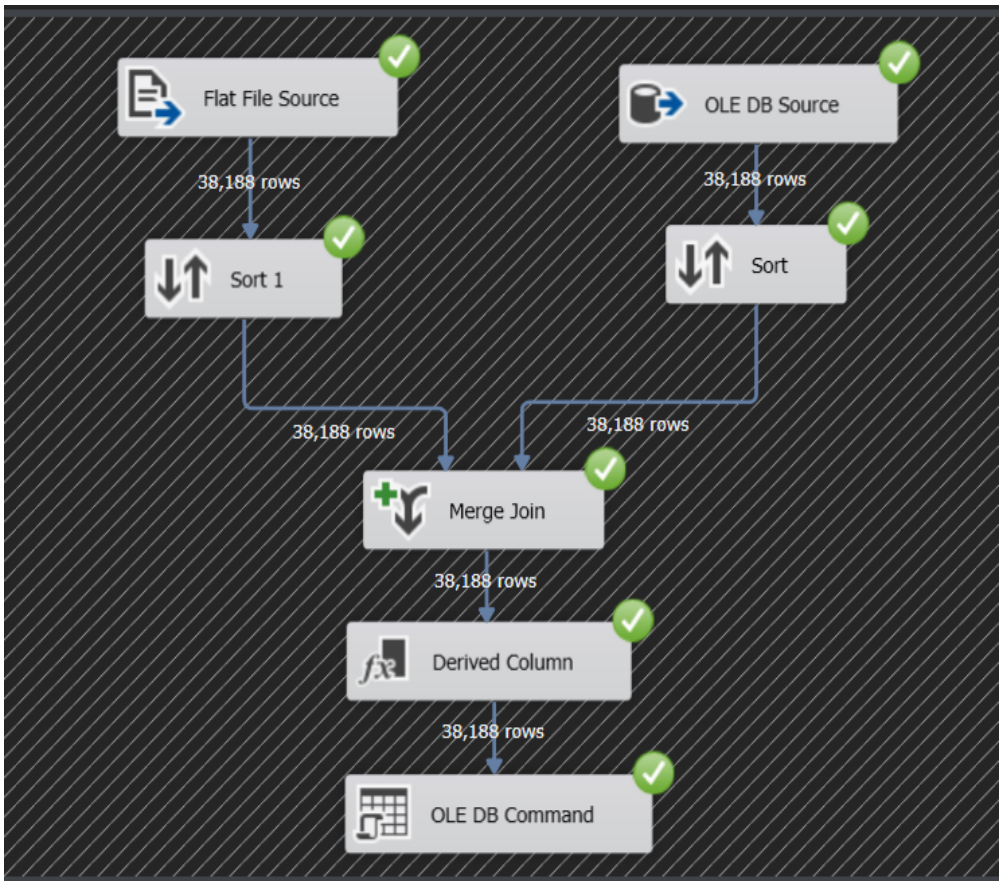
- The final step of Transformation & Loading is load data to the accumulative fact table. According to the dimensional model, stgAdvertice table is used to insert values into factAdvertice table.
- factAdvertice table has one date key which is related to Date Dimension as advertice_key.
- Then it has three separate columns as accm_txn_create_time, accm_txn_complete_time and txn_process_time_hours to update the corresponding accumulative complete times of each record.
- After loading to all the dimensions, data was loaded to the fact table as the last step. The steps are followed as below.

1. Data extracted from the Advertice staging table (stgAdvertice).
2. Sort operation is for sort date according to advertice_id.
3. Join operation is done for the advertice_date using look up.
4. Join operation is done for the Catalog using look up (dimCatalog).
5. Join operation is done for the Ad using look up (dimAd).
6. Insert date and modified date were derived.
7. Fact Data loaded to the factAdvertice table.



- Then separate csv file is there (CompleteTime.csv) to update the corresponding accumulative complete times of each record which is done by a separate package. For that the following SQL statement has been used.

  UPDATE [Car_Market_DW].[dbo].[factAdvertice] SET [txn_process_time_hours] = ? , [accm_txn_complete_time] = ?
  WHERE advertice_id = ?



- Can view the accumulative fact table as the final step in SQL Server Management Studio.

	advertice_date_key	ad_key	catalog_key	advertice_id	ad_price	numpictures	insert_date	modified_date	accm_txn_create_time	accm_txn_complete_time	txn_process_time_hours
1	20130303	35873	10842	50246	530000	5	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-20 13:00:00.000	1900-04-25 00:00:00.0
2	20121213	21708	10842	50357	1290000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-30 20:00:00.000	1900-12-28 00:00:00.0
3	20080213	28406	10842	50468	580000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-21 16:00:00.000	1900-05-22 00:00:00.0
4	20090831	28776	10842	50579	1450000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-13 17:00:00.000	1901-11-26 00:00:00.0
5	20130325	17045	10842	50690	9990000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-04 07:00:00.000	1901-04-14 00:00:00.0
6	20100510	25029	10842	50801	120000	5	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-25 04:00:00.000	1900-08-14 00:00:00.0
7	20151126	27543	10842	50912	2290000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-16 01:00:00.000	1902-01-21 00:00:00.0
8	20080224	27540	10842	51023	145000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-09 20:00:00.000	1901-08-25 00:00:00.0
9	20150427	2354	10821	51134	799000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-28 03:00:00.000	1900-10-24 00:00:00.0
10	20081006	4995	10821	51245	4850000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-07 23:00:00.000	1901-07-11 00:00:00.0
11	20071229	37897	10821	51356	4900000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-04 04:00:00.000	1901-04-11 00:00:00.0
12	20070906	5059	10821	51467	1990000	2	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-13 08:00:00.000	1901-11-17 00:00:00.0
13	20130908	20716	10821	51578	79000	12	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-09 19:00:00.000	1901-08-24 00:00:00.0
14	20100509	25263	10821	51689	170000	4	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-09 05:00:00.000	1901-08-10 00:00:00.0
15	20080518	25236	10821	51800	545000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-12 17:00:00.000	1901-11-02 00:00:00.0
16	20161225	30772	10821	51911	2199998	12	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-12 14:00:00.000	1901-10-30 00:00:00.0
17	20080401	5089	10803	52022	3850000	12	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-13 21:00:00.000	1901-11-30 00:00:00.0
18	20130425	25314	10803	52133	190000	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-27 07:00:00.000	1900-10-04 00:00:00.0
19	20071003	5023	10803	52244	10500...	6	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-06-13 11:00:00.000	1901-11-20 00:00:00.0
20	20151107	28770	10803	52355	130000	5	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-15 17:54:40.487	2022-05-21 22:00:00.000	1900-05-28 00:00:00.0

- Fact details were added to the factAdvertice table.