SRI LANKA INSTITUTE OF INFORMATION TECHNOLOGY

IT3021 – DATA WAREHOUSE AND BUSINESS INTELLIGENCE
ASSIGNMENT – 01
2022

SUBMITTED BY:

IT19991986 – RAEESUL ISLAM S.Z.

Y3S2.WE.DS.03

Table of Contents

1. Data Selection and Introduction

1.1. Introduction to Data set

- Data Set Name : **Predicting Coupon Redemption**

The data set represents information about a collection of an anonymized Sales Transaction information from an Export Company to Predict Coupon Redemption, in 2012. The data set has been modified to develop a scenario that meets the requirement of the assignment.

Dataset contains 5 csv files with information about Customers, Train, Items, Campaign and Transaction. Modifications were done accordingly to the data set derived from the source This data set reflects combinations between customer transactions and promotion campaigns.

1.2. Features of Dataset

- **Train Data**: containing the coupons offered to the given customers under the 18 campaigns.
- **Customer Data**: Customer Data containing information for some customers.
- **Campaign Data**: Campaign information for each of the campaign.
- **Transaction Data**: Transaction data for all customers for duration of campaigns in the train data
- **Item Data**: Item information for each item sold by the retailer

1.3. Link to Access Dataset

https://www.kaggle.com/datasets/meghakanojia/predicting-coupon-redemption

2. Preparations of Data Sources

All the data sources are provided in csv format by the web site. In preparation of data sources, some changes have done for the source format (some columns were added, separated into another table) of the given files as converting into text files and importing csv files into a source database.

Ultimately, 2 main sources were created:

1. A database source : **PredictingCouponRedemptionSourceDB**
2. A text file to maintain customer details : **customer_data.txt**

A database named **PredictingCouponRedemptionSourceDB** was created in SQL and the below mentioned files were imported:

- campaign_data.csv
- train_data.csv
- transaction_data.csv
- category_data.csv
- item_data.csv
- district_data.txt

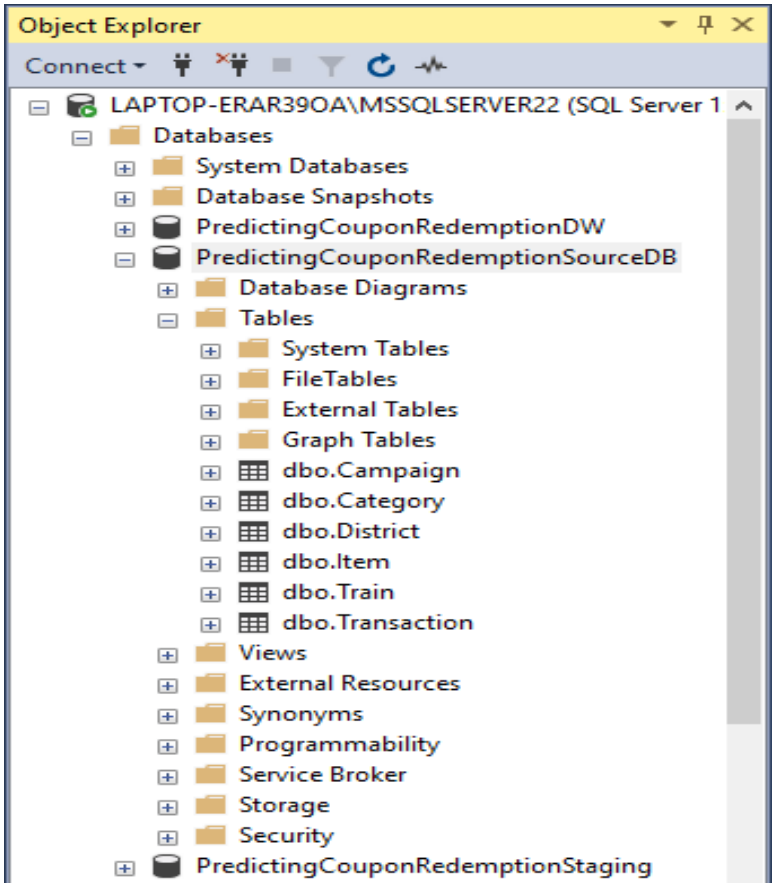


Figure 1. PredictingCouponRedemptionSourceDB

PredictingCouponRedemptionStaging database was created as a staging layer.

8.1. Staging Tables Creations

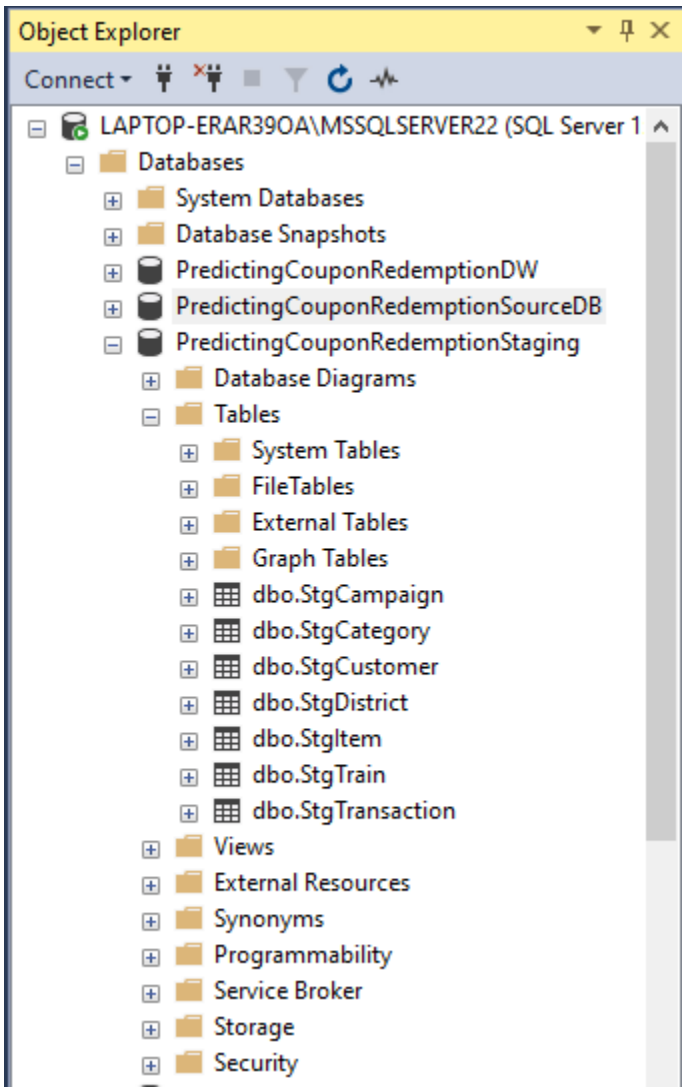


Figure 2. PredictingCouponRedemptionStaging

For data warehousing purposes a database named **PredictingCouponRedemptionDW** was created in SQL, including the dimensions and fact tables mentioned below.

- **DimCampaign**
- **DimTrain**
- **DimCategory**
- **DimItem**
- **DimCustomer**
- **DimDistrict**
- **DimDate**
- **FactTransaction**

8.2. Dimension Creations

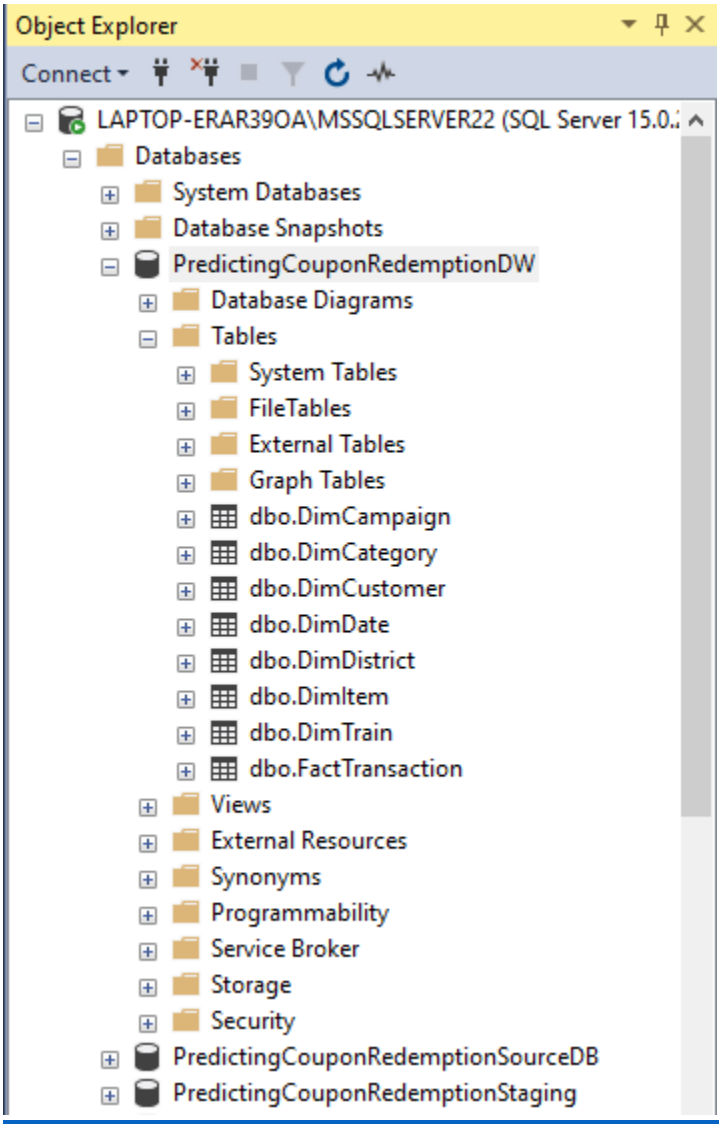


Figure 3. PredictingCouponRedemptionDW

3. ER Diagram Developed Using the Sources

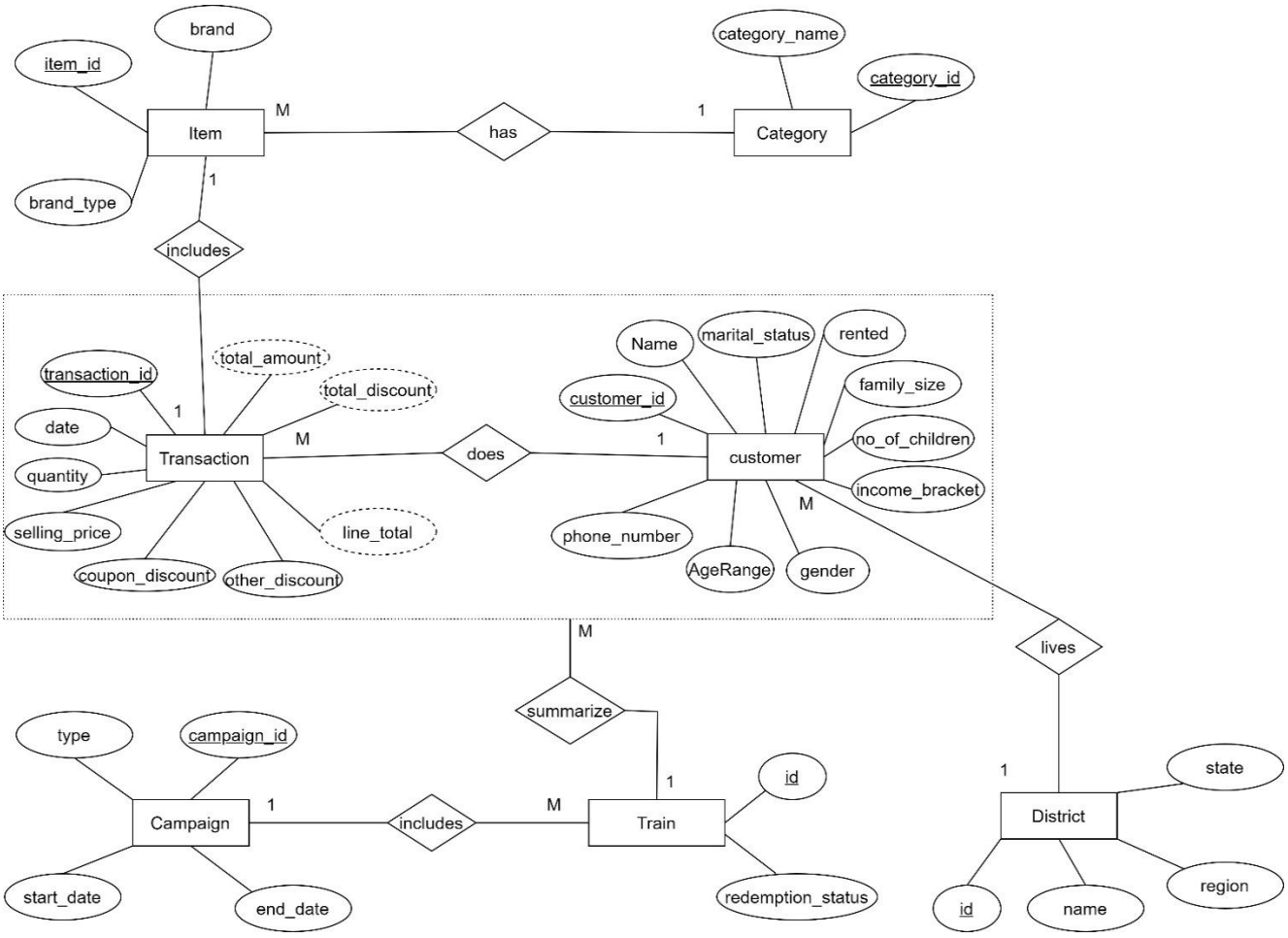


Figure 4. ER Diagram constructed using sources

- The above diagram shows the connection between the entities in the data set.
- **Assumptions:**
  - The particular transaction includes only a single item .
  - One summary report(train) summarizes many customer transactions.
  - There can be many campaign data sets in a single summary report.
  - One customer can have many transactions.

4. Solution Architecture

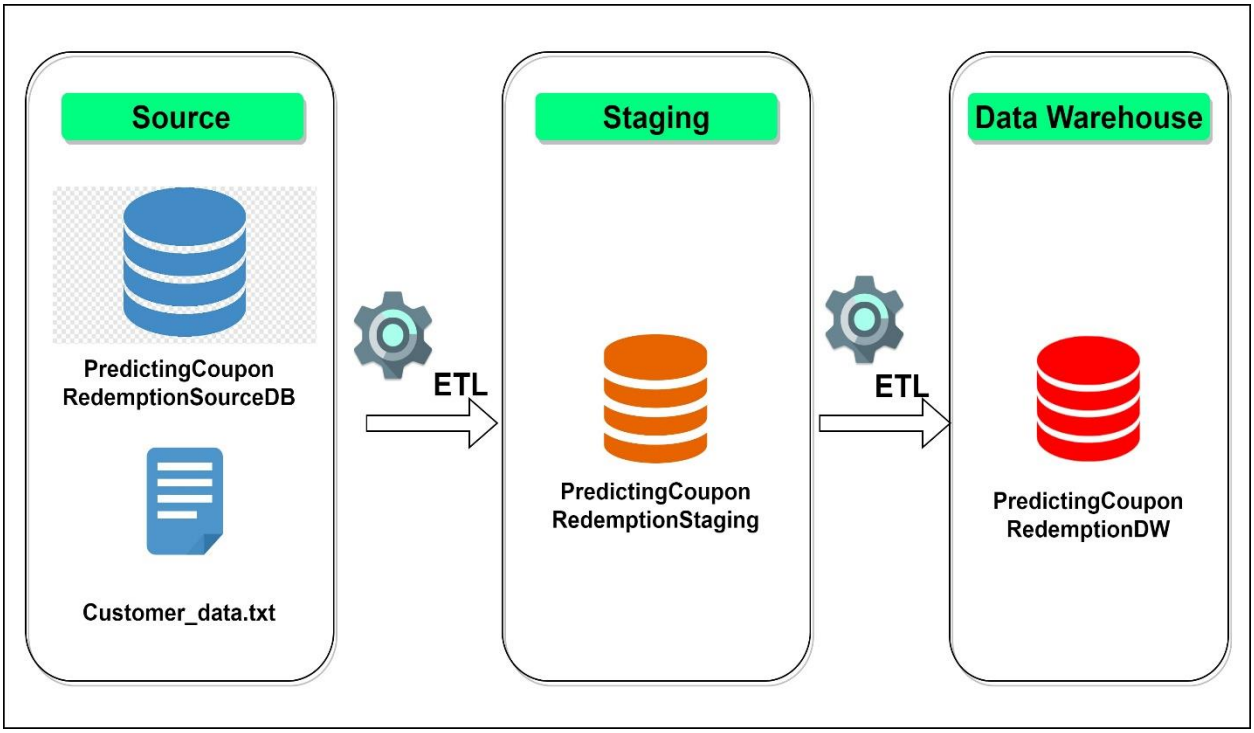


Figure 5. Solution Architecture

Data source:

- ✓ Several data sources can be available when implementing a data warehouse solution. Sources are simply the origin of location of the used data. A data source may be a database, flat file, live measurements from physical device, scraped web data, etc. Here, a database source namely '**PredictingCouponRedemptionSourceDB**' serves as the primary data source and a flat file source namely '**customer_data.txt**' serves as a secondary data source.

ETL:

- ✓ ETL is the abbreviation for the standard 'Extraction-Transformation-Loading.' It is the process of extracting data from one source, transform those data and finally load them to a destination. The extraction process followed here is a full extraction (Load all data in the source without filtering conditions). While performing the ETL process to load data to data warehouse, necessary steps like cleaning and aggregation were performed.

Staging Layer:

- ✓ This is an intermediate storage layer. This layer is added to prevent practical problems that could arise while transforming data to data warehouse. It is similar to the data source but contains all the data required for warehousing in a centralized location. A less amount of transformation is performed during the ETL process from source to staging.
- ✓ '**PredictingCouponRedemptionStaging**' is the database created as a staging layer in the scenario.

Data Warehouse:

- ✓ Data warehouse is a large collection of business data. Aggregated and transactional data are stored here for analytical purposes. It is a core component of business intelligence. A database named '**PredictingCouponRedemptionDW**' is created in SQL as the data warehouse layer.

5. Data Warehouse Design and Implementation

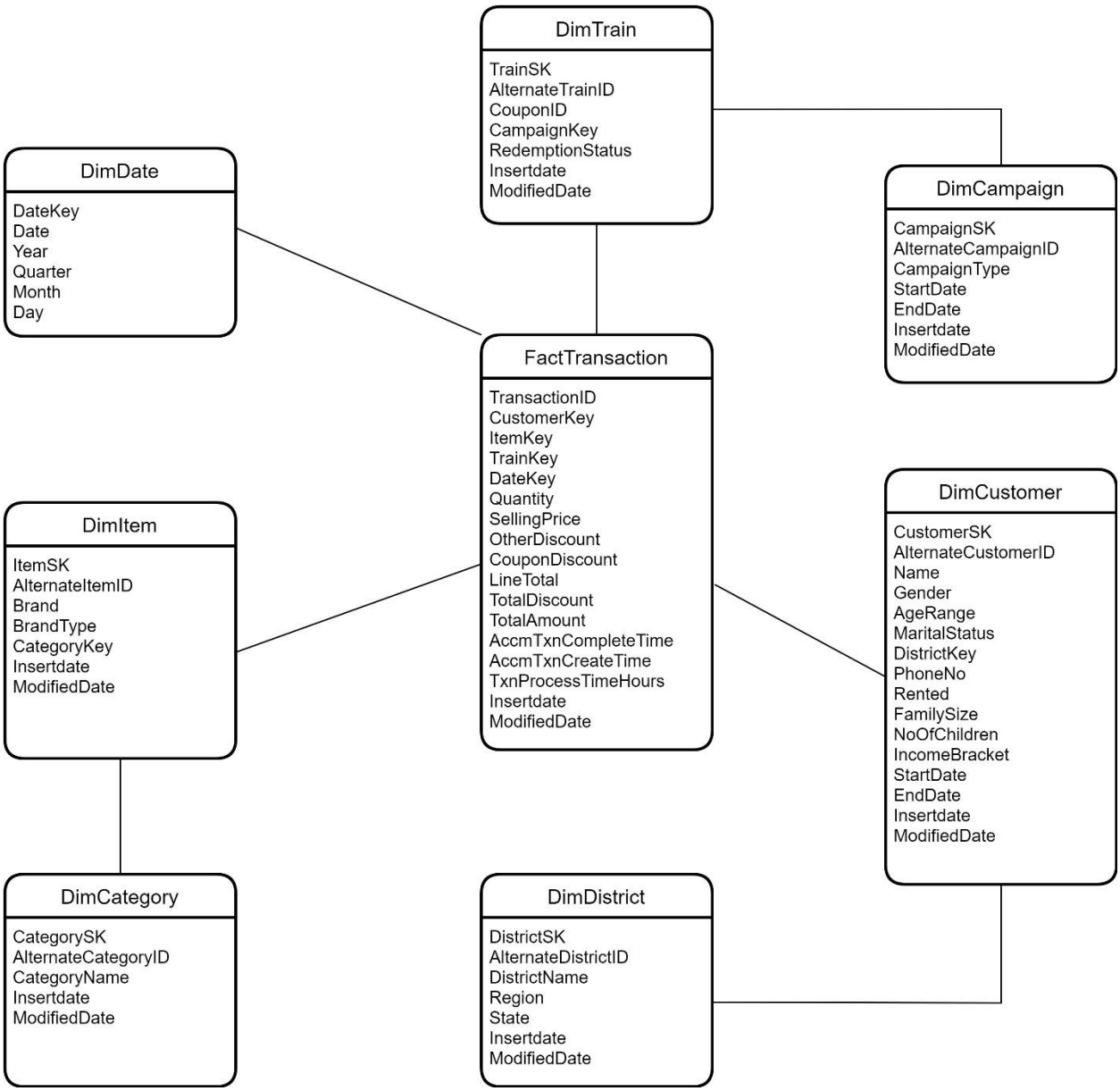


Figure 6. Snowflake Schema

The data warehouse design was implemented using the snowflake schema. It is an extension of star schema and consists of some dimensions that are normalized. According to the schema above, there are 7 dimensions and 1 fact table.

Hierarchies:

- ✓ **DimCategory** is applied as a hierarchical dimension of **DimItem** table.
- ✓ **DimCampaign** is applied as a hierarchical dimension of **DimTrain** table.
- ✓ **DimDistrict** is applied as a hierarchical dimension of **DimCustomer** table.

Calculation :

- ✓ Line Total is calculated in dbo.FactTransaction.LineTotal
  - ❖ ([SellingPrice] * [Quantity])
- ✓ Total Discount is calculated in dbo.FactTransaction.TotalDiscount
  - ❖ (-([OtherDiscount] + [CouponDiscount]))
- ✓ Total Amount is calculated in dbo.FactTransaction.TotalAmount
  - ❖ ([SellingPrice]*[Quantity]+([CouponDiscount]+[OtherDiscount]))

Assumption:

- ✓ **Customer** dimension is considered as a **Slowly Changing Dimension(SCD).**

6. ETL Development

6.1.Data Extraction from Source to Staging

As the initial step, the data from sources were extracted to a staging layer. These data were then transformed and loaded to the staging tables. The data flow task was used to perform this process.

Source table and staging tables are as below:

Source Table	Staging Table
Category	StgCategory
Item	StgItem
Campaign	StgCampaign
Train	StgTrain
District	StgDistrict
Transaction	StgTransaction
Customer_data.csv	StgCustomer

Control Flow:

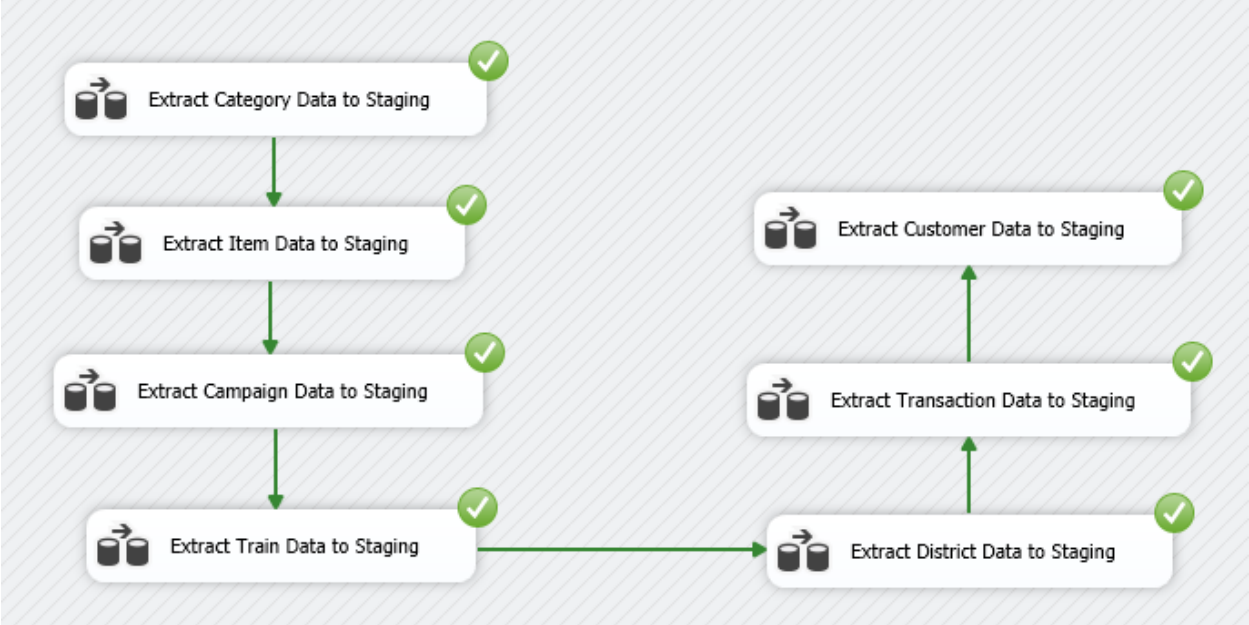


Figure 7. Control Flow Task for Staging

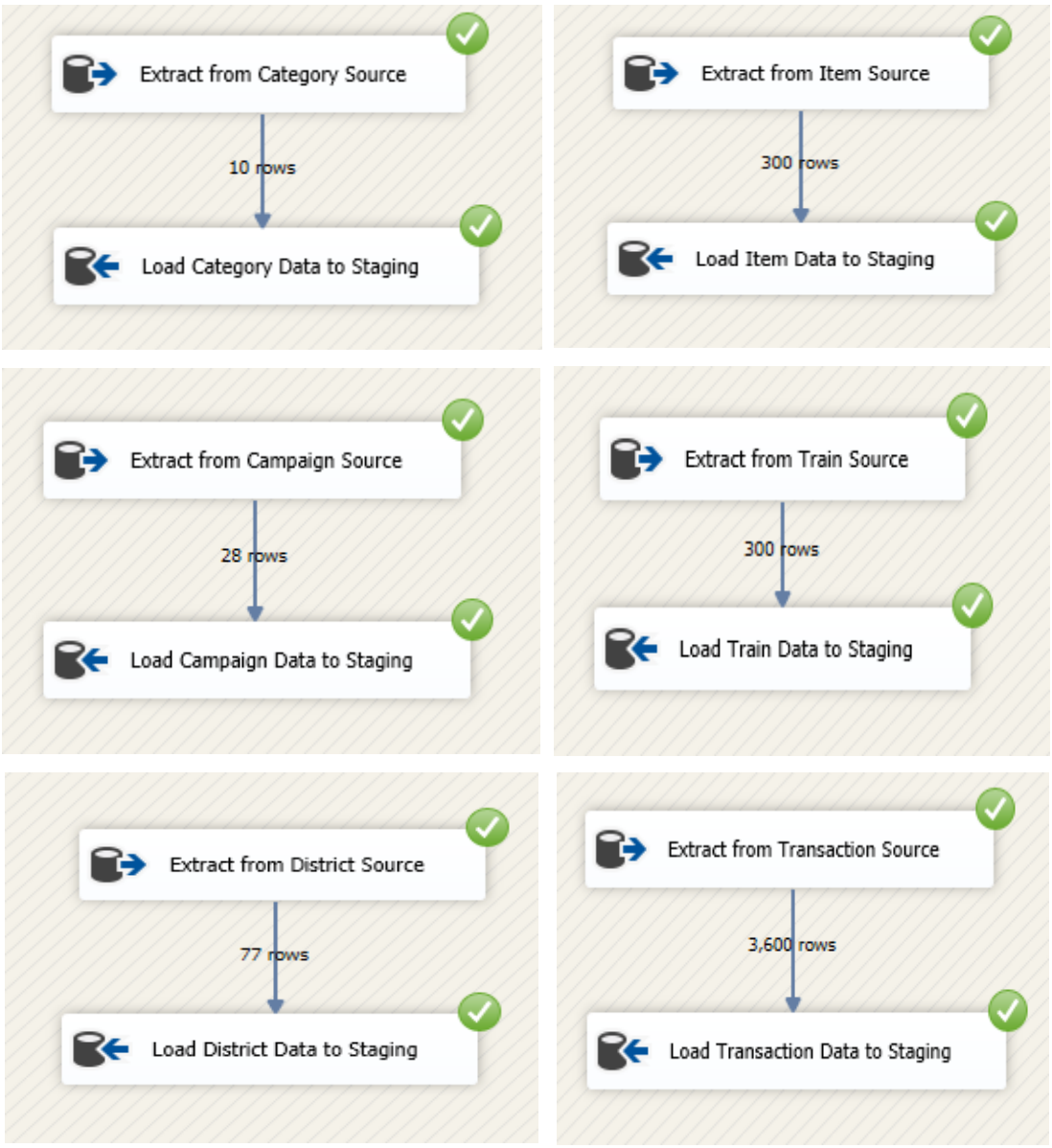


Figure 8. Staging from a Database Source to Database Destination

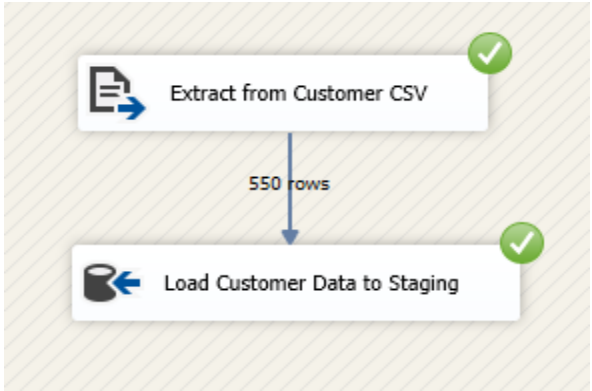
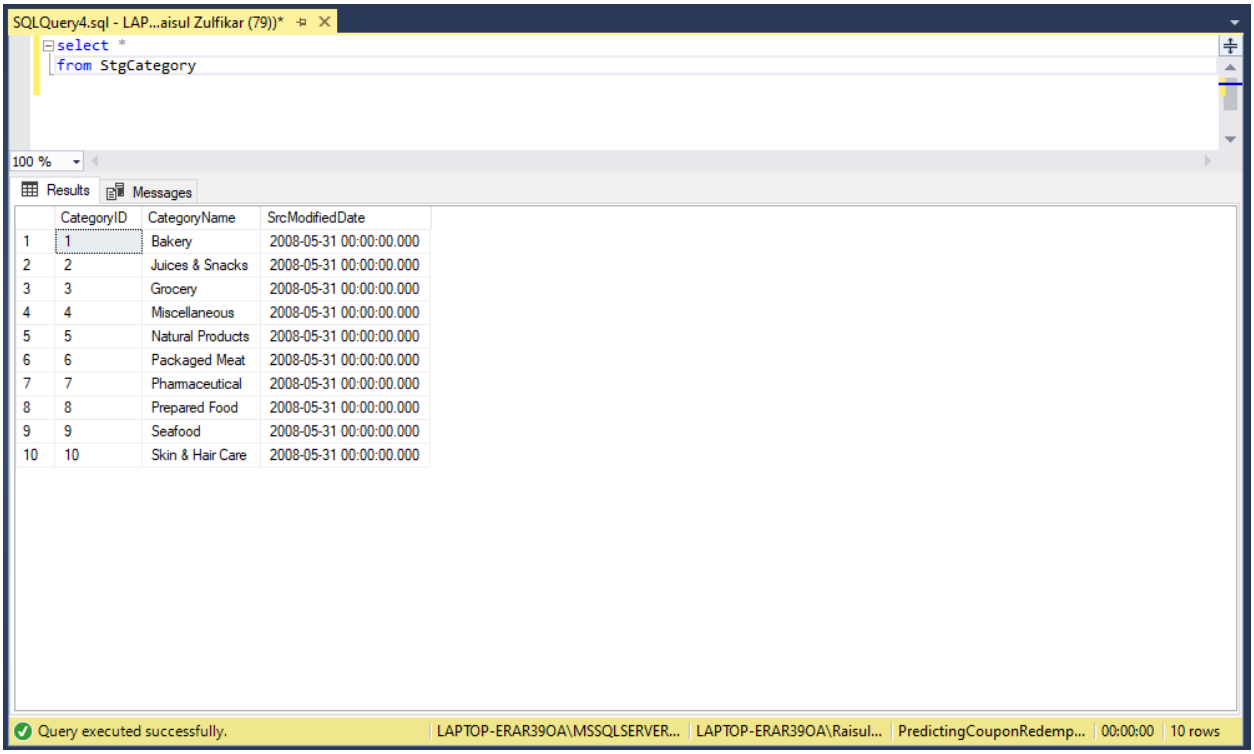


Figure 9. Staging from a Flat File Source to Database Destination

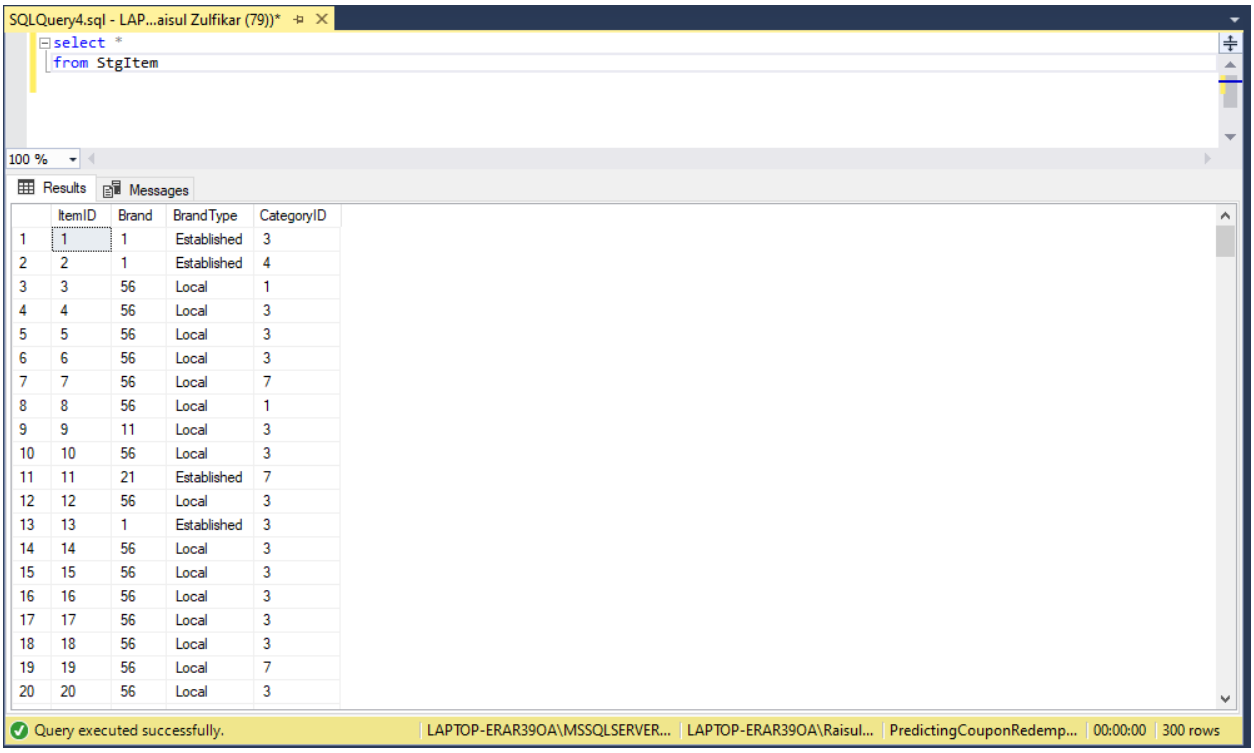
Test Data Loaded from Source to Staging

Test Scenario ID			1			
Test Case Description			Transform test data from source to staging			
Pre-Requisite			Test loaded from source to staging table			
ID	**Action**	**SQL Queries**	**Expected Output**	**Actual Output**	**Result**	**Refer**
1	Data passed from Category Source to StgCategory Staging	Select * From StgCategory	Display selected rows	Displayed selected rows	PASS	Figure 1.1
2	Data passed from Item Source to StgItem Staging	Select * From StgItem	Display selected rows	Displayed selected rows	PASS	Figure 1.2
3	Data passed from Campaign Source to StgCampaign Staging	Select * From StgCampaign	Display selected rows	Displayed selected rows	PASS	Figure 1.3
4	Data passed from Train Source to StgTrain Staging	Select * From StgTrain	Display selected rows	Displayed selected rows	PASS	Figure 1.4
5	Data passed from District Source to StgDistrict Staging	Select * From StgDistrict	Display selected rows	Displayed selected rows	PASS	Figure 1.5
6	Data passed from Customer Source to StgCustomer Staging	Select * From StgCustomer	Display selected rows	Displayed selected rows	PASS	Figure 1.6
7	Data passed from Transaction Source to StgTransaction Staging	Select * From StgTransaction	Display selected rows	Displayed selected rows	PASS	Figure 1.7



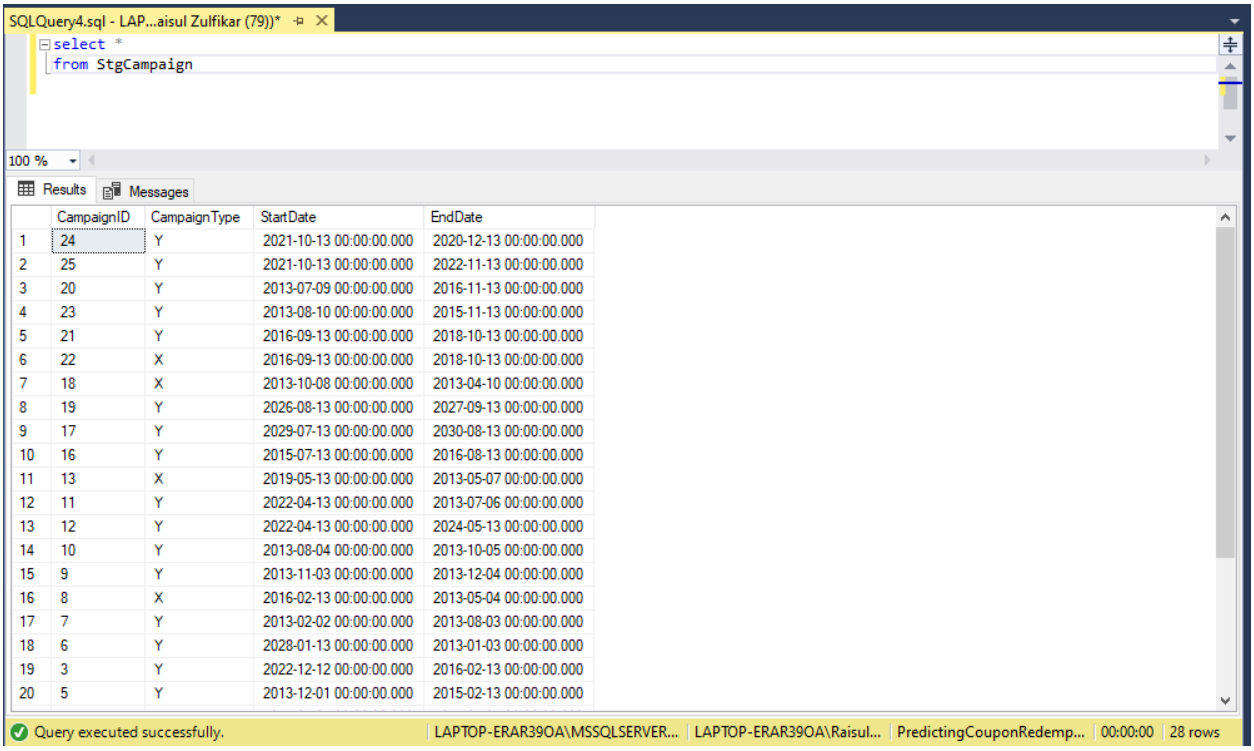
	CategoryID	CategoryName	SrcModifiedDate
1	1	Bakery	2008-05-31 00:00:00.000
2	2	Juices & Snacks	2008-05-31 00:00:00.000
3	3	Grocery	2008-05-31 00:00:00.000
4	4	Miscellaneous	2008-05-31 00:00:00.000
5	5	Natural Products	2008-05-31 00:00:00.000
6	6	Packaged Meat	2008-05-31 00:00:00.000
7	7	Pharmaceutical	2008-05-31 00:00:00.000
8	8	Prepared Food	2008-05-31 00:00:00.000
9	9	Seafood	2008-05-31 00:00:00.000
10	10	Skin & Hair Care	2008-05-31 00:00:00.000

Figure 1. 1. StgCategory



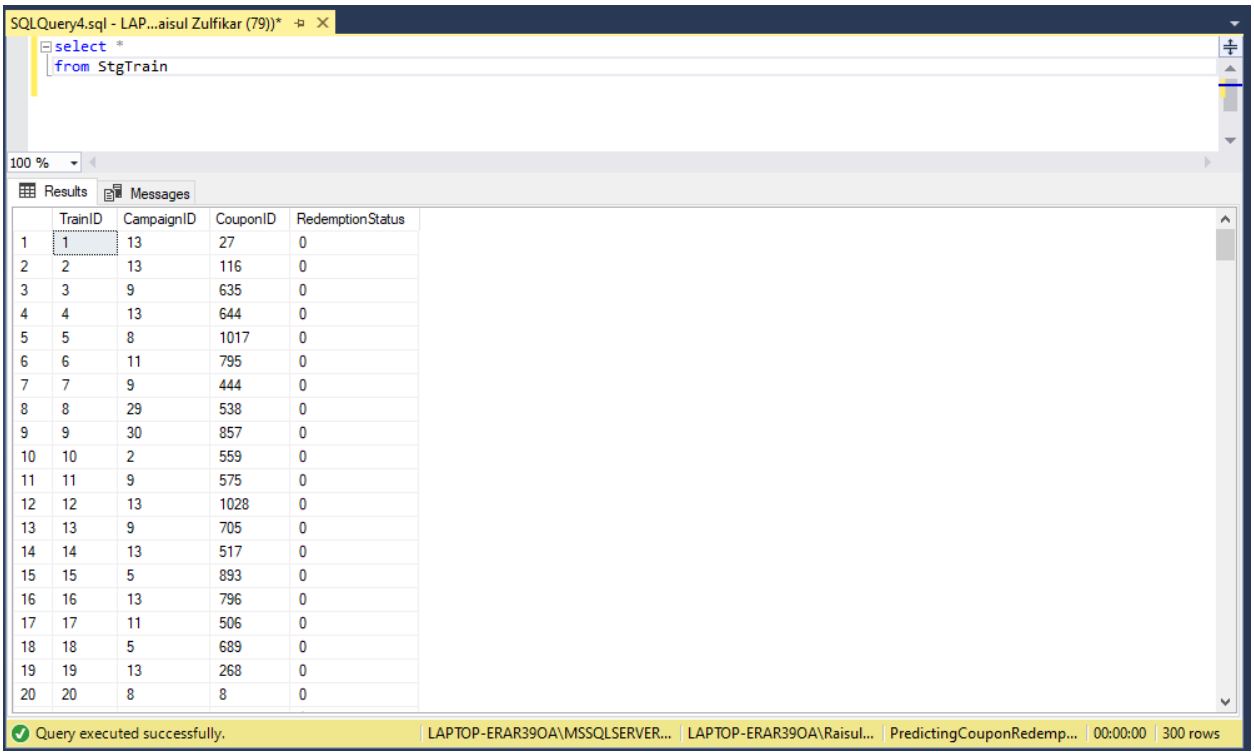
	ItemID	Brand	BrandType	CategoryID
1	1	1	Established	3
2	2	1	Established	4
3	3	56	Local	1
4	4	56	Local	3
5	5	56	Local	3
6	6	56	Local	3
7	7	56	Local	7
8	8	56	Local	1
9	9	11	Local	3
10	10	56	Local	3
11	11	21	Established	7
12	12	56	Local	3
13	13	1	Established	3
14	14	56	Local	3
15	15	56	Local	3
16	16	56	Local	3
17	17	56	Local	3
18	18	56	Local	3
19	19	56	Local	7
20	20	56	Local	3

Figure 1. 2. StgItem



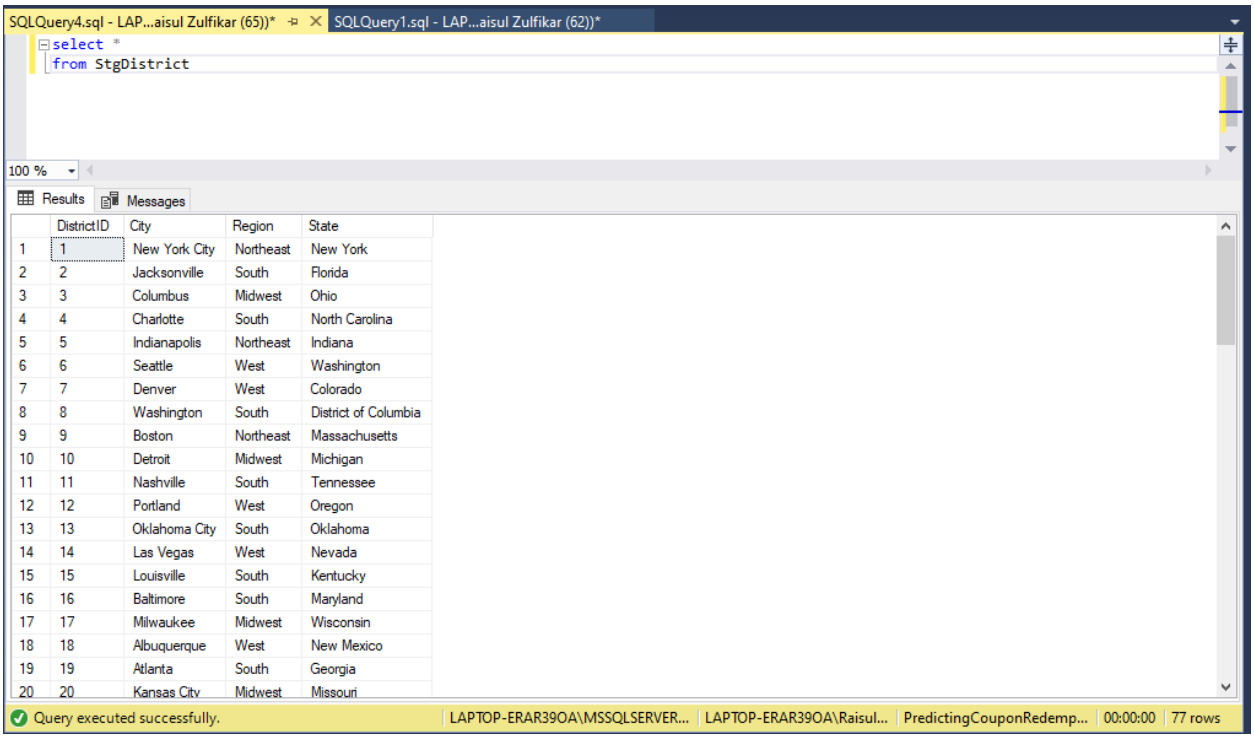
	CampaignID	CampaignType	StartDate	EndDate
1	24	Y	2021-10-13 00:00:00.000	2020-12-13 00:00:00.000
2	25	Y	2021-10-13 00:00:00.000	2022-11-13 00:00:00.000
3	20	Y	2013-07-09 00:00:00.000	2016-11-13 00:00:00.000
4	23	Y	2013-08-10 00:00:00.000	2015-11-13 00:00:00.000
5	21	Y	2016-09-13 00:00:00.000	2018-10-13 00:00:00.000
6	22	X	2016-09-13 00:00:00.000	2018-10-13 00:00:00.000
7	18	X	2013-10-08 00:00:00.000	2013-04-10 00:00:00.000
8	19	Y	2026-08-13 00:00:00.000	2027-09-13 00:00:00.000
9	17	Y	2029-07-13 00:00:00.000	2030-08-13 00:00:00.000
10	16	Y	2015-07-13 00:00:00.000	2016-08-13 00:00:00.000
11	13	X	2019-05-13 00:00:00.000	2013-05-07 00:00:00.000
12	11	Y	2022-04-13 00:00:00.000	2013-07-06 00:00:00.000
13	12	Y	2022-04-13 00:00:00.000	2024-05-13 00:00:00.000
14	10	Y	2013-08-04 00:00:00.000	2013-10-05 00:00:00.000
15	9	Y	2013-11-03 00:00:00.000	2013-12-04 00:00:00.000
16	8	X	2016-02-13 00:00:00.000	2013-05-04 00:00:00.000
17	7	Y	2013-02-02 00:00:00.000	2013-08-03 00:00:00.000
18	6	Y	2028-01-13 00:00:00.000	2013-01-03 00:00:00.000
19	3	Y	2022-12-12 00:00:00.000	2016-02-13 00:00:00.000
20	5	Y	2013-12-01 00:00:00.000	2015-02-13 00:00:00.000

Figure 1. 3. StgCampaign



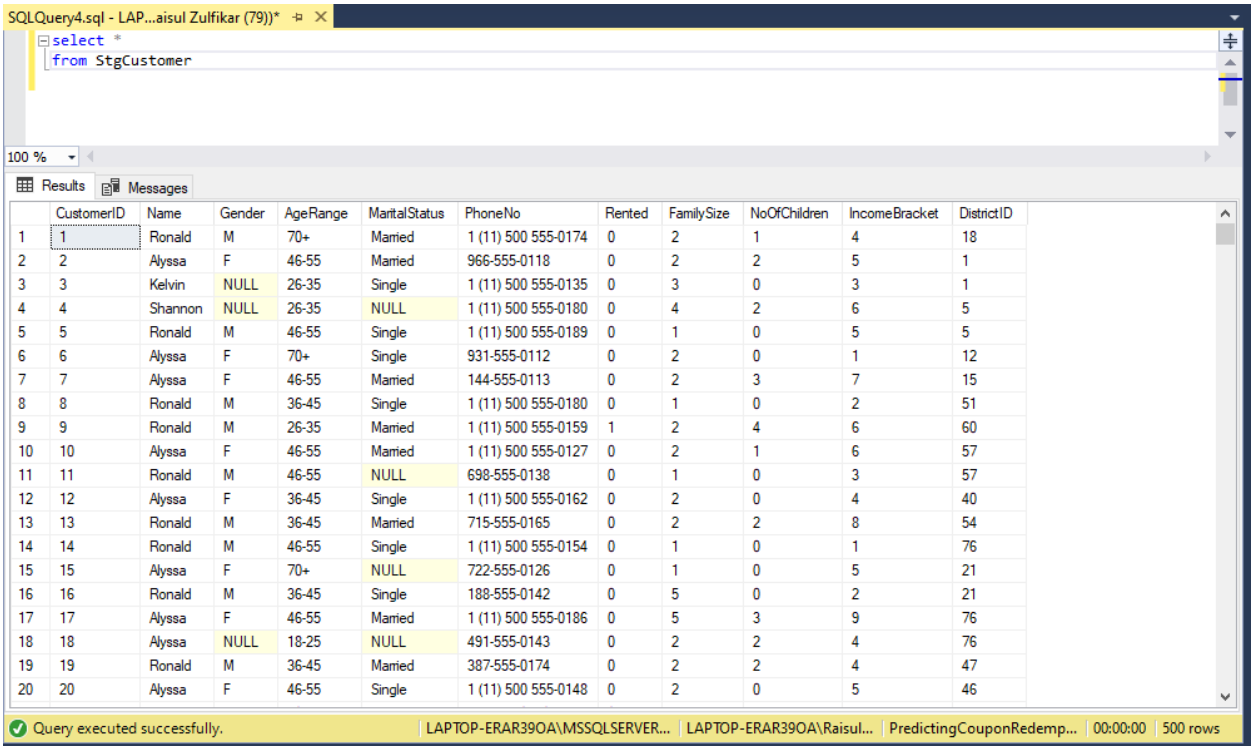
	TrainID	CampaignID	CouponID	RedemptionStatus
1	1	13	27	0
2	2	13	116	0
3	3	9	635	0
4	4	13	644	0
5	5	8	1017	0
6	6	11	795	0
7	7	9	444	0
8	8	29	538	0
9	9	30	857	0
10	10	2	559	0
11	11	9	575	0
12	12	13	1028	0
13	13	9	705	0
14	14	13	517	0
15	15	5	893	0
16	16	13	796	0
17	17	11	506	0
18	18	5	689	0
19	19	13	268	0
20	20	8	8	0

Figure 1. 4. StgTrain



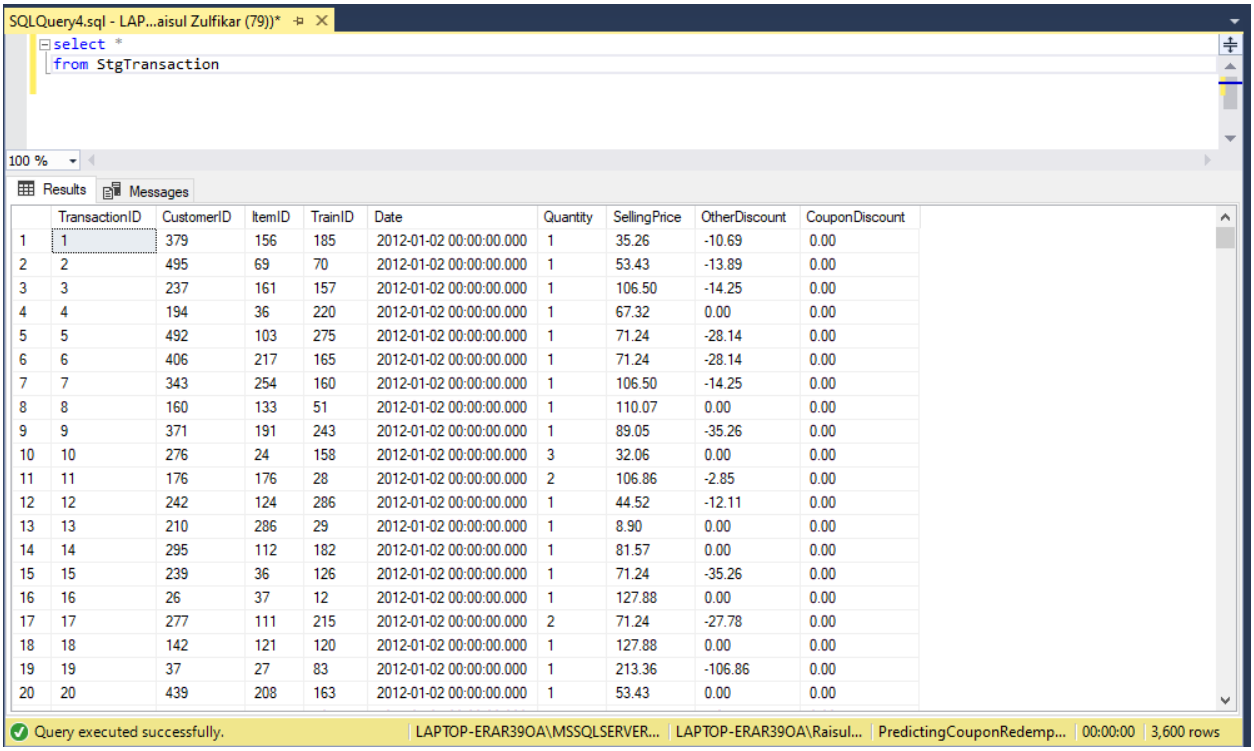
	DistrictID	City	Region	State
1	1	New York City	Northeast	New York
2	2	Jacksonville	South	Florida
3	3	Columbus	Midwest	Ohio
4	4	Charlotte	South	North Carolina
5	5	Indianapolis	Northeast	Indiana
6	6	Seattle	West	Washington
7	7	Denver	West	Colorado
8	8	Washington	South	District of Columbia
9	9	Boston	Northeast	Massachusetts
10	10	Detroit	Midwest	Michigan
11	11	Nashville	South	Tennessee
12	12	Portland	West	Oregon
13	13	Oklahoma City	South	Oklahoma
14	14	Las Vegas	West	Nevada
15	15	Louisville	South	Kentucky
16	16	Baltimore	South	Maryland
17	17	Milwaukee	Midwest	Wisconsin
18	18	Albuquerque	West	New Mexico
19	19	Atlanta	South	Georgia
20	20	Kansas City	Midwest	Missouri

Figure 1. 5. StgDistrict



	CustomerID	Name	Gender	AgeRange	MaritalStatus	PhoneNo	Rented	FamilySize	NoOfChildren	IncomeBracket	DistrictID
1	1	Ronald	M	70+	Married	1 (11) 500 555-0174	0	2	1	4	18
2	2	Alyssa	F	46-55	Married	966-555-0118	0	2	2	5	1
3	3	Kelvin	NULL	26-35	Single	1 (11) 500 555-0135	0	3	0	3	1
4	4	Shannon	NULL	26-35	NULL	1 (11) 500 555-0180	0	4	2	6	5
5	5	Ronald	M	46-55	Single	1 (11) 500 555-0189	0	1	0	5	5
6	6	Alyssa	F	70+	Single	931-555-0112	0	2	0	1	12
7	7	Alyssa	F	46-55	Married	144-555-0113	0	2	3	7	15
8	8	Ronald	M	36-45	Single	1 (11) 500 555-0180	0	1	0	2	51
9	9	Ronald	M	26-35	Married	1 (11) 500 555-0159	1	2	4	6	60
10	10	Alyssa	F	46-55	Married	1 (11) 500 555-0127	0	2	1	6	57
11	11	Ronald	M	46-55	NULL	698-555-0138	0	1	0	3	57
12	12	Alyssa	F	36-45	Single	1 (11) 500 555-0162	0	2	0	4	40
13	13	Ronald	M	36-45	Married	715-555-0165	0	2	2	8	54
14	14	Ronald	M	46-55	Single	1 (11) 500 555-0154	0	1	0	1	76
15	15	Alyssa	F	70+	NULL	722-555-0126	0	1	0	5	21
16	16	Ronald	M	36-45	Single	188-555-0142	0	5	0	2	21
17	17	Alyssa	F	46-55	Married	1 (11) 500 555-0186	0	5	3	9	76
18	18	Alyssa	NULL	18-25	NULL	491-555-0143	0	2	2	4	76
19	19	Ronald	M	36-45	Married	387-555-0174	0	2	2	4	47
20	20	Alyssa	F	46-55	Single	1 (11) 500 555-0148	0	2	0	5	46

Figure 1. 6. StgCustomer



	TransactionID	CustomerID	ItemID	TrainID	Date	Quantity	SellingPrice	OtherDiscount	CouponDiscount
1	1	379	156	185	2012-01-02 00:00:00.000	1	35.26	-10.69	0.00
2	2	495	69	70	2012-01-02 00:00:00.000	1	53.43	-13.89	0.00
3	3	237	161	157	2012-01-02 00:00:00.000	1	106.50	-14.25	0.00
4	4	194	36	220	2012-01-02 00:00:00.000	1	67.32	0.00	0.00
5	5	492	103	275	2012-01-02 00:00:00.000	1	71.24	-28.14	0.00
6	6	406	217	165	2012-01-02 00:00:00.000	1	71.24	-28.14	0.00
7	7	343	254	160	2012-01-02 00:00:00.000	1	106.50	-14.25	0.00
8	8	160	133	51	2012-01-02 00:00:00.000	1	110.07	0.00	0.00
9	9	371	191	243	2012-01-02 00:00:00.000	1	89.05	-35.26	0.00
10	10	276	24	158	2012-01-02 00:00:00.000	3	32.06	0.00	0.00
11	11	176	176	28	2012-01-02 00:00:00.000	2	106.86	-2.85	0.00
12	12	242	124	286	2012-01-02 00:00:00.000	1	44.52	-12.11	0.00
13	13	210	286	29	2012-01-02 00:00:00.000	1	8.90	0.00	0.00
14	14	295	112	182	2012-01-02 00:00:00.000	1	81.57	0.00	0.00
15	15	239	36	126	2012-01-02 00:00:00.000	1	71.24	-35.26	0.00
16	16	26	37	12	2012-01-02 00:00:00.000	1	127.88	0.00	0.00
17	17	277	111	215	2012-01-02 00:00:00.000	2	71.24	-27.78	0.00
18	18	142	121	120	2012-01-02 00:00:00.000	1	127.88	0.00	0.00
19	19	37	27	83	2012-01-02 00:00:00.000	1	213.36	-106.86	0.00
20	20	439	208	163	2012-01-02 00:00:00.000	1	53.43	0.00	0.00

Figure 1. 7. StgTransaction

6.2.Data Profiling

Data profiling is the process of reviewing data to understand the structure, content and inter relationships. It uncovers the issues related to data quality that can be corrected in ETL process.

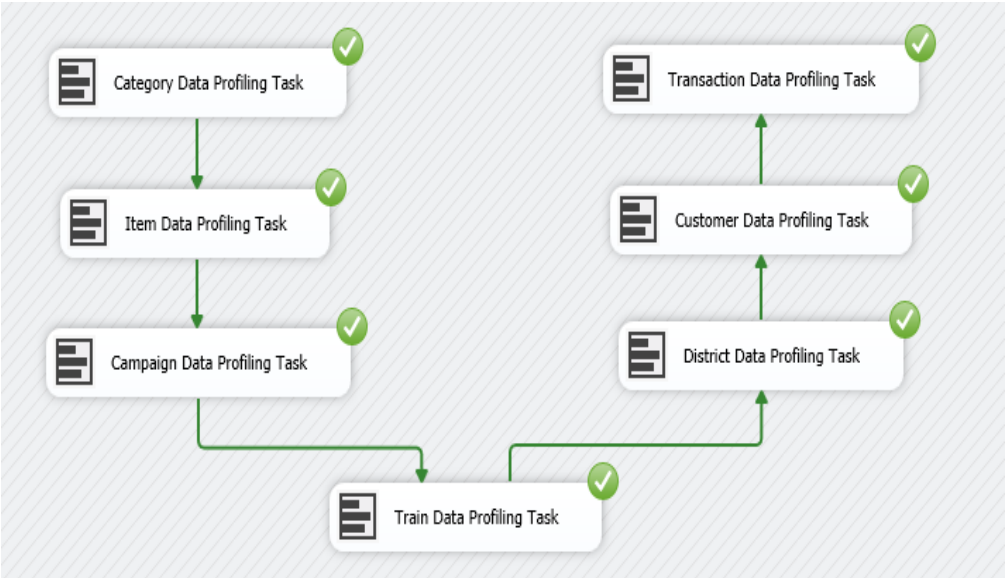


Figure 10. Data Profiling Task Flow

6.3.Transformation and Loading to Data Warehouse

When loading data from staged layer to Data Warehouse, the order of execution is very important. The reason for this is that the dimensions and facts contain dependencies with each other:

The order of execution is shown below in the control flow task of ETL:

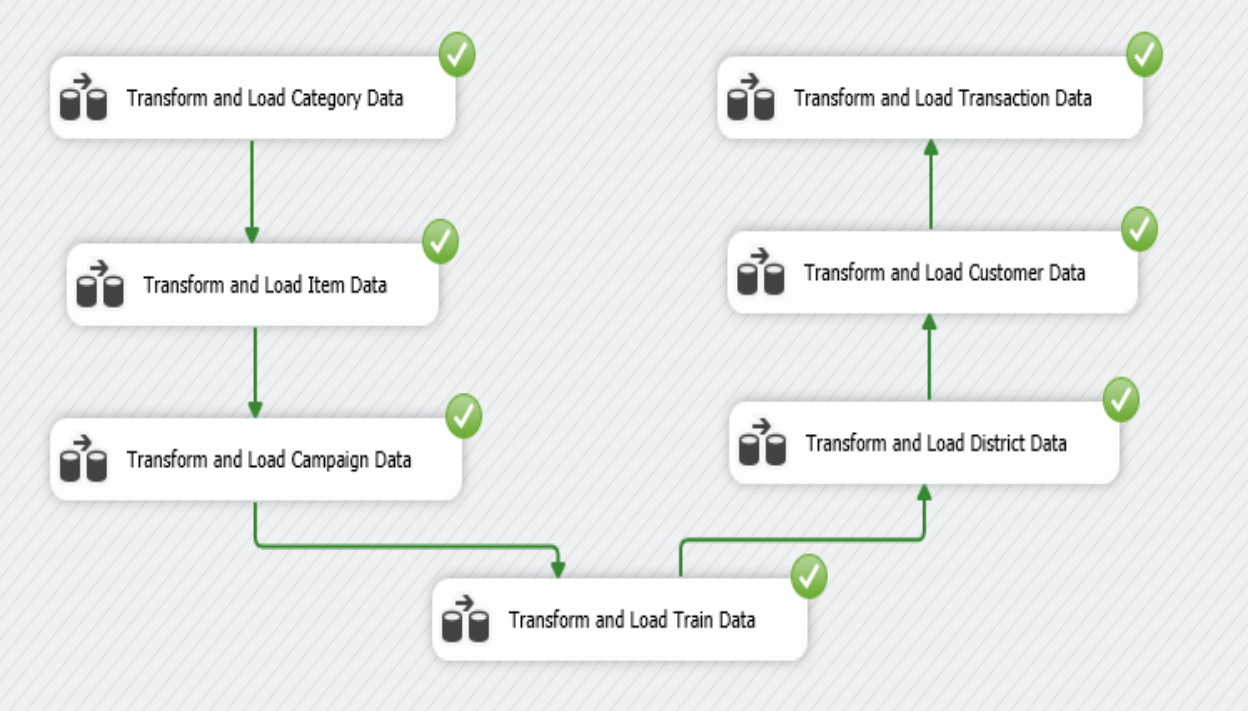


Figure 11. Control Flow Task of Data Warehouse Transformation and Loading

1. **Loading Category Data to DimCategory**

   The **Category** dimension has no dependencies with any other dimensions; therefore, it is loaded first.

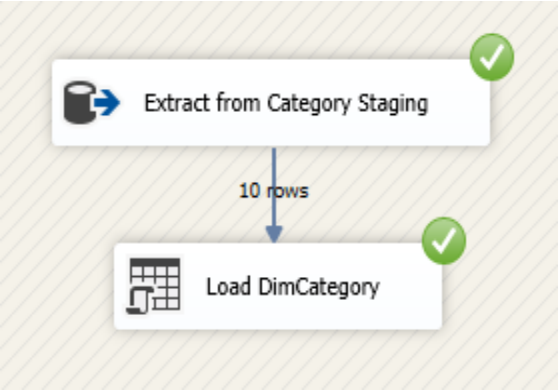


Figure 12. Data Flow Task of Category Dimension Transformation and Loading

2. **Loading Item Data to DimItem**

   Item Data can be loaded next since they contain reference to the **Category**.

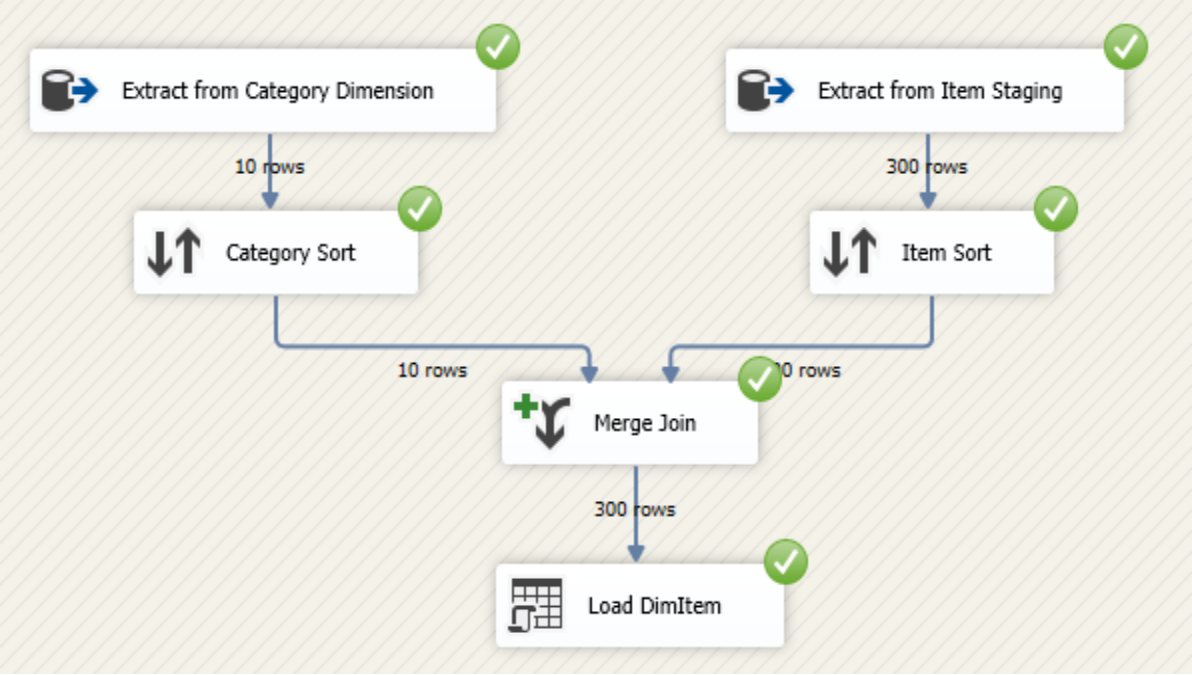


Figure 13. Data Flow Task of Item Dimension Transformation and Loading

DimItem contains a reference to **DimCategory**. In order to get the **Category** surrogate key to **Item** dimension, data was extracted from both dimensions and sorted based on **Category ID**. Then they were merged to load into **DimItem**.

Some **Items** may not have a **Category ID**; thus, the Merge join was done using left outer join.

3. Loading Campaign Data to DimCampaign

The **Campaign** dimension also has no dependencies with any other dimensions; therefore, it is loaded next.

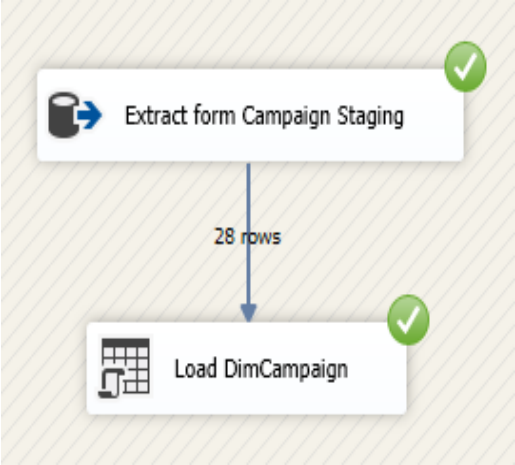


Figure 14. Data Flow Task of Campaign Dimension Transformation and Loading

4. **Loading Train Data to DimTrain**

   Train Data can be loaded next since they contain reference to the **Campaign**.

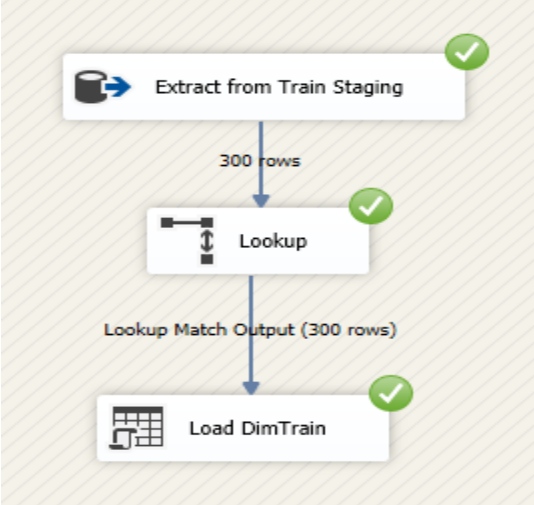


Figure 15. Data Flow Task of Train Dimension Transformation and Loading

Train dimension contains a reference to **Campaign** dimension. In order to get the surrogate keys of **Campaign** to **Train** dimension a lookup process was performed.

5. **Loading District Data to DimDistrict**

   The **District** dimension also has no dependencies with any other dimensions; therefore, it is loaded next.

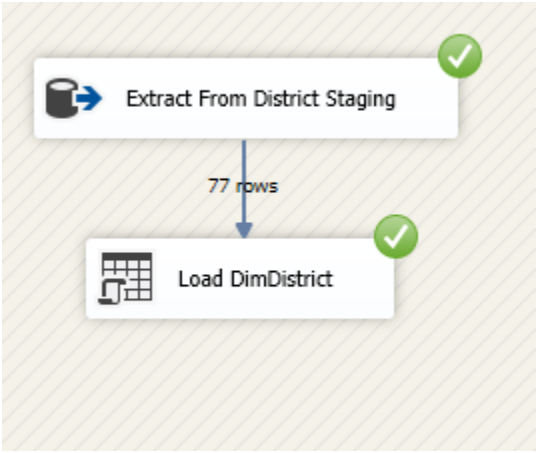
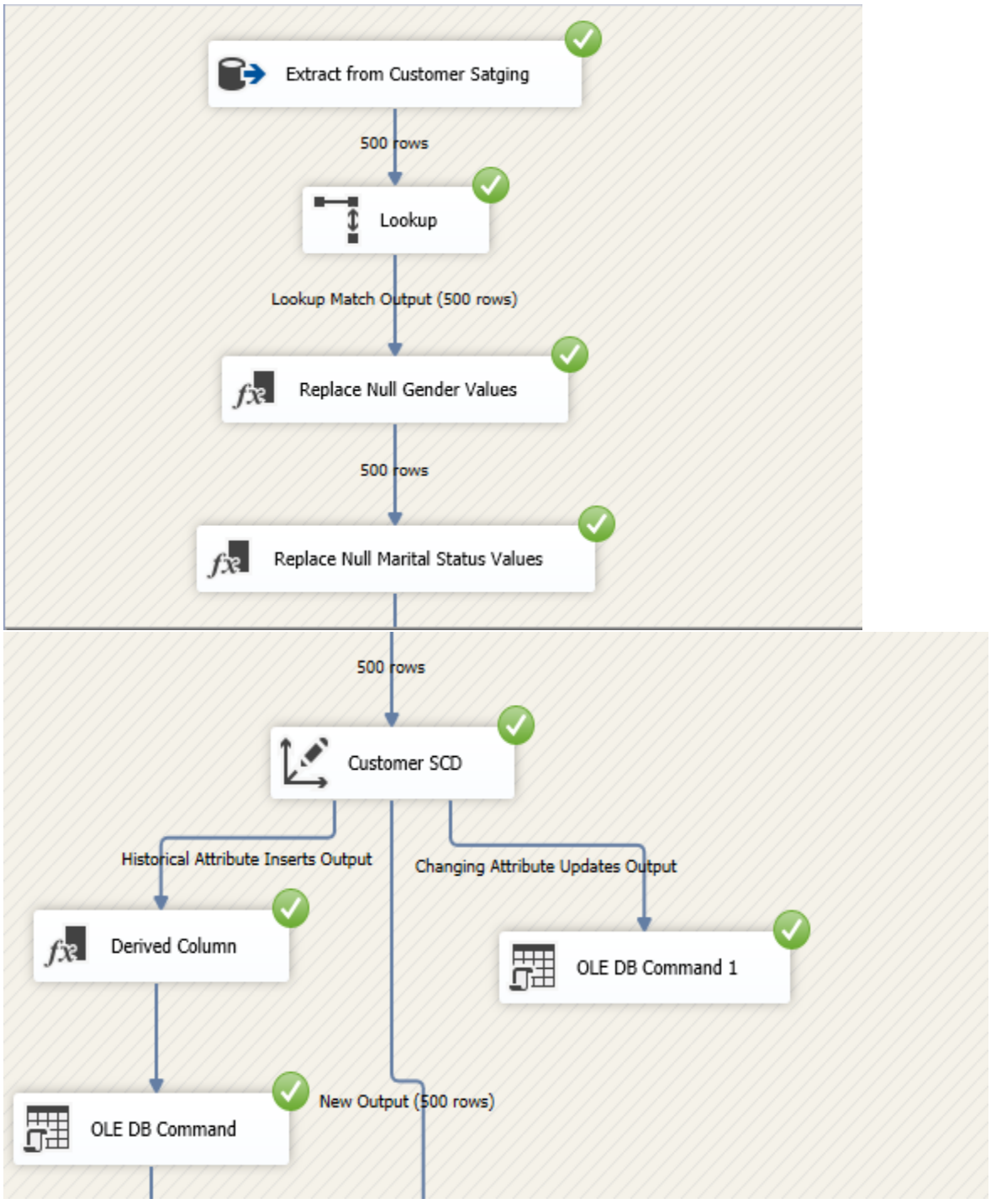


Figure 16. Data Flow Task of District Dimension Transformation and Loading

6. Load Customer Data to DimCustomer

Next, the Customer Dimension loaded



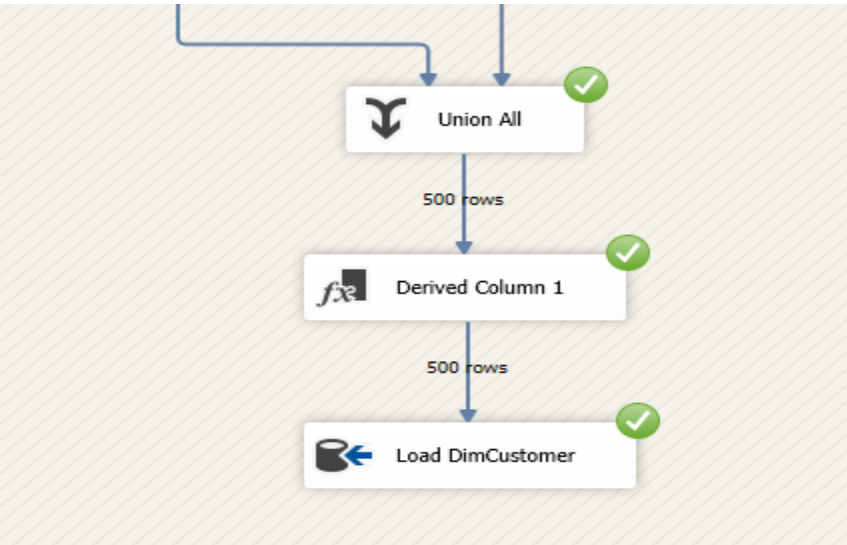


Figure 17. Data Flow Task of Customer Dimension Transformation and Loading

Customer dimension contains a reference to **District** dimension. In order to get the surrogate keys of **District** to **Customer** dimension a lookup process was performed.

DimCustomer 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 StartDate & EndDate 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.

Initially data cleansing is done in order to remove null values from the data source table. Based on data profiling result, null values from Gender column, and Marital Status column were removed.

Therefore, following attributes were set as **changing attributes** and **historical attributes**.

- Marital Status – Changing Attribute
- Phone No – Changing Attribute
- Age Range – Historical Attribute
- DistrictKey – Historical Attribute

After performing these tasks, the **Customer** dimension was loaded.

7. Load Transaction Data to FactTransaction

Finally, the **Fact Transaction** is loaded as it contains references to many other dimensions.

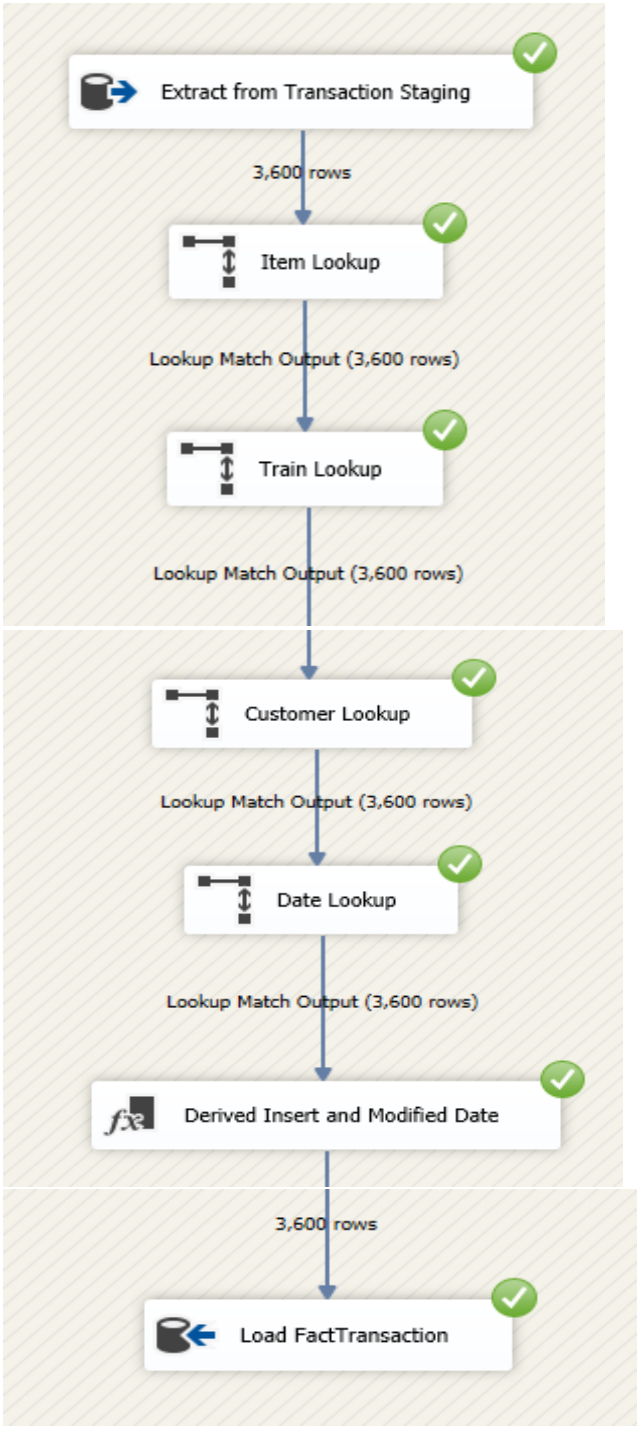


Figure 18. Data Flow Task of Transaction Fact Table Transformation and Loading

The **Fact Transaction** contains references to **Customer**, **Train, Item**, and **Date**. In order to get the surrogate keys as references, lookup processes were carried out for all references. Insert date and modified date are derived columns. Finally, the fact table was loaded to its destination.

For the Accumulating Fact Table need to update AccmTxnCreateTime, AccmTxnCompleteTime, and TxnProcessTimeHour columns.

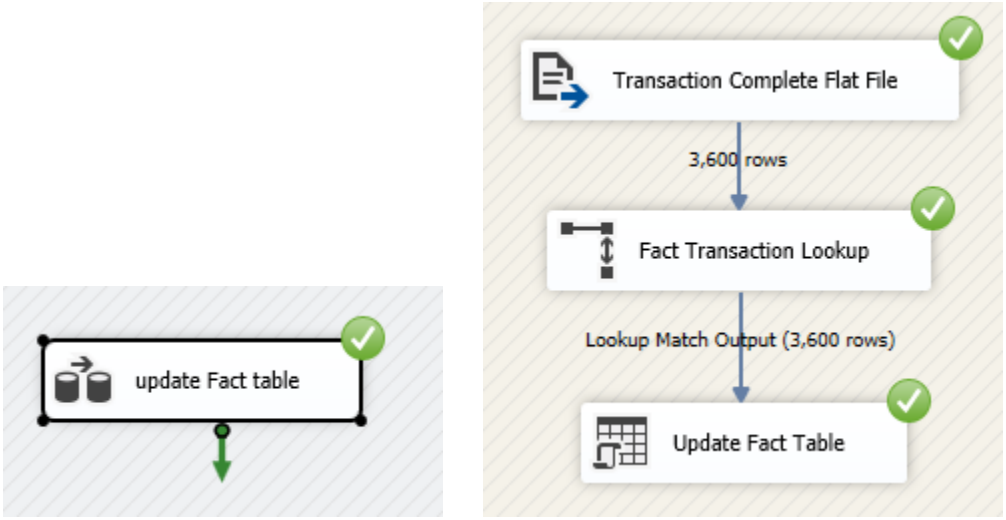


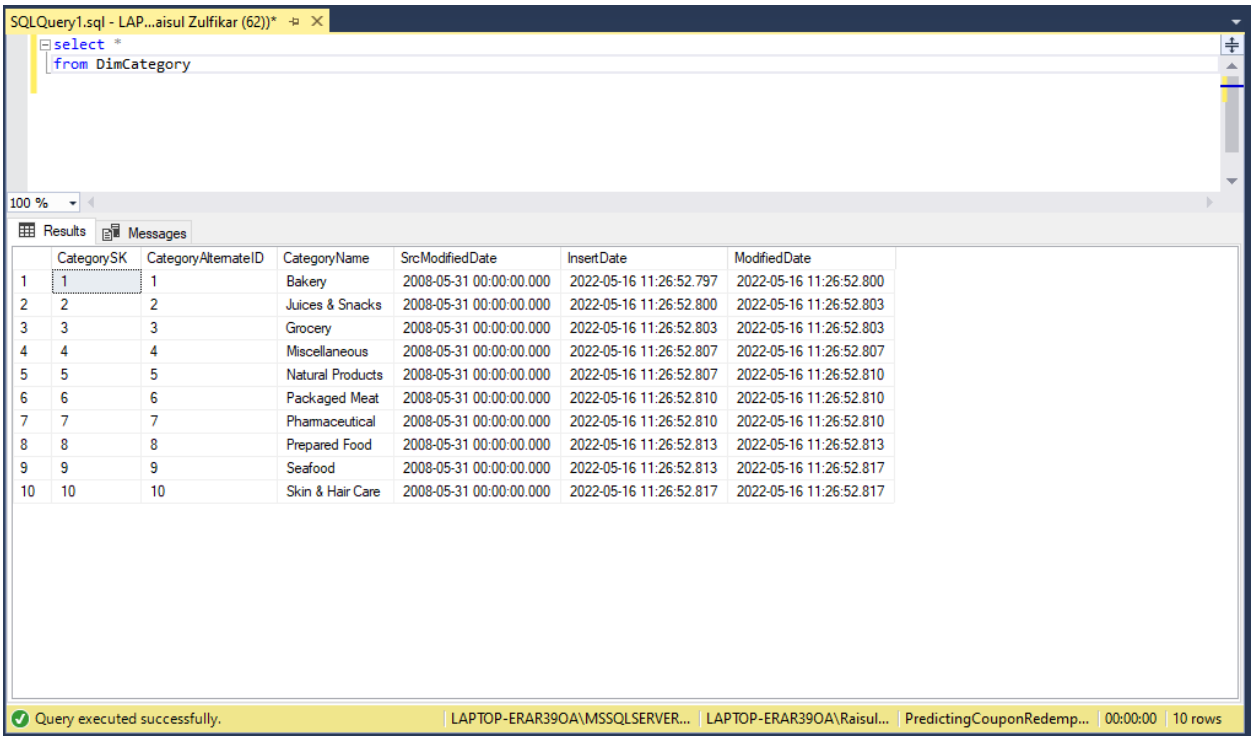
Figure 19. Update Fact Table Control Flow Task

✓ Dimensions like Category, Item, Campaign, and Train does not maintain history. Therefore, in order to maintain the latest record, stored procedures were created.

8.3. Stored Procedure Queries

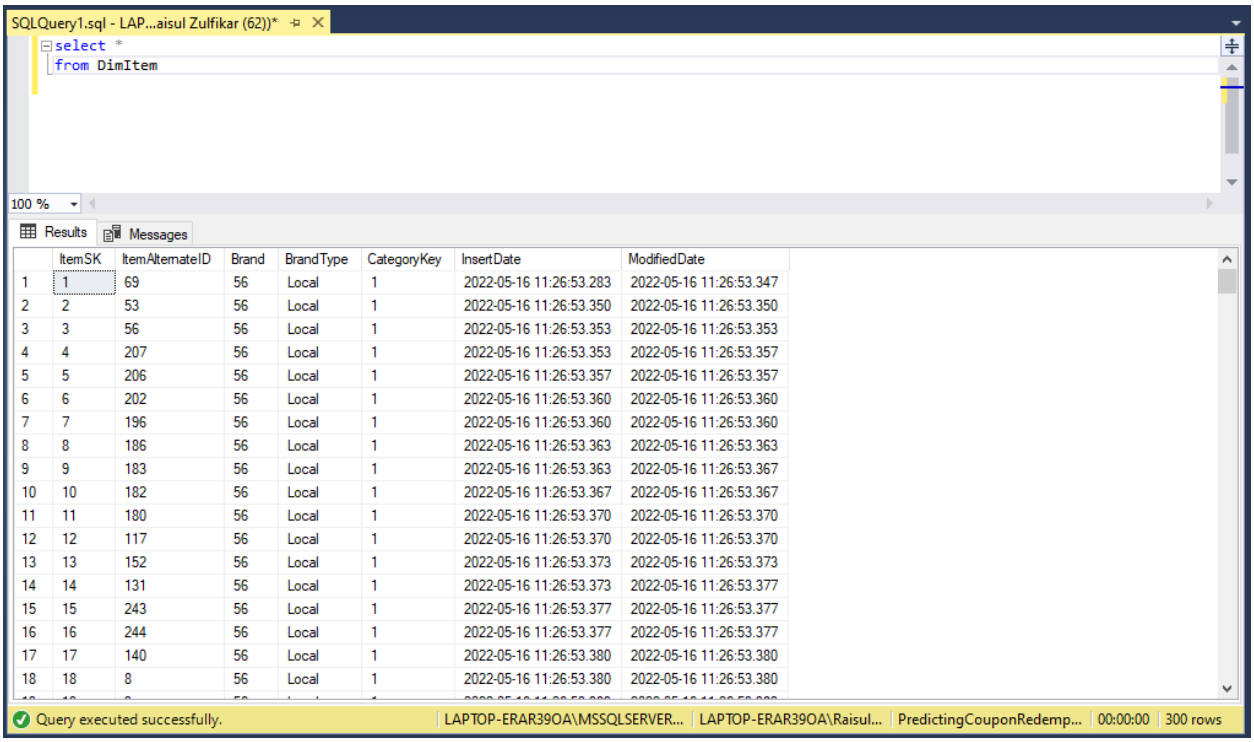
Test Data Loaded from Staging to Datawarehouse

Test Scenario ID			2			
Test Case Description			Transform staging data to dimension tables			
Pre-Requisite			Test loaded from staging to dimension table			
ID	**Action**	**SQL Queries**	**Expected Output**	**Actual Output**	**Result**	**Refer**
1	Data passed from StgCategory to DimCategory	Select * From DimCategory	Display selected rows	Displayed selected rows	PASS	Figure 2.1.
2	Data passed from StgItem to DimItem	Select * From DimItem	Display selected rows	Displayed selected rows	PASS	Figure 2.2.
3	Data passed from StgCampaign to DimCampaign	Select * From DimCampaign	Display selected rows	Displayed selected rows	PASS	Figure 2.3.
4	Data passed from StgTrain to DimTrain	Select * From DimTrain	Display selected rows	Displayed selected rows	PASS	Figure 2.4.
5	Data passed from StgDistrict to DimDistrict	Select * From DimDistrict	Display selected rows	Displayed selected rows	PASS	Figure 2.5.
6	Data passed from StgCustomer to DimCustomer	Select * From DimCustomer	Display selected rows	Displayed selected rows	PASS	Figure 2.6.
7	Data passed from StgTransaction to FactTransaction	Select * From FactTransaction	Display selected rows	Displayed selected rows	PASS	Figure 2.6.



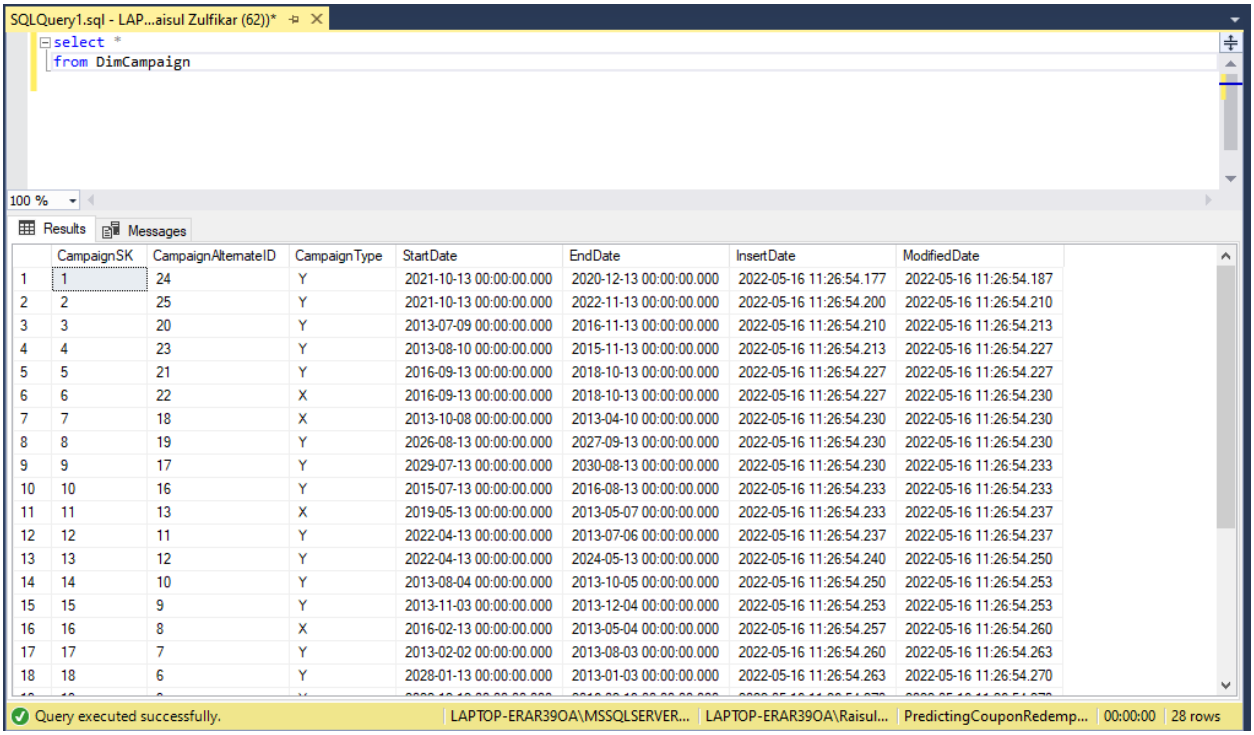
	CategorySK	CategoryAlternateID	CategoryName	SrcModifiedDate	InsertDate	ModifiedDate
1	1	1	Bakery	2008-05-31 00:00:00.000	2022-05-16 11:26:52.797	2022-05-16 11:26:52.800
2	2	2	Juices & Snacks	2008-05-31 00:00:00.000	2022-05-16 11:26:52.800	2022-05-16 11:26:52.803
3	3	3	Grocery	2008-05-31 00:00:00.000	2022-05-16 11:26:52.803	2022-05-16 11:26:52.803
4	4	4	Miscellaneous	2008-05-31 00:00:00.000	2022-05-16 11:26:52.807	2022-05-16 11:26:52.807
5	5	5	Natural Products	2008-05-31 00:00:00.000	2022-05-16 11:26:52.807	2022-05-16 11:26:52.810
6	6	6	Packaged Meat	2008-05-31 00:00:00.000	2022-05-16 11:26:52.810	2022-05-16 11:26:52.810
7	7	7	Pharmaceutical	2008-05-31 00:00:00.000	2022-05-16 11:26:52.810	2022-05-16 11:26:52.810
8	8	8	Prepared Food	2008-05-31 00:00:00.000	2022-05-16 11:26:52.813	2022-05-16 11:26:52.813
9	9	9	Seafood	2008-05-31 00:00:00.000	2022-05-16 11:26:52.813	2022-05-16 11:26:52.817
10	10	10	Skin & Hair Care	2008-05-31 00:00:00.000	2022-05-16 11:26:52.817	2022-05-16 11:26:52.817

Figure 2. 1. DimCategory



	ItemSK	ItemAlternateID	Brand	BrandType	CategoryKey	InsertDate	ModifiedDate
1	1	69	56	Local	1	2022-05-16 11:26:53.283	2022-05-16 11:26:53.347
2	2	53	56	Local	1	2022-05-16 11:26:53.350	2022-05-16 11:26:53.350
3	3	56	56	Local	1	2022-05-16 11:26:53.353	2022-05-16 11:26:53.353
4	4	207	56	Local	1	2022-05-16 11:26:53.353	2022-05-16 11:26:53.357
5	5	206	56	Local	1	2022-05-16 11:26:53.357	2022-05-16 11:26:53.357
6	6	202	56	Local	1	2022-05-16 11:26:53.360	2022-05-16 11:26:53.360
7	7	196	56	Local	1	2022-05-16 11:26:53.360	2022-05-16 11:26:53.360
8	8	186	56	Local	1	2022-05-16 11:26:53.363	2022-05-16 11:26:53.363
9	9	183	56	Local	1	2022-05-16 11:26:53.363	2022-05-16 11:26:53.367
10	10	182	56	Local	1	2022-05-16 11:26:53.367	2022-05-16 11:26:53.367
11	11	180	56	Local	1	2022-05-16 11:26:53.370	2022-05-16 11:26:53.370
12	12	117	56	Local	1	2022-05-16 11:26:53.370	2022-05-16 11:26:53.370
13	13	152	56	Local	1	2022-05-16 11:26:53.373	2022-05-16 11:26:53.373
14	14	131	56	Local	1	2022-05-16 11:26:53.373	2022-05-16 11:26:53.377
15	15	243	56	Local	1	2022-05-16 11:26:53.377	2022-05-16 11:26:53.377
16	16	244	56	Local	1	2022-05-16 11:26:53.377	2022-05-16 11:26:53.377
17	17	140	56	Local	1	2022-05-16 11:26:53.380	2022-05-16 11:26:53.380
18	18	8	56	Local	1	2022-05-16 11:26:53.380	2022-05-16 11:26:53.380

Figure 2. 2. DimItem



SQLQuery1.sql - LAP...aisul Zulfikar (62))*

```
select *
from DimCampaign
```

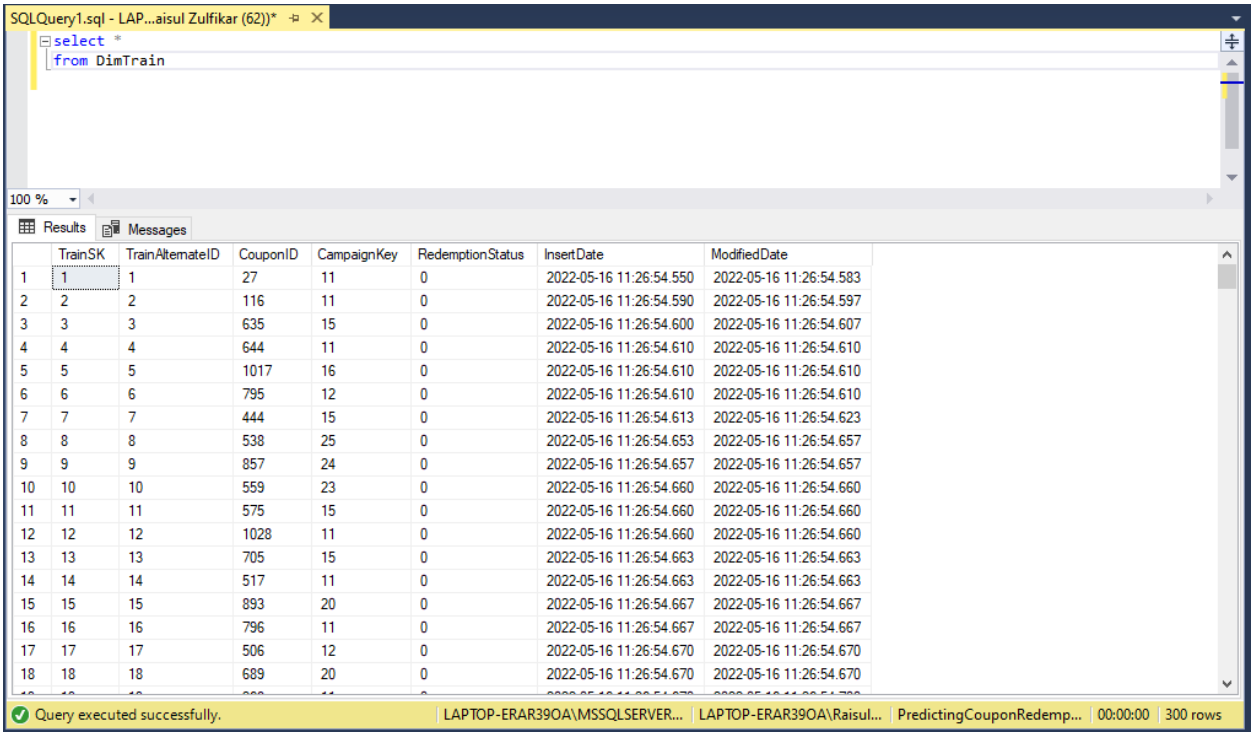
100 %

Results | Messages

	CampaignSK	CampaignAlternateID	CampaignType	StartDate	EndDate	InsertDate	ModifiedDate
1	1	24	Y	2021-10-13 00:00:00.000	2020-12-13 00:00:00.000	2022-05-16 11:26:54.177	2022-05-16 11:26:54.187
2	2	25	Y	2021-10-13 00:00:00.000	2022-11-13 00:00:00.000	2022-05-16 11:26:54.200	2022-05-16 11:26:54.210
3	3	20	Y	2013-07-09 00:00:00.000	2016-11-13 00:00:00.000	2022-05-16 11:26:54.210	2022-05-16 11:26:54.213
4	4	23	Y	2013-08-10 00:00:00.000	2015-11-13 00:00:00.000	2022-05-16 11:26:54.213	2022-05-16 11:26:54.227
5	5	21	Y	2016-09-13 00:00:00.000	2018-10-13 00:00:00.000	2022-05-16 11:26:54.227	2022-05-16 11:26:54.227
6	6	22	X	2016-09-13 00:00:00.000	2018-10-13 00:00:00.000	2022-05-16 11:26:54.227	2022-05-16 11:26:54.230
7	7	18	X	2013-10-08 00:00:00.000	2013-04-10 00:00:00.000	2022-05-16 11:26:54.230	2022-05-16 11:26:54.230
8	8	19	Y	2026-08-13 00:00:00.000	2027-09-13 00:00:00.000	2022-05-16 11:26:54.230	2022-05-16 11:26:54.230
9	9	17	Y	2029-07-13 00:00:00.000	2030-08-13 00:00:00.000	2022-05-16 11:26:54.230	2022-05-16 11:26:54.233
10	10	16	Y	2015-07-13 00:00:00.000	2016-08-13 00:00:00.000	2022-05-16 11:26:54.233	2022-05-16 11:26:54.233
11	11	13	X	2019-05-13 00:00:00.000	2013-05-07 00:00:00.000	2022-05-16 11:26:54.233	2022-05-16 11:26:54.237
12	12	11	Y	2022-04-13 00:00:00.000	2013-07-06 00:00:00.000	2022-05-16 11:26:54.237	2022-05-16 11:26:54.237
13	13	12	Y	2022-04-13 00:00:00.000	2024-05-13 00:00:00.000	2022-05-16 11:26:54.240	2022-05-16 11:26:54.250
14	14	10	Y	2013-08-04 00:00:00.000	2013-10-05 00:00:00.000	2022-05-16 11:26:54.250	2022-05-16 11:26:54.253
15	15	9	Y	2013-11-03 00:00:00.000	2013-12-04 00:00:00.000	2022-05-16 11:26:54.253	2022-05-16 11:26:54.253
16	16	8	X	2016-02-13 00:00:00.000	2013-05-04 00:00:00.000	2022-05-16 11:26:54.257	2022-05-16 11:26:54.260
17	17	7	Y	2013-02-02 00:00:00.000	2013-08-03 00:00:00.000	2022-05-16 11:26:54.260	2022-05-16 11:26:54.263
18	18	6	Y	2028-01-13 00:00:00.000	2013-01-03 00:00:00.000	2022-05-16 11:26:54.263	2022-05-16 11:26:54.270

Query executed successfully. | LAPTOP-ERAR39OA\MSSQLSERVER... | LAPTOP-ERAR39OA\Raisul... | PredictingCouponRedemp... | 00:00:00 | 28 rows

Figure 2. 3. DimCampaign



SQLQuery1.sql - LAP...aisul Zulfikar (62))*

```
select *
from DimTrain
```

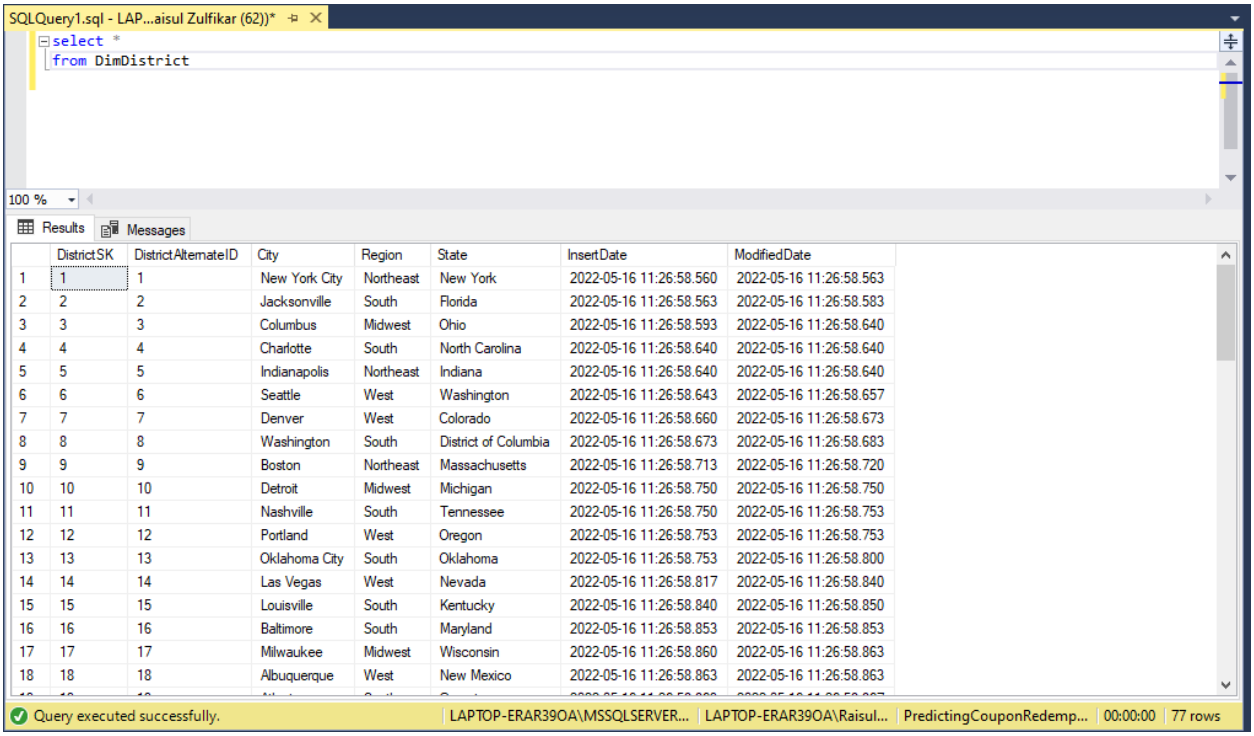
100 %

Results | Messages

	TrainSK	TrainAlternateID	CouponID	CampaignKey	RedemptionStatus	InsertDate	ModifiedDate
1	1	1	27	11	0	2022-05-16 11:26:54.550	2022-05-16 11:26:54.583
2	2	2	116	11	0	2022-05-16 11:26:54.590	2022-05-16 11:26:54.597
3	3	3	635	15	0	2022-05-16 11:26:54.600	2022-05-16 11:26:54.607
4	4	4	644	11	0	2022-05-16 11:26:54.610	2022-05-16 11:26:54.610
5	5	5	1017	16	0	2022-05-16 11:26:54.610	2022-05-16 11:26:54.610
6	6	6	795	12	0	2022-05-16 11:26:54.610	2022-05-16 11:26:54.610
7	7	7	444	15	0	2022-05-16 11:26:54.613	2022-05-16 11:26:54.623
8	8	8	538	25	0	2022-05-16 11:26:54.653	2022-05-16 11:26:54.657
9	9	9	857	24	0	2022-05-16 11:26:54.657	2022-05-16 11:26:54.657
10	10	10	559	23	0	2022-05-16 11:26:54.660	2022-05-16 11:26:54.660
11	11	11	575	15	0	2022-05-16 11:26:54.660	2022-05-16 11:26:54.660
12	12	12	1028	11	0	2022-05-16 11:26:54.660	2022-05-16 11:26:54.660
13	13	13	705	15	0	2022-05-16 11:26:54.663	2022-05-16 11:26:54.663
14	14	14	517	11	0	2022-05-16 11:26:54.663	2022-05-16 11:26:54.663
15	15	15	893	20	0	2022-05-16 11:26:54.667	2022-05-16 11:26:54.667
16	16	16	796	11	0	2022-05-16 11:26:54.667	2022-05-16 11:26:54.667
17	17	17	506	12	0	2022-05-16 11:26:54.670	2022-05-16 11:26:54.670
18	18	18	689	20	0	2022-05-16 11:26:54.670	2022-05-16 11:26:54.670

Query executed successfully. | LAPTOP-ERAR39OA\MSSQLSERVER... | LAPTOP-ERAR39OA\Raisul... | PredictingCouponRedemp... | 00:00:00 | 300 rows

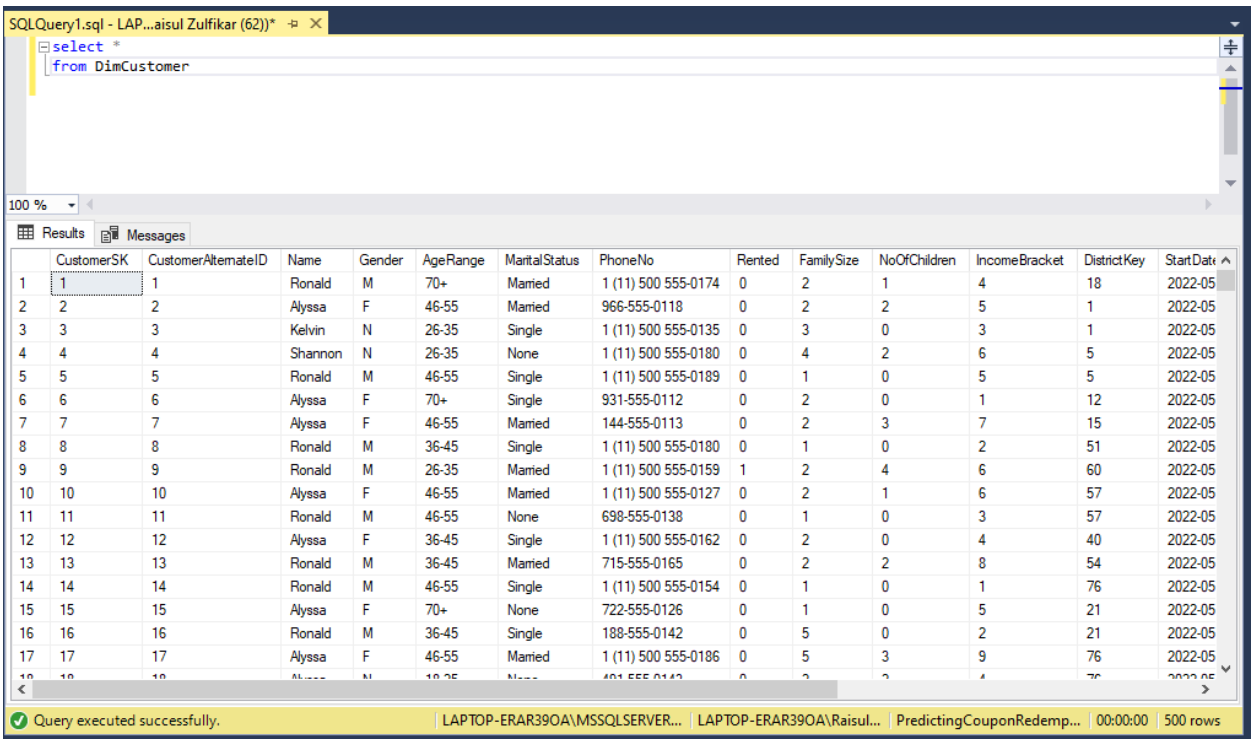
Figure 2. 4. DimTrain



```
select *
from DimDistrict
```

	DistrictSK	DistrictAlternateID	City	Region	State	InsertDate	ModifiedDate
1	1	1	New York City	Northeast	New York	2022-05-16 11:26:58.560	2022-05-16 11:26:58.563
2	2	2	Jacksonville	South	Florida	2022-05-16 11:26:58.563	2022-05-16 11:26:58.583
3	3	3	Columbus	Midwest	Ohio	2022-05-16 11:26:58.593	2022-05-16 11:26:58.640
4	4	4	Charlotte	South	North Carolina	2022-05-16 11:26:58.640	2022-05-16 11:26:58.640
5	5	5	Indianapolis	Northeast	Indiana	2022-05-16 11:26:58.640	2022-05-16 11:26:58.640
6	6	6	Seattle	West	Washington	2022-05-16 11:26:58.643	2022-05-16 11:26:58.657
7	7	7	Denver	West	Colorado	2022-05-16 11:26:58.660	2022-05-16 11:26:58.673
8	8	8	Washington	South	District of Columbia	2022-05-16 11:26:58.673	2022-05-16 11:26:58.683
9	9	9	Boston	Northeast	Massachusetts	2022-05-16 11:26:58.713	2022-05-16 11:26:58.720
10	10	10	Detroit	Midwest	Michigan	2022-05-16 11:26:58.750	2022-05-16 11:26:58.750
11	11	11	Nashville	South	Tennessee	2022-05-16 11:26:58.750	2022-05-16 11:26:58.753
12	12	12	Portland	West	Oregon	2022-05-16 11:26:58.753	2022-05-16 11:26:58.753
13	13	13	Oklahoma City	South	Oklahoma	2022-05-16 11:26:58.753	2022-05-16 11:26:58.800
14	14	14	Las Vegas	West	Nevada	2022-05-16 11:26:58.817	2022-05-16 11:26:58.840
15	15	15	Louisville	South	Kentucky	2022-05-16 11:26:58.840	2022-05-16 11:26:58.850
16	16	16	Baltimore	South	Maryland	2022-05-16 11:26:58.853	2022-05-16 11:26:58.853
17	17	17	Milwaukee	Midwest	Wisconsin	2022-05-16 11:26:58.860	2022-05-16 11:26:58.863
18	18	18	Albuquerque	West	New Mexico	2022-05-16 11:26:58.863	2022-05-16 11:26:58.863

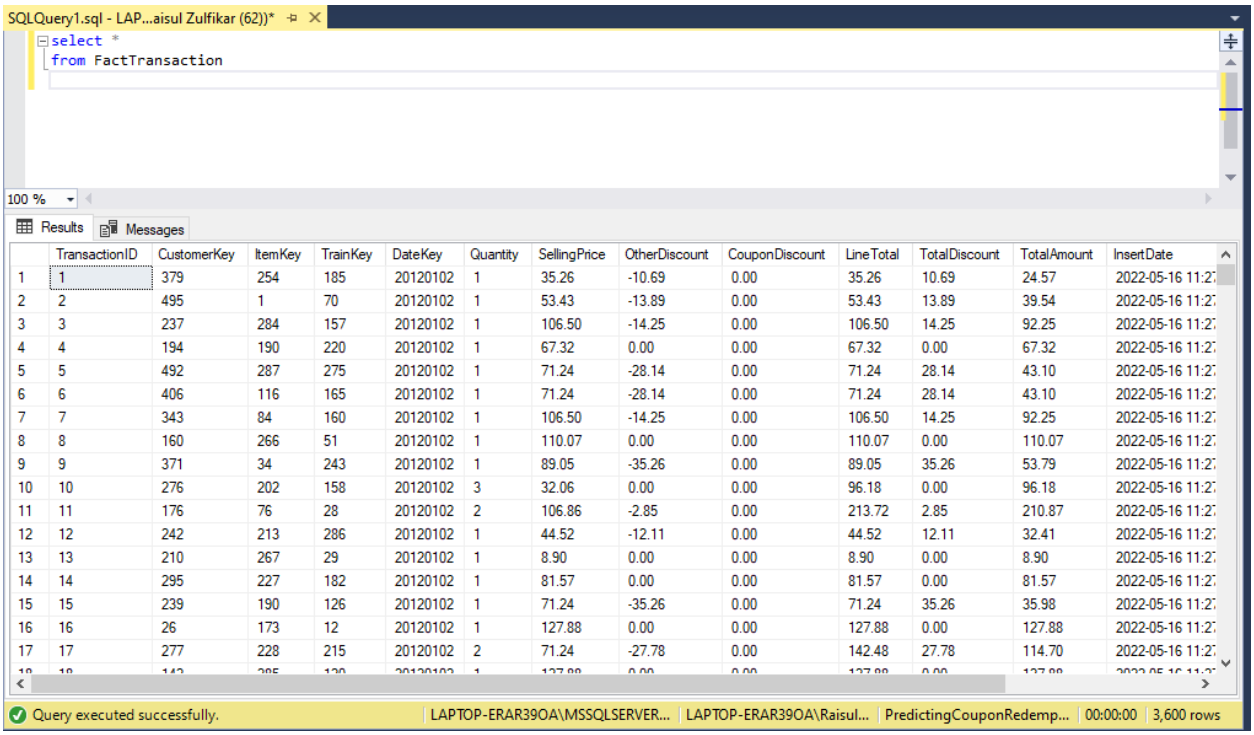
Figure 2. 5. DimDistrict



```
select *
from DimCustomer
```

	CustomerSK	CustomerAlternateID	Name	Gender	AgeRange	MaritalStatus	PhoneNo	Rented	FamilySize	NoOfChildren	IncomeBracket	DistrictKey	StartDate
1	1	1	Ronald	M	70+	Married	1 (11) 500 555-0174	0	2	1	4	18	2022-05
2	2	2	Alyssa	F	46-55	Married	966-555-0118	0	2	2	5	1	2022-05
3	3	3	Kelvin	N	26-35	Single	1 (11) 500 555-0135	0	3	0	3	1	2022-05
4	4	4	Shannon	N	26-35	None	1 (11) 500 555-0180	0	4	2	6	5	2022-05
5	5	5	Ronald	M	46-55	Single	1 (11) 500 555-0189	0	1	0	5	5	2022-05
6	6	6	Alyssa	F	70+	Single	931-555-0112	0	2	0	1	12	2022-05
7	7	7	Alyssa	F	46-55	Married	144-555-0113	0	2	3	7	15	2022-05
8	8	8	Ronald	M	36-45	Single	1 (11) 500 555-0180	0	1	0	2	51	2022-05
9	9	9	Ronald	M	26-35	Married	1 (11) 500 555-0159	1	2	4	6	60	2022-05
10	10	10	Alyssa	F	46-55	Married	1 (11) 500 555-0127	0	2	1	6	57	2022-05
11	11	11	Ronald	M	46-55	None	698-555-0138	0	1	0	3	57	2022-05
12	12	12	Alyssa	F	36-45	Single	1 (11) 500 555-0162	0	2	0	4	40	2022-05
13	13	13	Ronald	M	36-45	Married	715-555-0165	0	2	2	8	54	2022-05
14	14	14	Ronald	M	46-55	Single	1 (11) 500 555-0154	0	1	0	1	76	2022-05
15	15	15	Alyssa	F	70+	None	722-555-0126	0	1	0	5	21	2022-05
16	16	16	Ronald	M	36-45	Single	188-555-0142	0	5	0	2	21	2022-05
17	17	17	Alyssa	F	46-55	Married	1 (11) 500 555-0186	0	5	3	9	76	2022-05

Figure 2. 6. DimCustomer



	TransactionID	CustomerKey	ItemKey	TrainKey	DateKey	Quantity	SellingPrice	OtherDiscount	CouponDiscount	LineTotal	TotalDiscount	TotalAmount	InsertDate
1	1	379	254	185	20120102	1	35.26	-10.69	0.00	35.26	10.69	24.57	2022-05-16 11:2
2	2	495	1	70	20120102	1	53.43	-13.89	0.00	53.43	13.89	39.54	2022-05-16 11:2
3	3	237	284	157	20120102	1	106.50	-14.25	0.00	106.50	14.25	92.25	2022-05-16 11:2
4	4	194	190	220	20120102	1	67.32	0.00	0.00	67.32	0.00	67.32	2022-05-16 11:2
5	5	492	287	275	20120102	1	71.24	-28.14	0.00	71.24	28.14	43.10	2022-05-16 11:2
6	6	406	116	165	20120102	1	71.24	-28.14	0.00	71.24	28.14	43.10	2022-05-16 11:2
7	7	343	84	160	20120102	1	106.50	-14.25	0.00	106.50	14.25	92.25	2022-05-16 11:2
8	8	160	266	51	20120102	1	110.07	0.00	0.00	110.07	0.00	110.07	2022-05-16 11:2
9	9	371	34	243	20120102	1	89.05	-35.26	0.00	89.05	35.26	53.79	2022-05-16 11:2
10	10	276	202	158	20120102	3	32.06	0.00	0.00	96.18	0.00	96.18	2022-05-16 11:2
11	11	176	76	28	20120102	2	106.86	-2.85	0.00	213.72	2.85	210.87	2022-05-16 11:2
12	12	242	213	286	20120102	1	44.52	-12.11	0.00	44.52	12.11	32.41	2022-05-16 11:2
13	13	210	267	29	20120102	1	8.90	0.00	0.00	8.90	0.00	8.90	2022-05-16 11:2
14	14	295	227	182	20120102	1	81.57	0.00	0.00	81.57	0.00	81.57	2022-05-16 11:2
15	15	239	190	126	20120102	1	71.24	-35.26	0.00	71.24	35.26	35.98	2022-05-16 11:2
16	16	26	173	12	20120102	1	127.88	0.00	0.00	127.88	0.00	127.88	2022-05-16 11:2
17	17	277	228	215	20120102	2	71.24	-27.78	0.00	142.48	27.78	114.70	2022-05-16 11:2

Figure 2. 7. FactTransaction

7. Appendix

7.1.Staging Tables Creation

--- Create StgCategory Table ---

```
CREATE TABLE [dbo].[StgCategory](
    [CategoryID] [int] NULL,
    [CategoryName] [varchar](50) NULL,
    [SrcModifiedDate] [datetime] NULL
) ON [PRIMARY]
```

--- Create StgItem Table ---

```
CREATE TABLE [dbo].[StgItem](
    [ItemID] [int] NULL,
    [Brand] [int] NULL,
    [BrandType] [varchar](50) NULL,
    [CategoryID] [int] NULL
) ON [PRIMARY]
```

--- Create StgCampaign Table ---

```
CREATE TABLE [dbo].[StgCampaign](
    [CampaignID] [int] NULL,
    [CampaignType] [varchar](1) NULL,
    [StartDate] [datetime] NULL,
    [EndDate] [datetime] NULL
) ON [PRIMARY]
```

--- Create StgTrain Table ---

```
CREATE TABLE [dbo].[StgTrain](
    [TrainID] [int] NULL,
    [CampaignID] [int] NULL,
    [CouponID] [int] NULL,
    [RedemptionStatus] [int] NULL
) ON [PRIMARY]
```

--- Create StgDistrict Table ---

```
CREATE TABLE [dbo].[StgDistrict](
    [DistrictID] [int] NULL,
    [DistrictName] [varchar](50) NULL,
    [Region] [varchar](50) NULL,
    [State] [varchar](50) NULL
) ON [PRIMARY]
```

--- Create StgCustomer Table ---

```
CREATE TABLE [dbo].[StgCustomer](
    [CustomerID] [int] NULL,
    [Name] [varchar](50) NULL,
    [Gender] [varchar](1) NULL,
    [AgeRange] [varchar](50) NULL,
    [MaritalStatus] [varchar](50) NULL,
    [PhoneNo] [varchar](25) NULL,
    [Rented] [int] NULL,
    [FamilySize] [int] NULL,
    [NoOfChildren] [int] NULL,
    [IncomeBracket] [int] NULL,
    [DistrictID] [int] NULL
) ON [PRIMARY]
```

--- Create StgTransaction Table ---

```
CREATE TABLE [dbo].[StgTransaction](
    [TransactionID] [int] NULL,
    [CustomerID] [int] NULL,
    [ItemID] [int] NULL,
    [TrainID] [int] NULL,
    [Date] [datetime] NULL,
    [Quantity] [int] NULL,
    [SellingPrice] [money] NULL,
    [OtherDiscount] [money] NULL,
    [CouponDiscount] [money] NULL
) ON [PRIMARY]
```

7.2.Dimension Tables and Fact Table Creation

--- Create DimCategory Table ---

```
CREATE TABLE [dbo].[DimCategory](
	[CategorySK] [int] IDENTITY(1,1) NOT NULL,
	[CategoryAlternateID] [int] NULL,
	[CategoryName] [varchar](50) NULL,
	[SrcModifiedDate] [datetime] NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
 CONSTRAINT [PK_DimCategory] PRIMARY KEY CLUSTERED
(
	[CategorySK] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
```

--- Create DimItem Table ---

```
CREATE TABLE [dbo].[DimItem](
	[ItemSK] [int] IDENTITY(1,1) NOT NULL,
	[ItemAlternateID] [int] NULL,
	[Brand] [int] NULL,
	[BrandType] [varchar](50) NULL,
	[CategoryKey] [int] NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
 CONSTRAINT [PK_DimItem] PRIMARY KEY CLUSTERED
(
	[ItemSK] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
```

--- Create DimCampaign Table ---

```
CREATE TABLE [dbo].[DimCampaign](
	[CampaignSK] [int] IDENTITY(1,1) NOT NULL,
	[CampaignAlternateID] [int] NULL,
	[CampaignType] [varchar](1) NULL,
	[StartDate] [datetime] NULL,
	[EndDate] [datetime] NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
 CONSTRAINT [PK_DimCampaign] PRIMARY KEY CLUSTERED
(
	[CampaignSK] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
```

--- Create DimTrain Table ---

```
CREATE TABLE [dbo].[DimTrain](
	[TrainSK] [int] IDENTITY(1,1) NOT NULL,
	[TrainAlternateID] [int] NULL,
	[CouponID] [int] NULL,
	[CampaignKey] [int] NULL,
	[RedemptionStatus] [int] NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
 CONSTRAINT [PK_DimTrain] PRIMARY KEY CLUSTERED
(
	[TrainSK] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
```

--- Create DimDistrict Table ---

```
CREATE TABLE [dbo].[DimDistrict](
	[DistrictSK] [int] IDENTITY(1,1) NOT NULL,
	[DistrictAlternateID] [int] NULL,
	[City] [varchar](50) NULL,
	[Region] [varchar](50) NULL,
	[State] [varchar](50) NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
 CONSTRAINT [PK_DimDistrict] PRIMARY KEY CLUSTERED
(
	[DistrictSK] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
```

--- Create DimCustomer Table ---

```
CREATE TABLE [dbo].[DimCustomer](
	[CustomerSK] [int] IDENTITY(1,1) NOT NULL,
	[CustomerAlternateID] [int] NULL,
	[Name] [varchar](50) NULL,
	[Gender] [varchar](1) NULL,
	[AgeRange] [varchar](50) NULL,
	[MaritalStatus] [varchar](50) NULL,
	[PhoneNo] [varchar](30) NULL,
	[Rented] [int] NULL,
```

```
	[FamilySize] [int] NULL,
	[NoOfChildren] [int] NULL,
	[IncomeBracket] [int] NULL,
	[DistrictKey] [int] NULL,
	[StartDate] [datetime] NULL,
	[EndDate] [datetime] NULL,
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
 CONSTRAINT [PK_DimCustomer] PRIMARY KEY CLUSTERED
(
	[CustomerSK] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY =
OFF) ON [PRIMARY]
) ON [PRIMARY]
```

--- Create FactTransaction Table ---

```
CREATE TABLE [dbo].[FactTransaction](
	[TransactionID] [int] NULL,
	[CustomerKey] [int] NULL,
	[ItemKey] [int] NULL,
	[TrainKey] [int] NULL,
	[DateKey] [int] NULL,
	[Quantity] [int] NULL,
	[SellingPrice] [money] NULL,
	[OtherDiscount] [money] NULL,
	[CouponDiscount] [money] NULL,
	[LineTotal]  AS ([SellingPrice]*[Quantity]),
	[TotalDiscount]  AS ( -([OtherDiscount]+[CouponDiscount])),
	[TotalAmount]  AS
([SellingPrice]*[Quantity]+([CouponDiscount]+[OtherDiscount])),
	[InsertDate] [datetime] NULL,
	[ModifiedDate] [datetime] NULL,
	[AccmTxnCompleteTime] [datetime] NULL,
	[AccmTxnCreateTime] [datetime] NULL,
	[TxnProcessTimeHours]  AS
(datediff(hour,[AccmTxnCompleteTime],[AccmTxnCreateTime]))
) ON [PRIMARY]
```

7.3.Stored Procedures SQL Queries

--- Stored Procedure for DimCategory ---

```
CREATE PROCEDURE dbo.UpdateDimCategory
@CategoryID int,
@CategoryName varchar(50),
@ModifiedDate datetime
AS
BEGIN
if not exists (select CategorySK
from dbo.DimCategory
where CategoryAlternateID= @CategoryID)
BEGIN
insert into dbo.DimCategory
(CategoryAlternateID, CategoryName, SrcModifiedDate, InsertDate, ModifiedDate)
values
(@CategoryID, @CategoryName, @ModifiedDate, GETDATE(), GETDATE())
END;
if exists (select CategorySK
from dbo.DimCategory
where CategoryAlternateID = @CategoryID)
BEGIN
update dbo.DimCategory
set CategoryName = @CategoryName,
SrcModifiedDate = @ModifiedDate,
ModifiedDate = GETDATE()
where CategoryAlternateID = @CategoryID
END;
END;
```

--- Stored Procedure for DimItem ---

```
CREATE PROCEDURE dbo.UpdateDimItem
@ItemID int,
@brand int,
@brandtype varchar(50),
@categoryKey int
AS
BEGIN
if not exists (select ItemSK
from dbo.DimItem
where ItemAlternateID = @ItemID)
BEGIN
insert into dbo.DimItem
(ItemAlternateID, Brand, BrandType, CategoryKey, InsertDate, ModifiedDate)
values
(@ItemID, @brand, @brandtype, @categoryKey, GETDATE(), GETDATE())
END;
if exists (select ItemSK
from dbo.DimItem
where ItemAlternateID = @ItemID)
BEGIN
```

```
update dbo.DimItem
set Brand = @brand,
BrandType = @brandtype,
CategoryKey = @categoryKey,
ModifiedDate = GETDATE()
where ItemAlternateID = @ItemID
END;
END;
```

--- Stored Procedure for DimCampaign ---

```
CREATE PROCEDURE dbo.UpdateDimCampaign
@CampaignID int,
@type varchar(1),
@startdate datetime,
@enddate datetime
AS
BEGIN
if not exists (select CampaignSK
from dbo.DimCampaign
where CampaignAlternateID = @CampaignID)
BEGIN
insert into dbo.DimCampaign
(CampaignAlternateID , CampaignType, StartDate, EndDate, InsertDate,
ModifiedDate)
values
(@CampaignID, @type, @startdate, @enddate, GETDATE(), GETDATE())
END;
if exists (select CampaignSK
from dbo.DimCampaign
where CampaignAlternateID = @CampaignID)
BEGIN
update dbo.DimCampaign
set CampaignType = @type,
StartDate = @startdate,
EndDate = @enddate,
ModifiedDate = GETDATE()
where CampaignAlternateID = @CampaignID
END;
END;
```

--- Stored Procedure for DimTrain ---

```
CREATE PROCEDURE dbo.UpdateDimTrain
@trainID int,
@couponID int,
@campaignKey int,
@redemption int
AS
BEGIN
if not exists (select TrainSK
from dbo.DimTrain
```

```
where TrainAlternateID = @trainID)
BEGIN
insert into dbo.DimTrain
(TrainAlternateID, CouponID, CampaignKey, RedemptionStatus, InsertDate,
ModifiedDate)
values
(@trainID, @couponID, @campaignKey, @redemption, GETDATE(), GETDATE())
END;
if exists (select TrainSK
from dbo.DimTrain
where TrainAlternateID = @trainID)
BEGIN
update dbo.DimTrain
set CouponID = @couponID,
CampaignKey = @campaignKey,
RedemptionStatus = @redemption,
ModifiedDate = GETDATE()
where TrainAlternateID = @trainID
END;
END;
```

--- Stored Procedure for DimDistrict ---

```
CREATE PROCEDURE dbo.UpdateDimDistrict
@districtID int,
@DistrictName varchar(50),
@region varchar(50),
@state varchar(50)
AS BEGIN
if not exists (
select DistrictSK
from dbo.DimDistrict
where DistrictAlternateID = @districtID)
BEGIN
insert into dbo.DimDistrict
(DistrictAlternateID, DistrictName, Region, State, InsertDate, ModifiedDate)
values
(@districtID, @DistrictName, @region, @state, GETDATE(), GETDATE())
END;
if exists (
select DistrictSK
from dbo.DimDistrict
where DistrictAlternateID = @districtID)
BEGIN
update dbo.DimDistrict
set DistrictName = @DistrictName,
Region = @region,
State = @state,
ModifiedDate = GETDATE()
where DistrictAlternateID = @districtID
END;
END;
```

```
--- Stored Procedure for FactTransaction ---
CREATE PROCEDURE dbo.UpdateFactTransaction
@tranID int,
@complete datetime,
@create datetime
AS
BEGIN
if exists (select TransactionID
from dbo.FactTransaction
where TransactionID = @tranID)
BEGIN
update dbo.FactTransaction
set AccmTxnCompleteTime = @complete,
AccmTxnCreateTime = @create
where TransactionID = @tranID
END;
END;
```