Sri Lanka Institute of Information Technology

Data Warehouse and Business Intelligence – Assignment 1 Submission

Name: Andrew Asher

Reg-No: IT20220624

Batch: Year 03 Semester 01

Contents

1.DATA SET SELECTION

Data Set Name: AmExpert 2019 - Predicting Coupon Redemption

Provided by: kaggle.com

Source link: https://www.kaggle.com/bharath901/amexpert-2019

About Dataset:

The selected data source is a collection of transactional data.

Discount marketing and coupon usage are very widely used promotional techniques to attract new customers and to retain & reinforce loyalty of existing customers. The measurement of a consumer's propensity towards coupon usage and the prediction of the redemption behavior are crucial parameters in assessing the effectiveness of a marketing campaign.

ABC is an established Brick & Mortar retailer that frequently conducts marketing campaigns for its diverse product range. ABC's promotions are shared across various channels including email, notifications, etc. A number of these campaigns include coupon discounts that are offered for a specific product/range of products. The retailer would like the ability to predict whether customers redeem the coupons received across channels, which will enable the retailer's marketing team to accurately design coupon construct, and develop more precise and targeted marketing strategies.

Dataset contains five 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.

- train.csv: Train data containing the coupons offered to the given customers under the campaigns
- campaign_data.csv: Campaign information for each of the campaign
- customer_demographics.csv: Customer demographic information for some customers
- customertransactiondata.csv: Transaction data for all customers for duration of campaigns in the train data
- item_data.csv: Item information for each item sold by the retailer

2.PREPARATION OF DATA SOURCE

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 a another table) of the given files as converting into text files and importing csv files into a source database.

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

- category.txt
- AmExpert_SourceDB (Source Database) Tables: -
  - dbo.Campaign
  - dbo.Item
  - dbo.Train
  - dbo.Transaction
  - dbo.CustomerDetails

ER-Diagram

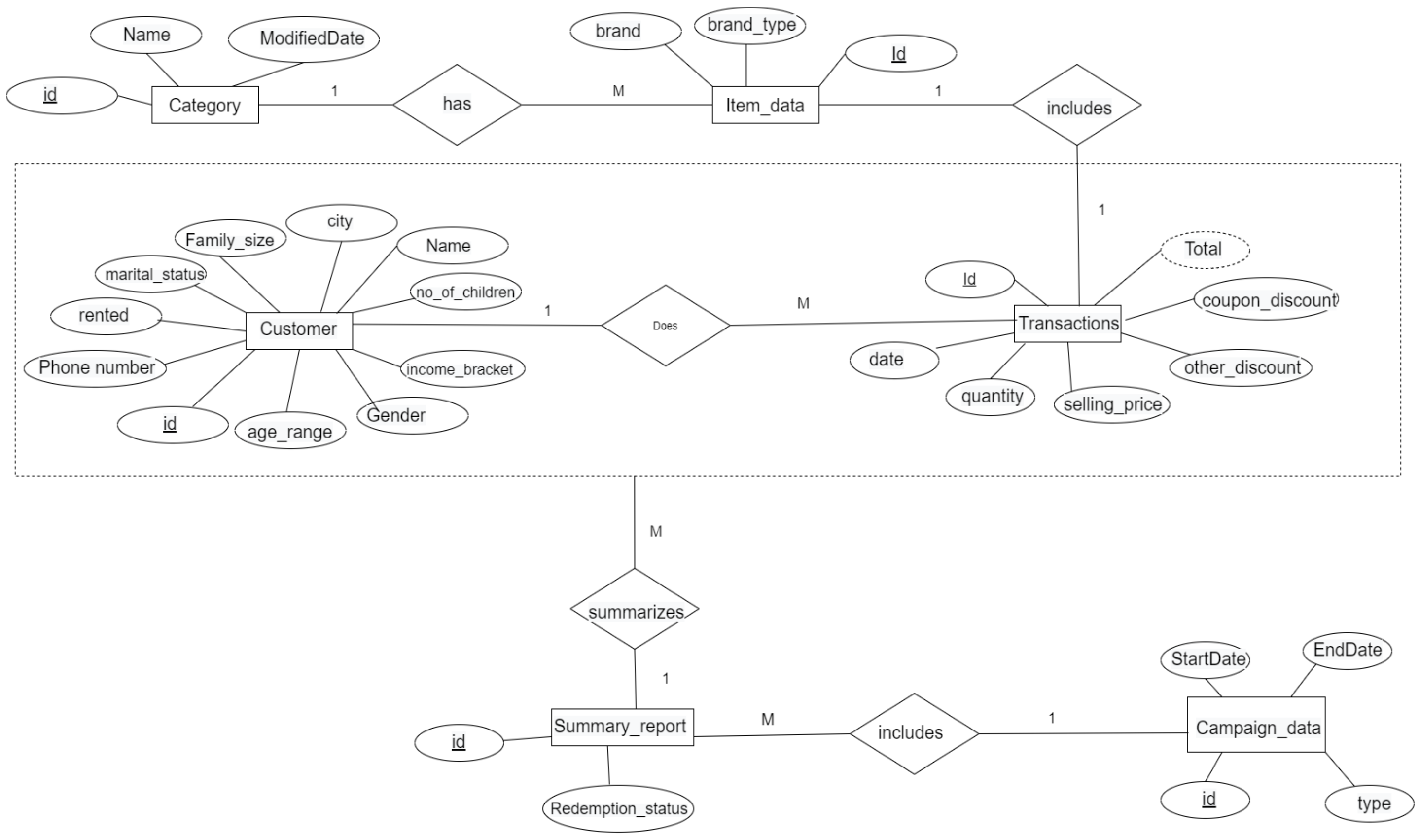


Figure 1: ER-Diagram

- 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.

Description of the data set

Source Type	**Table Name**	**Include**		
Category.txt	Category	Column	Data type	Description
		CategoryID	int	Category key
		Name	nvarchar(50)	Category name
		ModifiedDate	datetime	Modification date of category
AmExpert_SourceDB	Campaign	Column	Data type	Description
		CampaignID	int	Campaign key
		CampaignTpe	nvarchar(1)	Anonymized Campaign Type (X/Y)
		StartDate	datetime	Campaign Start date
		EndDate	datetime	Campaign end Date
	Item	Column	Data type	Description
		ItemID	int	Unique id for item
		Brand	int	Unique id for item brand
		BrandType	nvarchar(50)	Brand Type (local/Established)
		CategoryID	int	Item CategoryID
	Train	Column	Data type	Description
		TrainID	Int	Unique id for coupon customer impression
		CampaignID	Int	Unique id for a discount campaign
		CouponID	Int	Unique id for a discount coupon
		RedemptionStatus	Int	target) (0-Coupon not redeemed, 1 - Coupon redeemed)

Transaction

Column	Data type	Description
TransactionID	int	Unique id for transaction
CustomerID	int	Unique id for a customer
ItemID	int	Unique id for an item
TrainID	int	Unique id for an train
Date	datetime	Date of Transaction
Quantity	int	Quantity of item bought
SellingPrice	money	Sales value of the transaction
OtherDiscount	money	Discount from other sources such as manufacturer coupon/loyalty card
CouponDiscount	money	Discount availed from retailer coupon

Customer

Column	Data type	Description
CustomerID	int	Unique id for a customer
Name	nvarchar(20)	Customer name
Gender	nvarchar(1)	Gender of customer
AgeRange	nvarchar(50)	Age range of customer family in years
MaritalStatus	nvarchar(50)	Married/Single
City	nvarchar(50)	City name
Phone	nvarchar(25)	Contact number
Rented	int	0 - not rented accommodation, 1 - rented accommodation
FamilySize	int	Number of family members
NoOfChild	int	Number of children in the family
IncomeBracket	int	Label Encoded Income Bracket (Higher income corresponds to higher number)

3.SOLUTION ARCHITECTURE

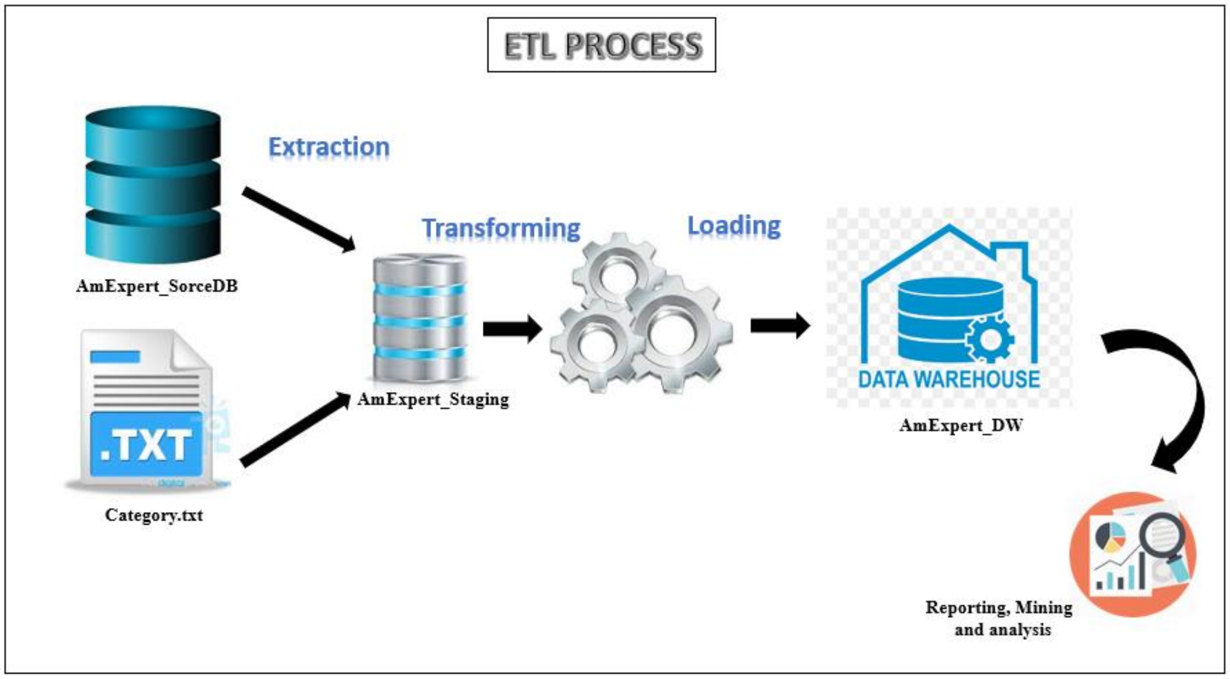


Figure 2: High-level BI Solution Architecture

As the figure 2 shows for the ETL processing, initially

AmExpert_SourceDB: Source Database,

category.txt: Text file,

used for the data extraction to the Staging Destination.

4.DATA WARE HOUSE DESIGN & DEVELOPMENT

i. Design

The AmExpert_DW (ware house) is designed according to the given below snowflake schema with one fact table (dbo.FactTransaction) and six dimension tables including Date dimension.

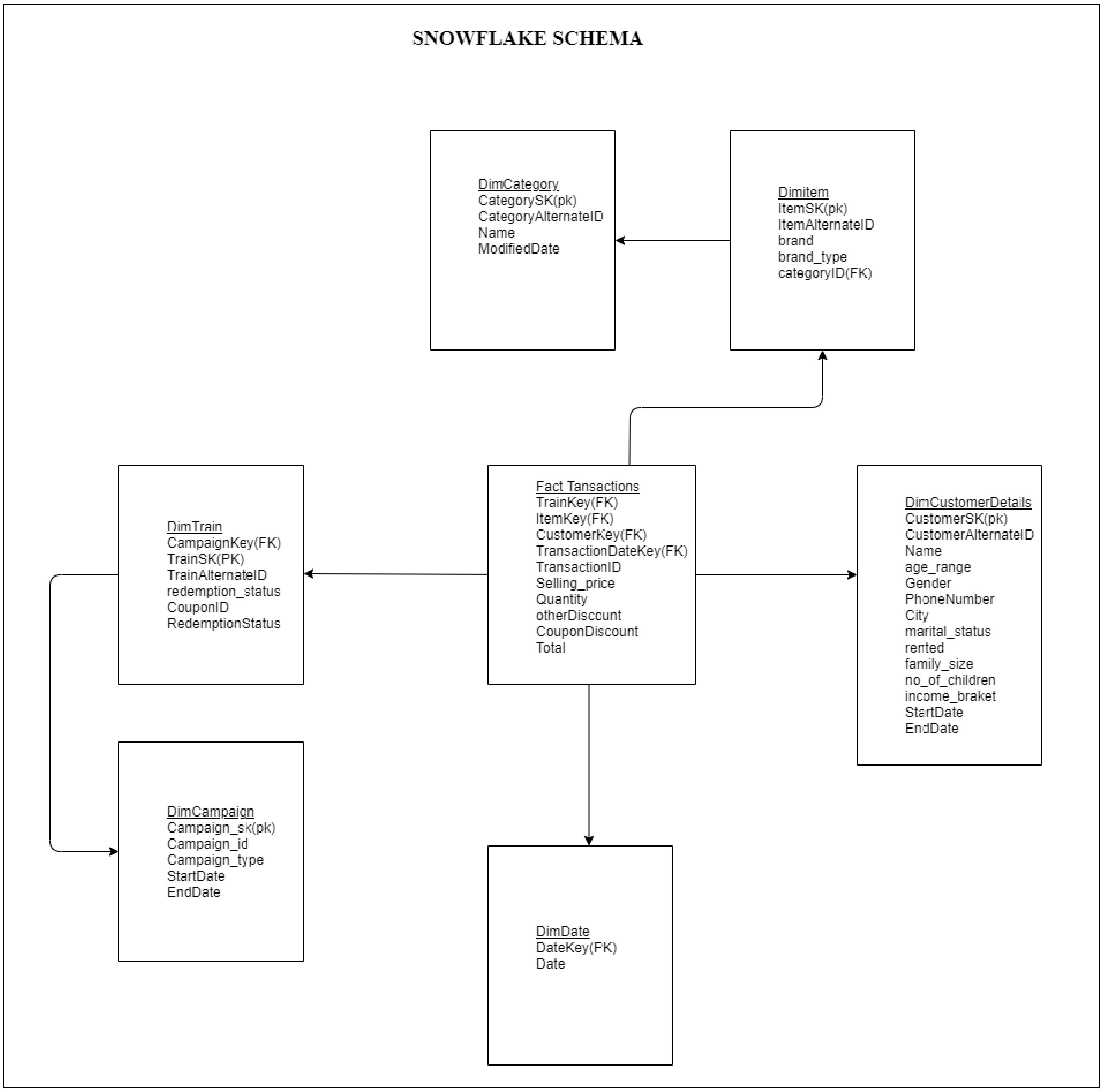


Figure 3: Snowflake schema

- Hierarchies

  - DimCategory is applied as a hierarchical dimension of DimItem table.

  - DimCampaign is applied as a hierarchical dimension of DimTrain table.

- Calculation

  - Total payment is calculated in dbo.FactTransaction.Total

    (([SellingPrice]-([OtherDiscount]+[CouponDiscount]))*[Quantity])

ii. Assumptions

- dbo.DimDate is added to the Data Warehouse for better performance.
- dbo.Transaction is used in creating the fact table.
- The transactions per customer was considered as the grain when designing.

iii. Slowly changing dimensions

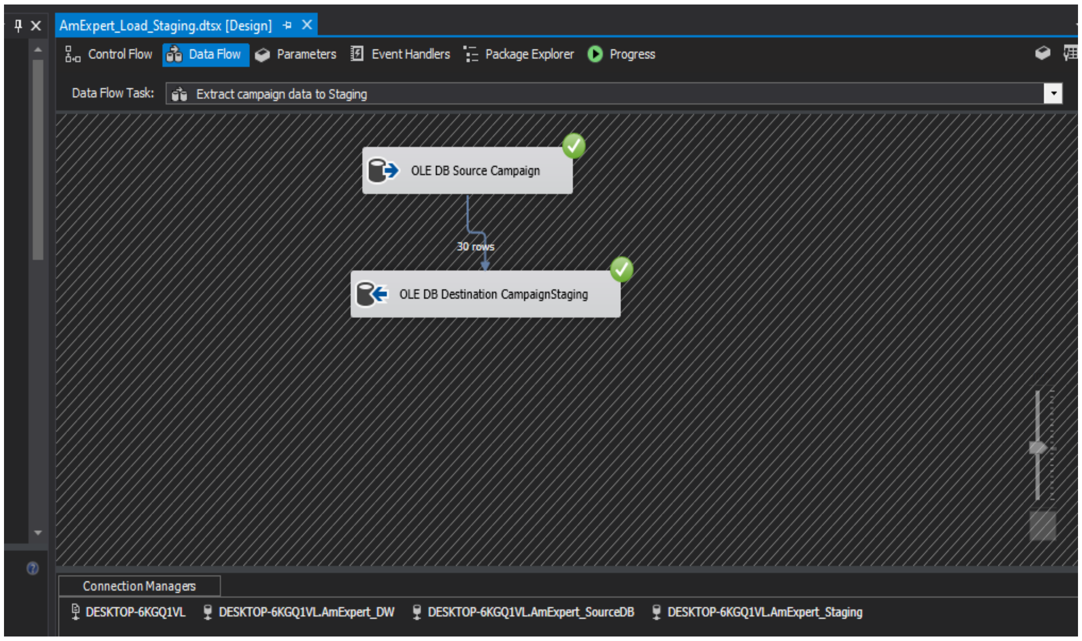
- Customer Details were considered as a slowly changing dimension

Dimension table	**Attributes**
DimCustomerDeails	MaritalStatus (changing attribute) PhoneNumber (changing attribute) AgeRange (Hostorical) City (Hostorical)

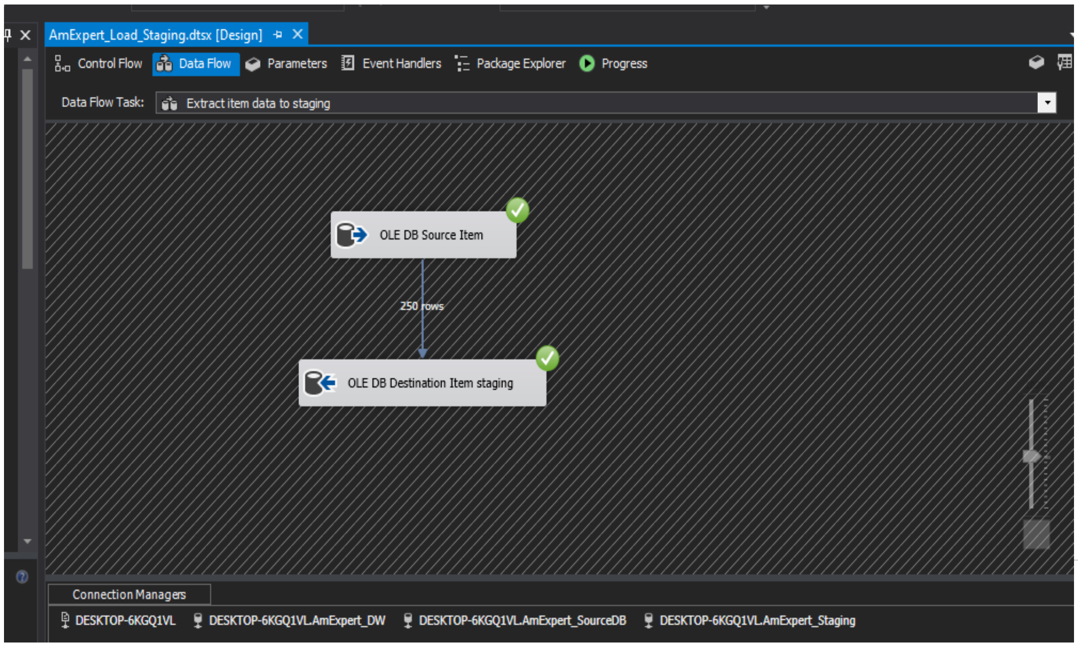
5.ETL DEVELOPMENT

i. Data Extraction & Load into Staging tables

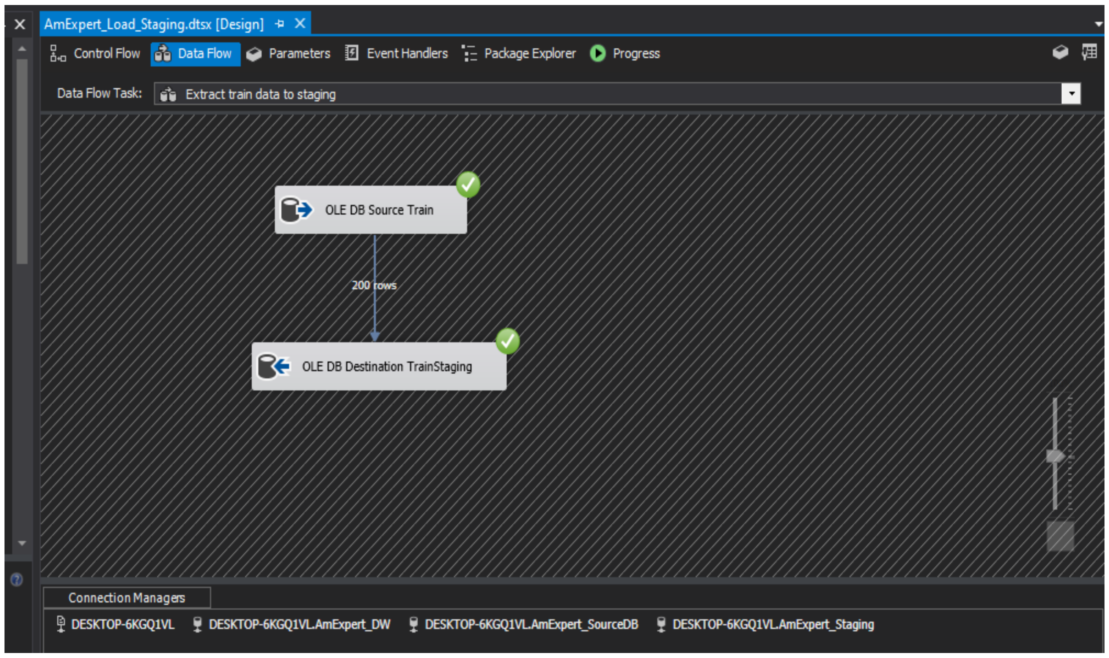
- Data Extraction is done by using the provided data sources mentioned above in Visual Studio 2015 (Data Tool) development environment. The text file and the source database were used here.
- Initially, **OLE DB SOURCE** (for source database) or **FLAT FILE SOURCE** (for flat files txt) is used to extract data for the Staging criteria. In this step developer is able to 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 **AmExpert_Staging**.


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Extract campaign data to Staging
OLE DB Source Campaign
30 rows
OLE DB Destination CampaignStaging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

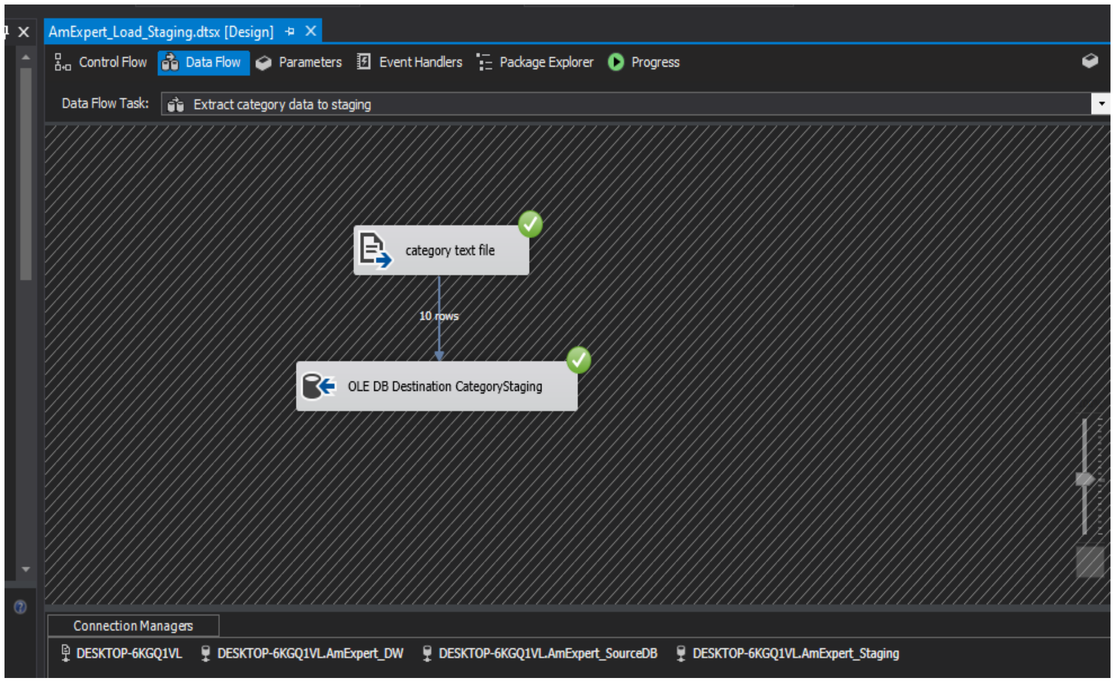
Campaign data is extracted from the Campaign table in the source database and inserted to the CampaignStaging table


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Extract item data to staging
OLE DB Source Item
250 rows
OLE DB Destination Item staging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

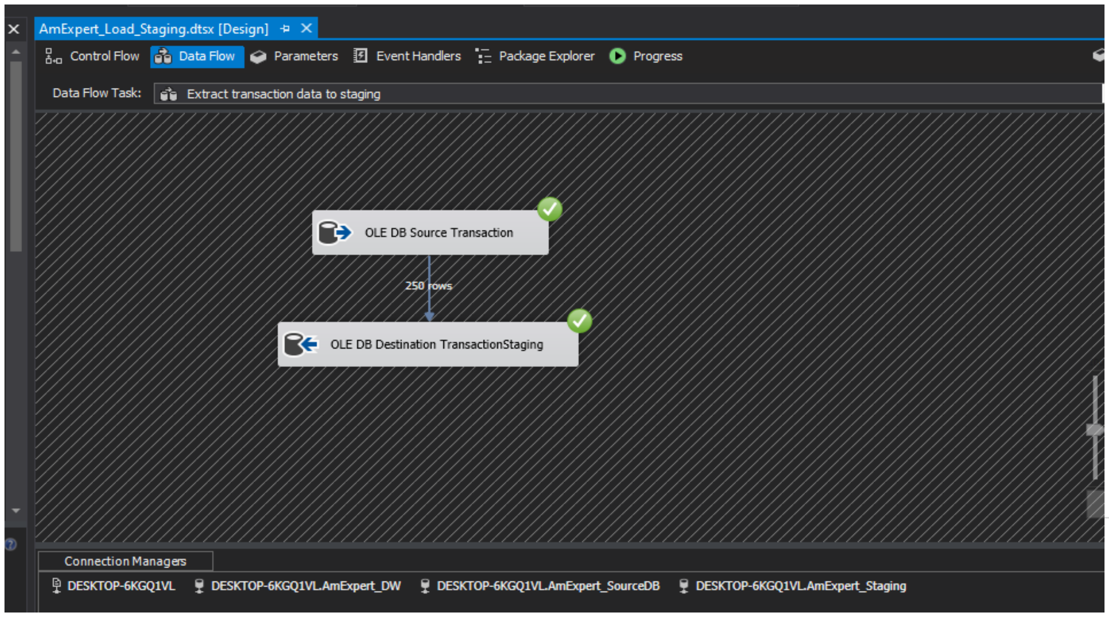
Item data is extracted from the Item table in the source database and inserted to the ItemStaging table


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task: Extract train data to staging
OLE DB Source Train
200 rows
OLE DB Destination TrainStaging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

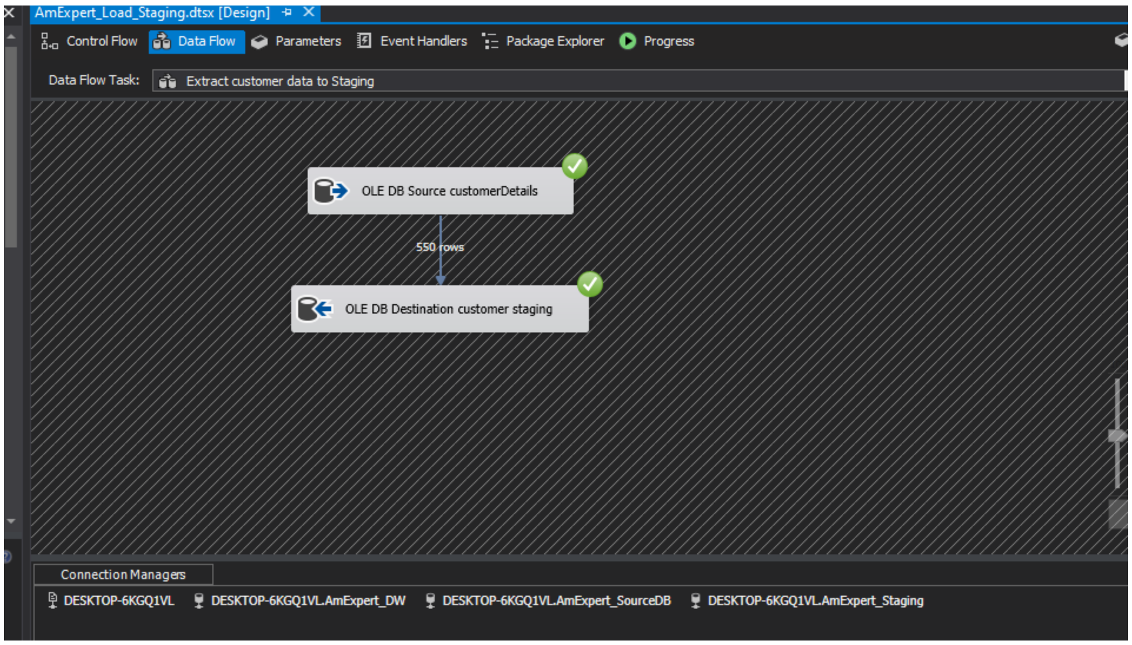
Train data is extracted from the Train table in the source database and inserted to the TrainStaging table


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task: Extract category data to staging
category text file
10 rows
OLE DB Destination CategoryStaging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

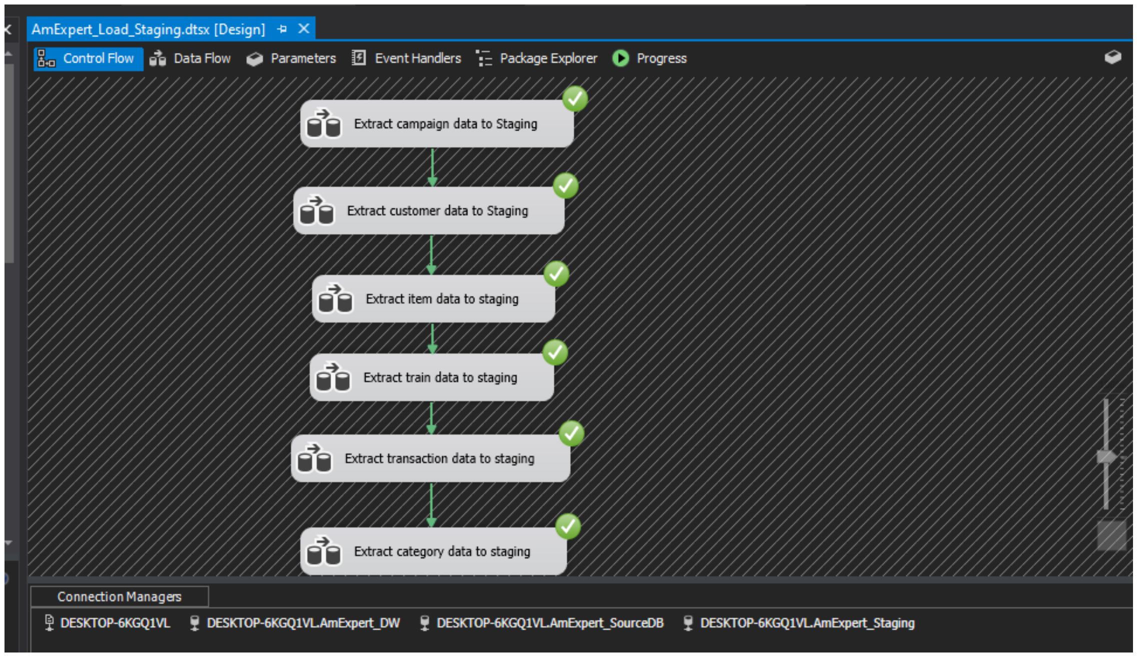
Category data is extracted from the category table in the text file and inserted to the CategoryStaging table


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task: Extract transaction data to staging
OLE DB Source Transaction
250 rows
OLE DB Destination TransactionStaging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

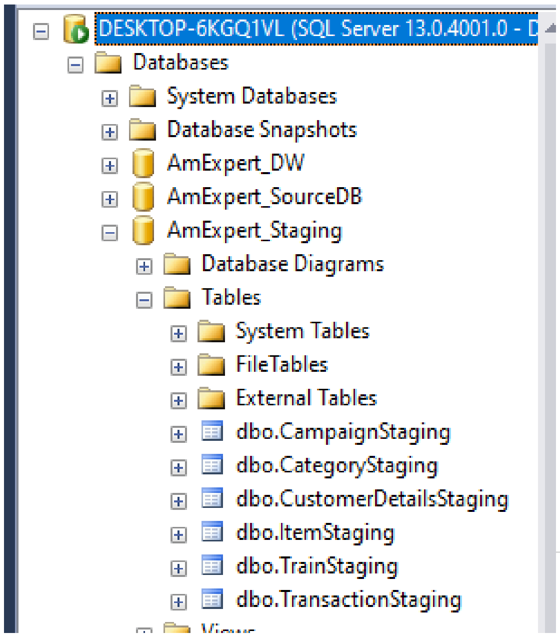
transaction data is extracted from the Transaction table in the source database and inserted to the TransactionStaging table


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task: Extract customer data to Staging
OLE DB Source customerDetails
550 rows
OLE DB Destination customer staging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

Customer data is extracted from the Customer Details table in the source database and inserted to the CustomerStaging table


AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Extract campaign data to Staging
Extract customer data to Staging
Extract item data to staging
Extract train data to staging
Extract transaction data to staging
Extract category data to staging
Connection Managers
DESKTOP-6KGQ1VL
DESKTOP-6KGQ1VL.AmExpert_DW
DESKTOP-6KGQ1VL.AmExpert_SourceDB
DESKTOP-6KGQ1VL.AmExpert_Staging

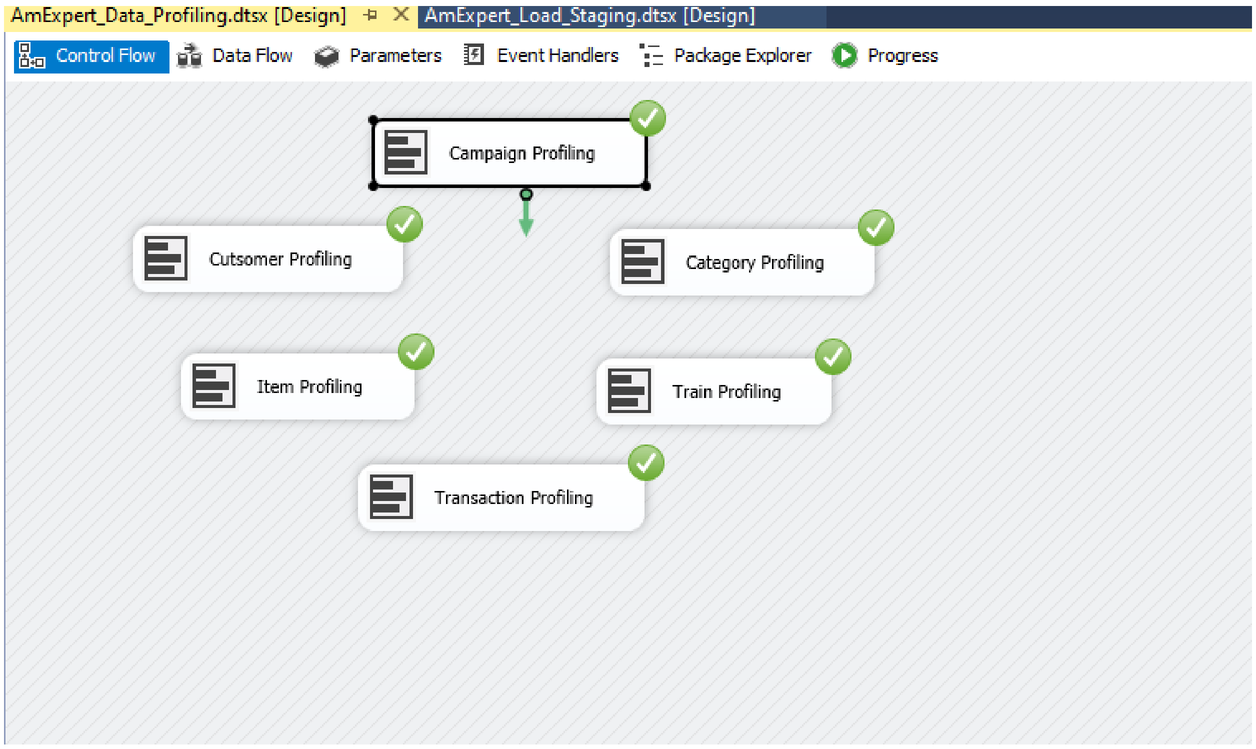
The Control Flow of 'Extract Data and Load into Staging' Step can illustrate as the give figure.



Staging Tables created and values inserted

ii. Data Profiling

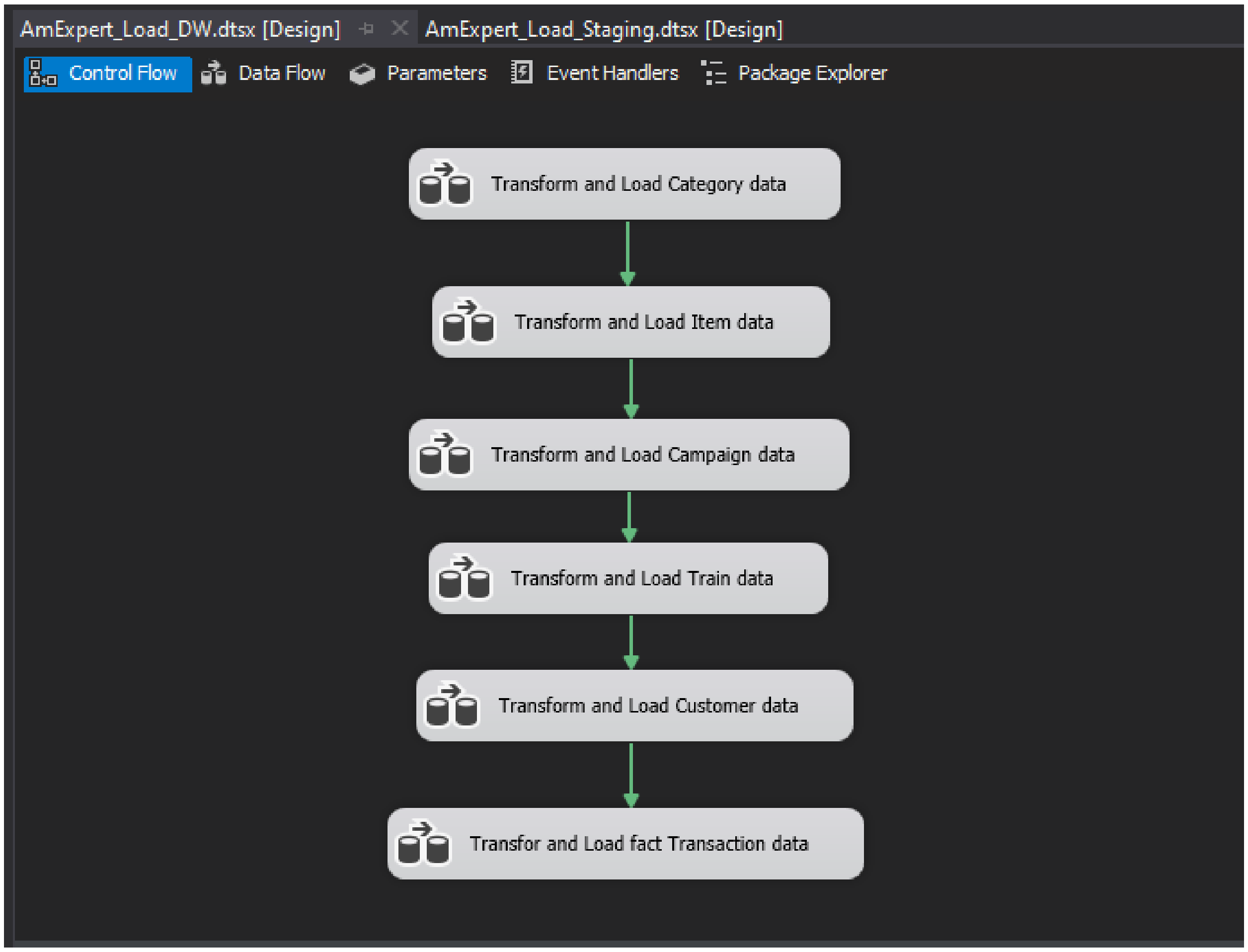
Data Profiling provides the means of analyzing large amount of data using different kind of processes. In this step, null values, repeated values and quality of the data is checked.



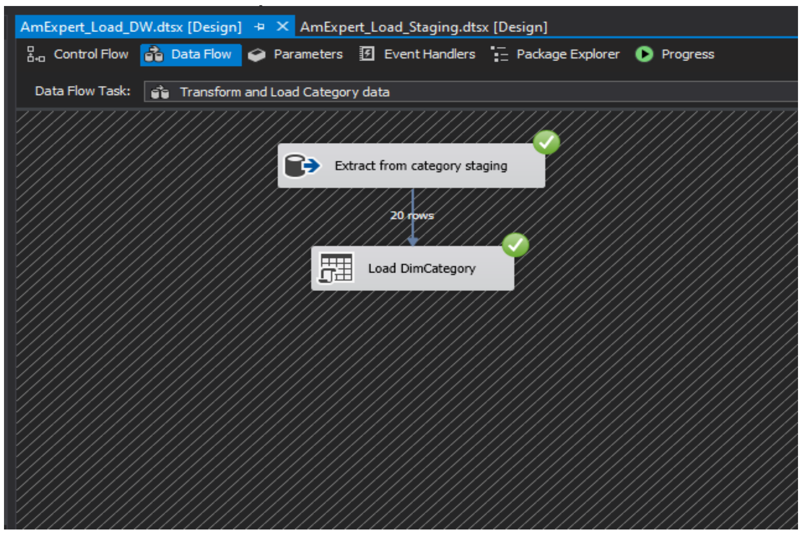
- Every staging table is profiled and saved in a selected location.
- As the figure shows, after the Staging step doing this task shows the things what the developer has to consider about the data which are stored in staging table and the developer is able to identify the issues with staging data by data profiling (such as null values).
- The given figure illustrated the complete part of Data Profiling relevant to the Staging.

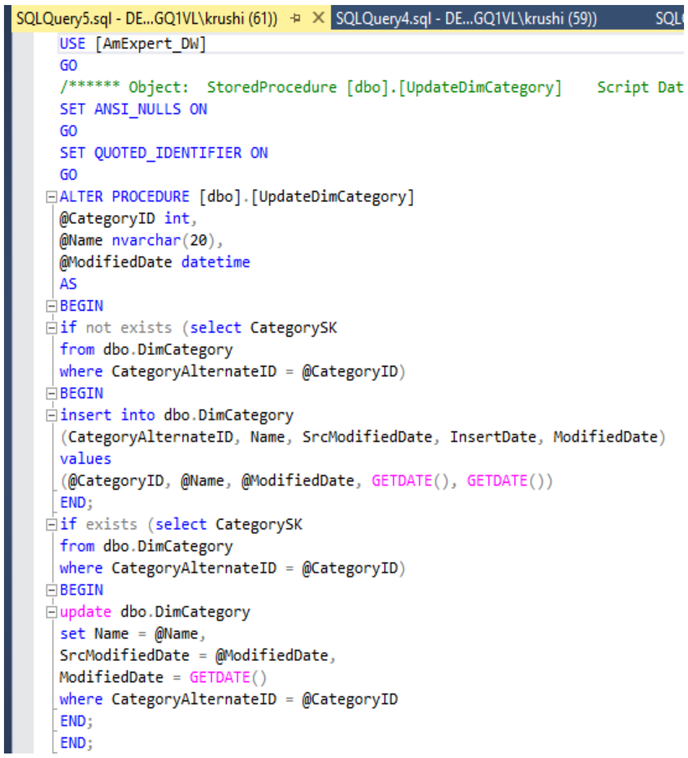
iii. Data Transformation and Loading

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


AmExpert_Load_DW.dtsx [Design]
AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Transform and Load Category data
Transform and Load Item data
Transform and Load Campaign data
Transform and Load Train data
Transform and Load Customer data
Transfor and Load fact Transaction data

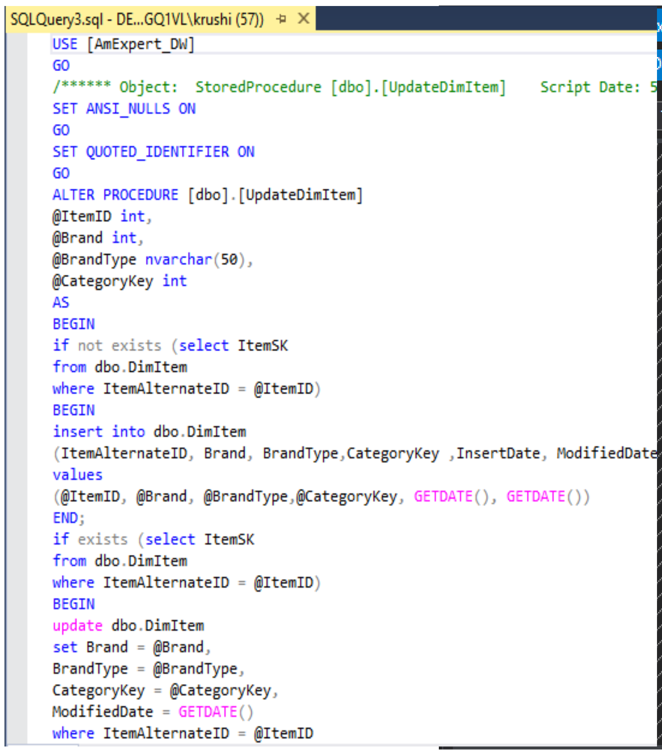
In this step, the Dimension Tables created in AmExpert_DW are loaded with the data of relevant staging tables.


AmExpert_Load_DW.dtsx [Design]
AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Transform and Load Category data
Extract from category staging
20 rows
Load DimCategory

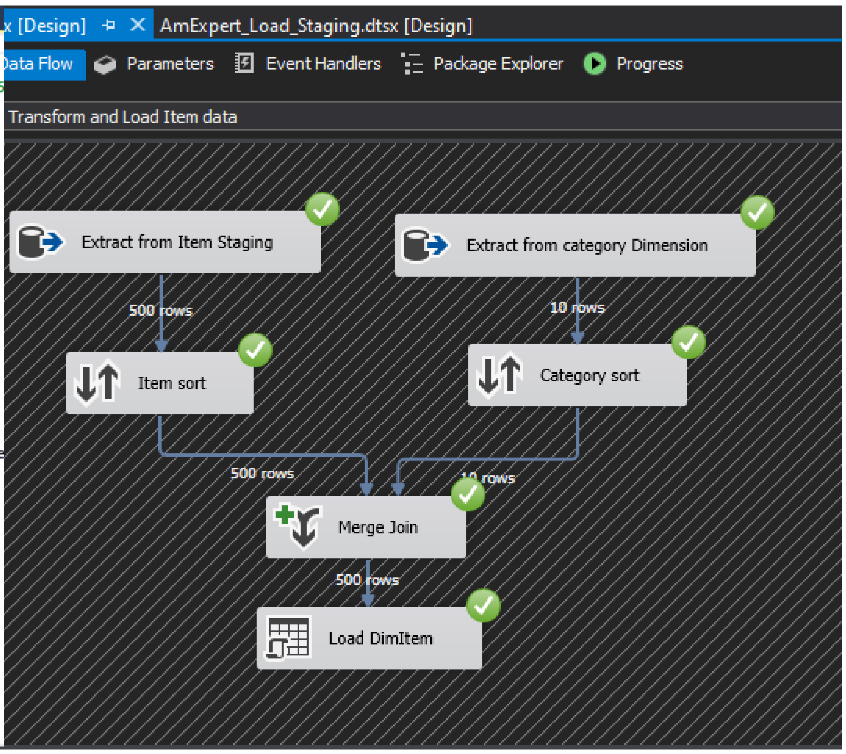

SQLQuery5.sql - DE...GQ1VL\krushi (61))
SQLQuery4.sql - DE...GQ1VL\krushi (59))

```
USE [AmExpert_DW]
GO
/****** Object:  StoredProcedure [dbo].[UpdateDimCategory]    Script Dat
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[UpdateDimCategory]
@CategoryID int,
@Name nvarchar(20),
@ModifiedDate datetime
AS
BEGIN
if not exists (select CategorySK
from dbo.DimCategory
where CategoryAlternateID = @CategoryID)
BEGIN
insert into dbo.DimCategory
(CategoryAlternateID, Name, SrcModifiedDate, InsertDate, ModifiedDate)
values
(@CategoryID, @Name, @ModifiedDate, GETDATE(), GETDATE())
END;
if exists (select CategorySK
from dbo.DimCategory
where CategoryAlternateID = @CategoryID)
BEGIN
update dbo.DimCategory
set Name = @Name,
SrcModifiedDate = @ModifiedDate,
ModifiedDate = GETDATE()
where CategoryAlternateID = @CategoryID
END;
END;
```

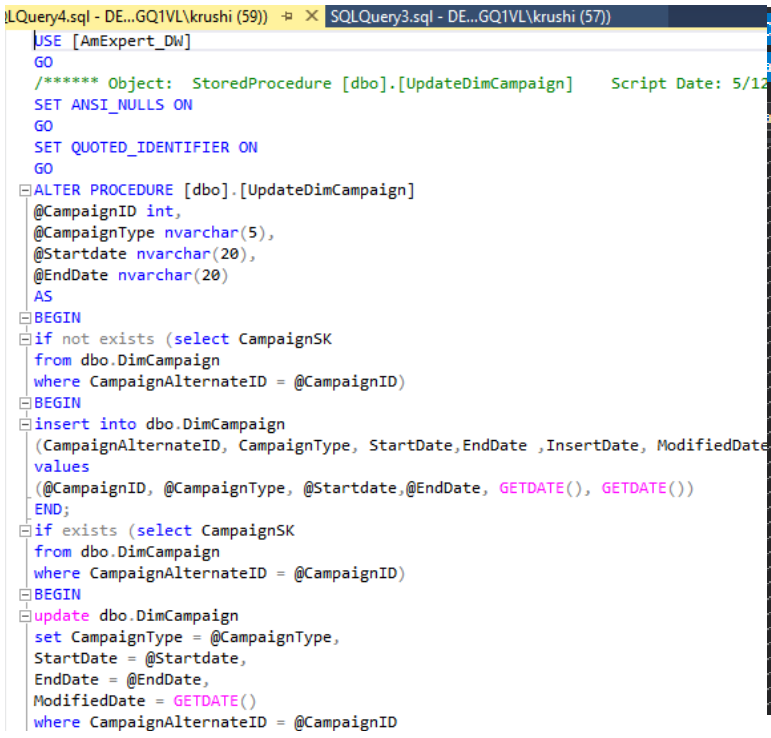
- Category data is loaded to the DimCategory
- UpdateDimCategory procedure is used to check whether the data inserted or not.



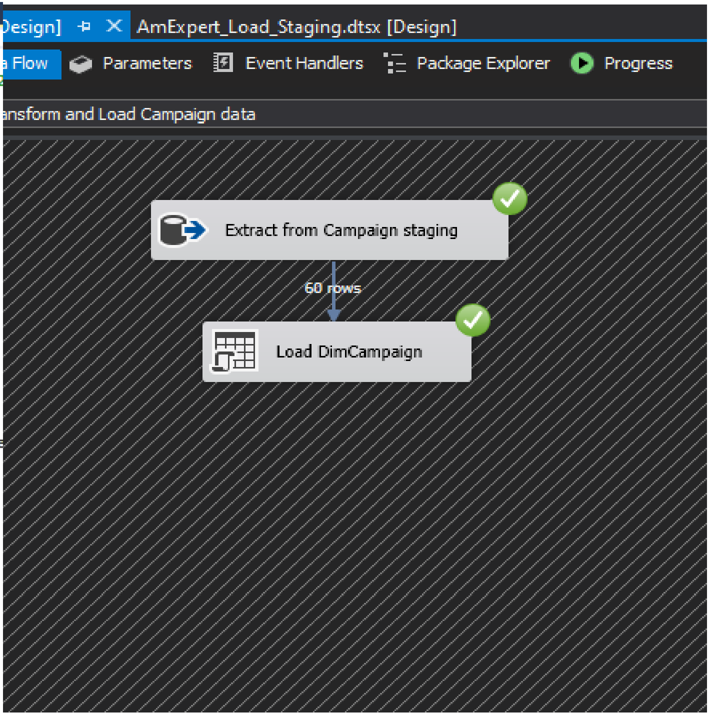
```
USE [AmExpert_DW]
GO
/****** Object:  StoredProcedure [dbo].[UpdateDimItem]    Script Date: 5
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[UpdateDimItem]
@ItemID int,
@Brand int,
@BrandType nvarchar(50),
@CategoryKey int
AS
BEGIN
if not exists (select ItemSK
from dbo.DimItem
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
```



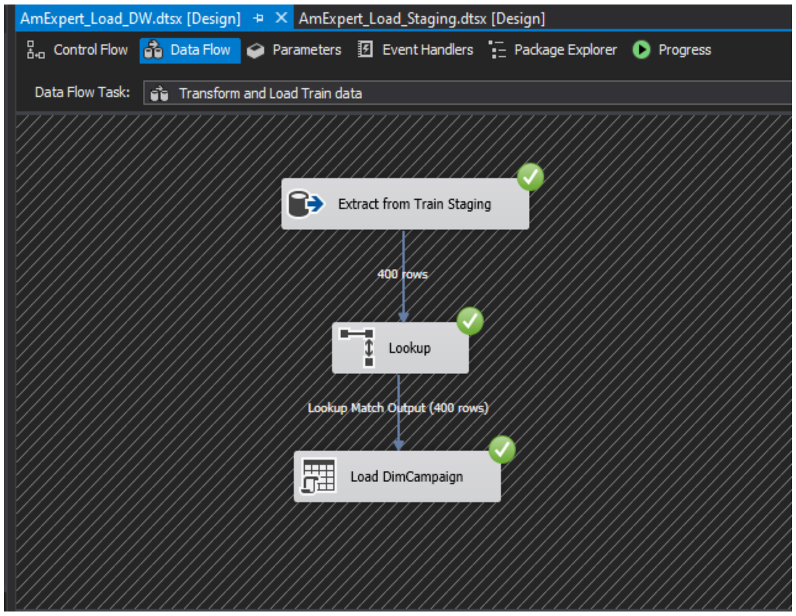
- UpdateDimItem procedure is used to check whether the data inserted or not.
- Sort and merge transformation tasks are used.
- Item data sorted according to the category ID , and category data extracted from the category dimensional table and sorted according to the category ID and merged by Mergr Join component.
- Item data is loaded to the DimItem table.

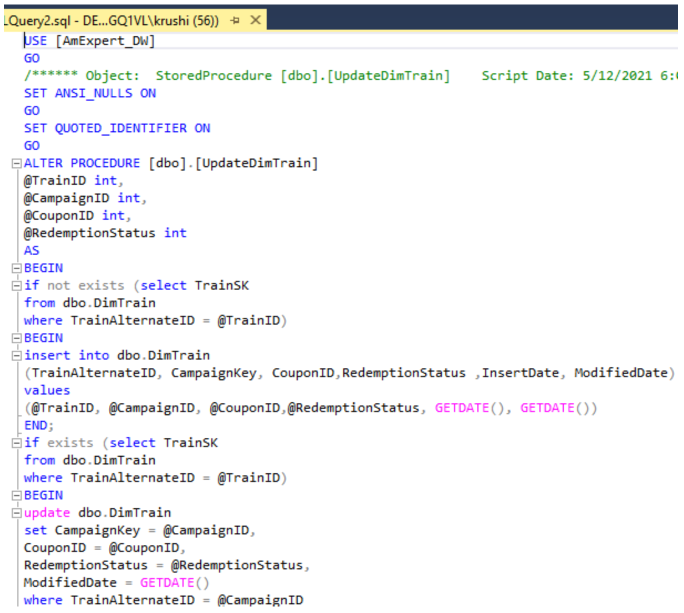

LQuery4.sql - DE...GQ1VL\krushi (59))
SQLQuery3.sql - DE...GQ1VL\krushi (57))

```sql
USE [AmExpert_DW]
GO
/****** Object:  StoredProcedure [dbo].[UpdateDimCampaign]    Script Date: 5/12
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[UpdateDimCampaign]
@CampaignID int,
@CampaignType nvarchar(5),
@Startdate nvarchar(20),
@EndDate nvarchar(20)
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, @CampaignType, @Startdate,@EndDate, GETDATE(), GETDATE())
END;
if exists (select CampaignSK
from dbo.DimCampaign
where CampaignAlternateID = @CampaignID)
BEGIN
update dbo.DimCampaign
set CampaignType = @CampaignType,
StartDate = @Startdate,
EndDate = @EndDate,
ModifiedDate = GETDATE()
where CampaignAlternateID = @CampaignID
```


esign]
AmExpert_Load_Staging.dtsx [Design]
Flow
Parameters
Event Handlers
Package Explorer
Progress
ansform and Load Campaign data
Extract from Campaign staging
60 rows
Load DimCampaign

- ➢ Campaign data is loaded to the DimCampaign table.
- ➢ UpdateDimCampaign procedure is used to check whether the data inserted or not.


AmExpert_Load_DW.dtsx [Design]
AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Transform and Load Train data
Extract from Train Staging
400 rows
Lookup
Lookup Match Output (400 rows)
Load DimCampaign

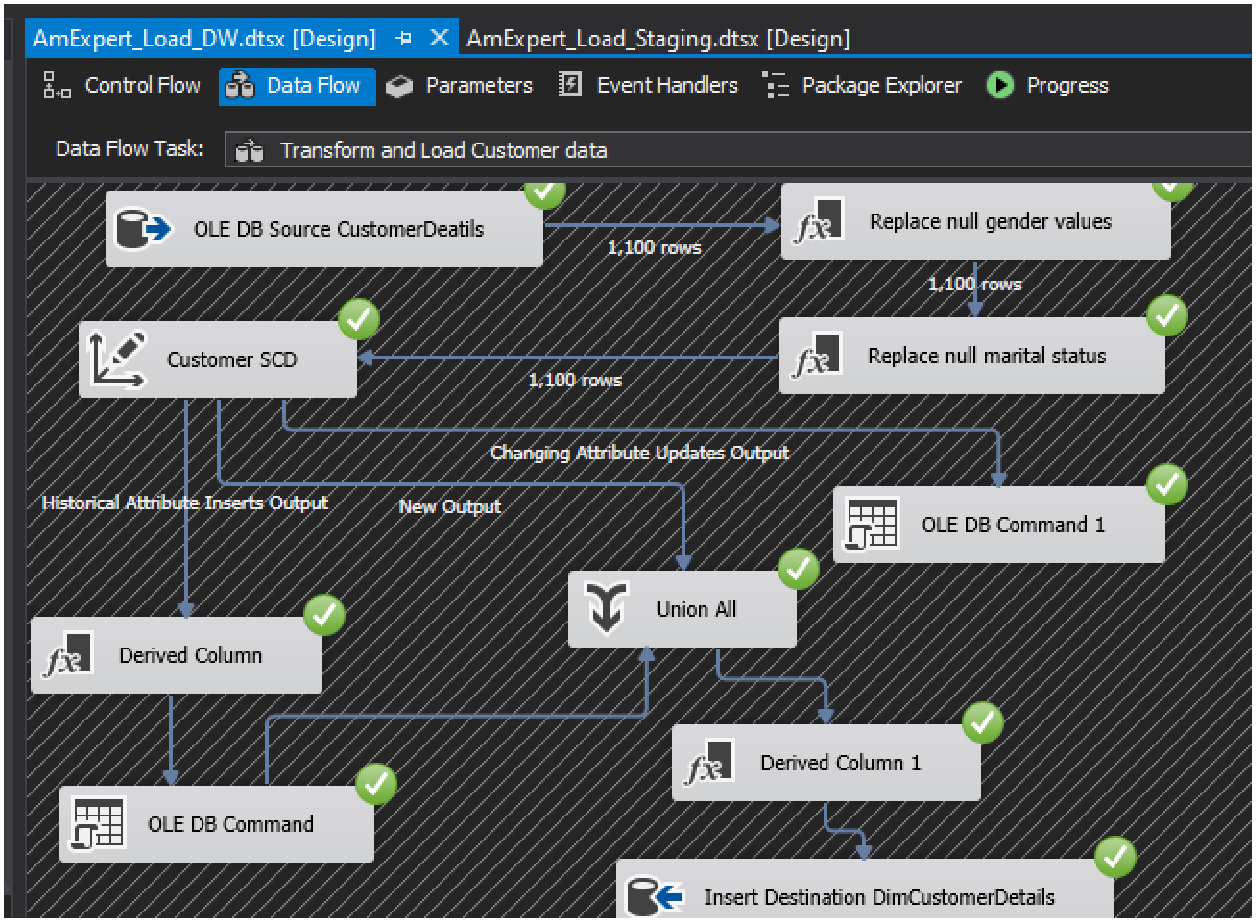


```sql
USE [AmExpert_DW]
GO
/****** Object:  StoredProcedure [dbo].[UpdateDimTrain]    Script Date: 5/12/2021 6:
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[UpdateDimTrain]
@TrainID int,
@CampaignID int,
@CouponID int,
@RedemptionStatus int
AS
BEGIN
if not exists (select TrainSK
from dbo.DimTrain
where TrainAlternateID = @TrainID)
BEGIN
insert into dbo.DimTrain
(TrainAlternateID, CampaignKey, CouponID,RedemptionStatus ,InsertDate, ModifiedDate)
values
(@TrainID, @CampaignID, @CouponID,@RedemptionStatus, GETDATE(), GETDATE())
END;
if exists (select TrainSK
from dbo.DimTrain
where TrainAlternateID = @TrainID)
BEGIN
update dbo.DimTrain
set CampaignKey = @CampaignID,
CouponID = @CouponID,
RedemptionStatus = @RedemptionStatus,
ModifiedDate = GETDATE()
where TrainAlternateID = @CampaignID
```

- ➢ LookUp transformation task is used.
- ➢ Train data is loaded to the DimTrain table.
- ➢ UpdateDimTrain procedure is used to check whether the data inserted or not.

Loading Slowly Changing Dimension

- DimCustomerDetails 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.


AmExpert_Load_DW.dtsx [Design]
AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Transform and Load Customer data
OLE DB Source CustomerDeatils
1,100 rows
Replace null gender values
1,100 rows
Replace null marital status
Customer SCD
1,100 rows
Changing Attribute Updates Output
Historical Attribute Inserts Output
New Output
OLE DB Command 1
Union All
Derived Column
Derived Column 1
OLE DB Command
Insert Destination DimCustomerDetails

- ➢ 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.
- ➢ As mentioned earlier under assumptions, customer details were considered as slowly changing details.
- ➢ The below mentioned columns were set as changing attributes:
    1. Marital status (changing)
    2. Phone number (changing)
    3. Age range (historical)
    4. City (historical)

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

Load data to Fact table

- The final step of Transformation & Loading is load data to fact table. According to the dimensional model, TransactionStaging table is used to insert values into DimTransaction table.
- FactTransaction table has one date key which are related to Date Dimension as TransactionDateKey.
- After loading to all the dimensions, lastly data was loaded to the fact table. The below steps were followed:

1. Data extracted from the customer transaction staging

2. Join operation is done for the date using look up.

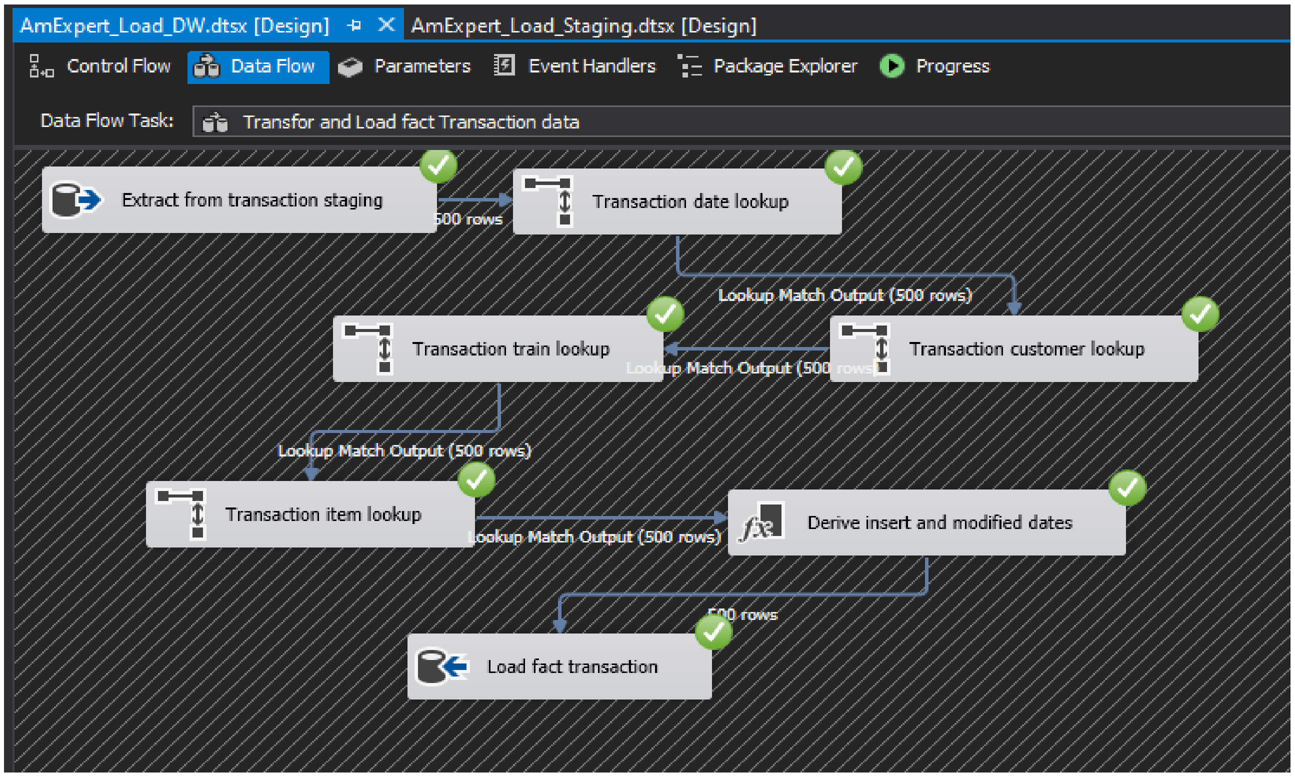
3. Join operation is done for the customer using look up.

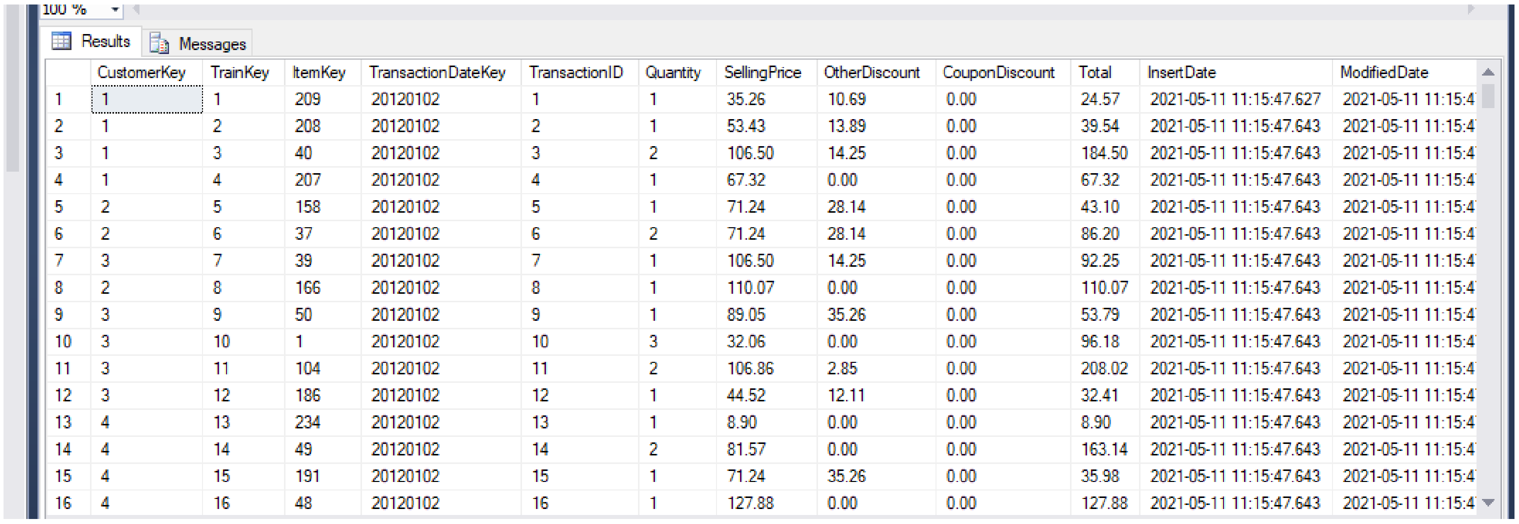
4. Join operation is done for the train using look up.

5. Join operation is done for the Item using look up.

6. insert and modified date were derived.

7. Fact details loaded to the DimTransaction table.


AmExpert_Load_DW.dtsx [Design]
AmExpert_Load_Staging.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Transfor and Load fact Transaction data
Extract from transaction staging
500 rows
Transaction date lookup
Lookup Match Output (500 rows)
Transaction customer lookup
Lookup Match Output (500 rows)
Transaction train lookup
Lookup Match Output (500 rows)
Transaction item lookup
Lookup Match Output (500 rows)
Derive insert and modified dates
500 rows
Load fact transaction



	CustomerKey	TrainKey	ItemKey	TransactionDateKey	TransactionID	Quantity	SellingPrice	OtherDiscount	CouponDiscount	Total	InsertDate	ModifiedDate
1	1	1	209	20120102	1	1	35.26	10.69	0.00	24.57	2021-05-11 11:15:47.627	2021-05-11 11:15:4
2	1	2	208	20120102	2	1	53.43	13.89	0.00	39.54	2021-05-11 11:15:47.643	2021-05-11 11:15:4
3	1	3	40	20120102	3	2	106.50	14.25	0.00	184.50	2021-05-11 11:15:47.643	2021-05-11 11:15:4
4	1	4	207	20120102	4	1	67.32	0.00	0.00	67.32	2021-05-11 11:15:47.643	2021-05-11 11:15:4
5	2	5	158	20120102	5	1	71.24	28.14	0.00	43.10	2021-05-11 11:15:47.643	2021-05-11 11:15:4
6	2	6	37	20120102	6	2	71.24	28.14	0.00	86.20	2021-05-11 11:15:47.643	2021-05-11 11:15:4
7	3	7	39	20120102	7	1	106.50	14.25	0.00	92.25	2021-05-11 11:15:47.643	2021-05-11 11:15:4
8	2	8	166	20120102	8	1	110.07	0.00	0.00	110.07	2021-05-11 11:15:47.643	2021-05-11 11:15:4
9	3	9	50	20120102	9	1	89.05	35.26	0.00	53.79	2021-05-11 11:15:47.643	2021-05-11 11:15:4
10	3	10	1	20120102	10	3	32.06	0.00	0.00	96.18	2021-05-11 11:15:47.643	2021-05-11 11:15:4
11	3	11	104	20120102	11	2	106.86	2.85	0.00	208.02	2021-05-11 11:15:47.643	2021-05-11 11:15:4
12	3	12	186	20120102	12	1	44.52	12.11	0.00	32.41	2021-05-11 11:15:47.643	2021-05-11 11:15:4
13	4	13	234	20120102	13	1	8.90	0.00	0.00	8.90	2021-05-11 11:15:47.643	2021-05-11 11:15:4
14	4	14	49	20120102	14	2	81.57	0.00	0.00	163.14	2021-05-11 11:15:47.643	2021-05-11 11:15:4
15	4	15	191	20120102	15	1	71.24	35.26	0.00	35.98	2021-05-11 11:15:47.643	2021-05-11 11:15:4
16	4	16	48	20120102	16	1	127.88	0.00	0.00	127.88	2021-05-11 11:15:47.643	2021-05-11 11:15:4

- Fact details were added to the FactTransaction table.