

# Sri Lanka Institute of Information Technology



## **Data Warehousing and Business Intelligence IT3021**

B.Sc. (Hons) in Information Technology

DATA SCIENCE

# Table of Contents

<b>Declaration .....</b>	<b>3</b>
<b>Acknowledgement .....</b>	<b>4</b>
<b>1.0 Data Set Selection.....</b>	<b>5</b>
<b>2.0 Preparation of Data Source.....</b>	<b>6</b>
<b>2.1 Entity Relationship Diagram .....</b>	<b>7</b>
<b>2.2 Data Descriptions .....</b>	<b>8</b>
<b>3.0 High Level Design Architecture .....</b>	<b>10</b>
<b>4.0 Data Warehouse design and Development .....</b>	<b>12</b>
<b>4.1 Conceptual Data model.....</b>	<b>12</b>
<b>4.2 Relational Schema .....</b>	<b>13</b>
<b>5.0 ETL Development .....</b>	<b>15</b>
<b>5.1 Data Extraction &amp; Staging Area Loading .....</b>	<b>15</b>
<b>5.2 Data Profiling.....</b>	<b>18</b>
<b>5.3 Data Extraction and Datawarehouse Loading.....</b>	<b>19</b>
<b>1. Loading Hierarchichal dimensions. ....</b>	<b>19</b>
<b>2. Slowly Changing Dimensions .....</b>	<b>23</b>
<b>3. Fact table .....</b>	<b>27</b>

# Declaration

I declare that this project report or part of it was not a copy of document done by any organization, university and other institute or a previous student project at SLIIT and was not copied from the internet or other resources.

## Student Detail

Registration Number : IT19080840

Name in Full : K.Kovishwakarunya

Batch : Y3S1.15 Weekday

# Acknowledgement

I would like to express my sincere gratitude to the Lecturer in charge Ms.Lumini Wickramasinghe , Mr. Oshada Senaweera for the constant guidance and supervision provided to complete this assignment succesfully.

A special gratitude goes to Mr.Sheron Dinushka for guiding me with industrial best practices and knowledge while completing this assignment.

Finally, a special thanks to Sri Lanka Institute of Information Technology for taking measures to provide continous support in carrying out this specialization amidst this pandemic situation too.

# 1.0 Data Set Selection

**Data Set Name** : Predicting Coupon Redemption Feature selection

**Provided by** : kaggle.com

**Source link** : <https://www.kaggle.com/vasudeva009/predicting-coupon-redemption-feature-selection>

This is a data set provided by an established Brick and Mortar retailer that frequently conducts marketing campaign for its diverse product range.

Discount marketing and Coupon usage are very widely used promotional techniques to attract new customers and to retain & enforce loyalty of existing customers.

The data set has information on customer transactions on products and the usage of coupons provided by various campaigns to complete their transactions. The features of this data set allow visualization of data set in various perspectives such as category based, brand based and Item based too. The original data set contains nine csv files with information on Customers , Campaigns , Products in the retailer, Transactions done by customers and the rest are all summary and compilation files of the above mentioned four subject areas.

## 2.0 Preparation of Data Source

The entire data set contains various subject content in csv format. In preparation of data sources, some changes were done to create a completed data source with satisfied project criteria. In such aspect few additional details such as customer addresses and contact details were added additionally to the data set. Some changes were made to the format of the source data to convert them into text files and to convert the csv files into a source database.

Final source data used for transformation process are as follows :-

- CouponCustomerAddress.xls – Customer Address Information.
- CouponItemCategory.txt – Category of Items Information in retailer.
- Coupon\_SourceDB
  - dbo.CampaignData
  - dbo.CouponRedemp
  - dbo.customer\_Info
  - dbo.Customer\_ItemTransaction
  - dbo.Items
  - dbo.productBrand

According to the final data set, data related to customers involved in transactions with the company is recorded in dbo.customer\_Info table. The location address of the customer is brought as a separate source namely CouponCustomerAddress.xls. The various campaigns initiated for this marketing process is recorded in dbo.CampaignData table. Coupons offered by campaigns for customer transactions are stored in dbo.CouponRedemp table. The products sold in the Retailer is divided into various categories and is stored in CouponItemCategory.txt, whereas each of these categories can further contain various brands and is stored in dbo.productBrand table. Ultimately each brand can have various items to be sold to the customer and is stored in dbo.Items table, respectively. The customer transactions with the retailer shop are stored in dbo.Customer\_ItemTransaction.

## 2.1 Entity Relationship Diagram

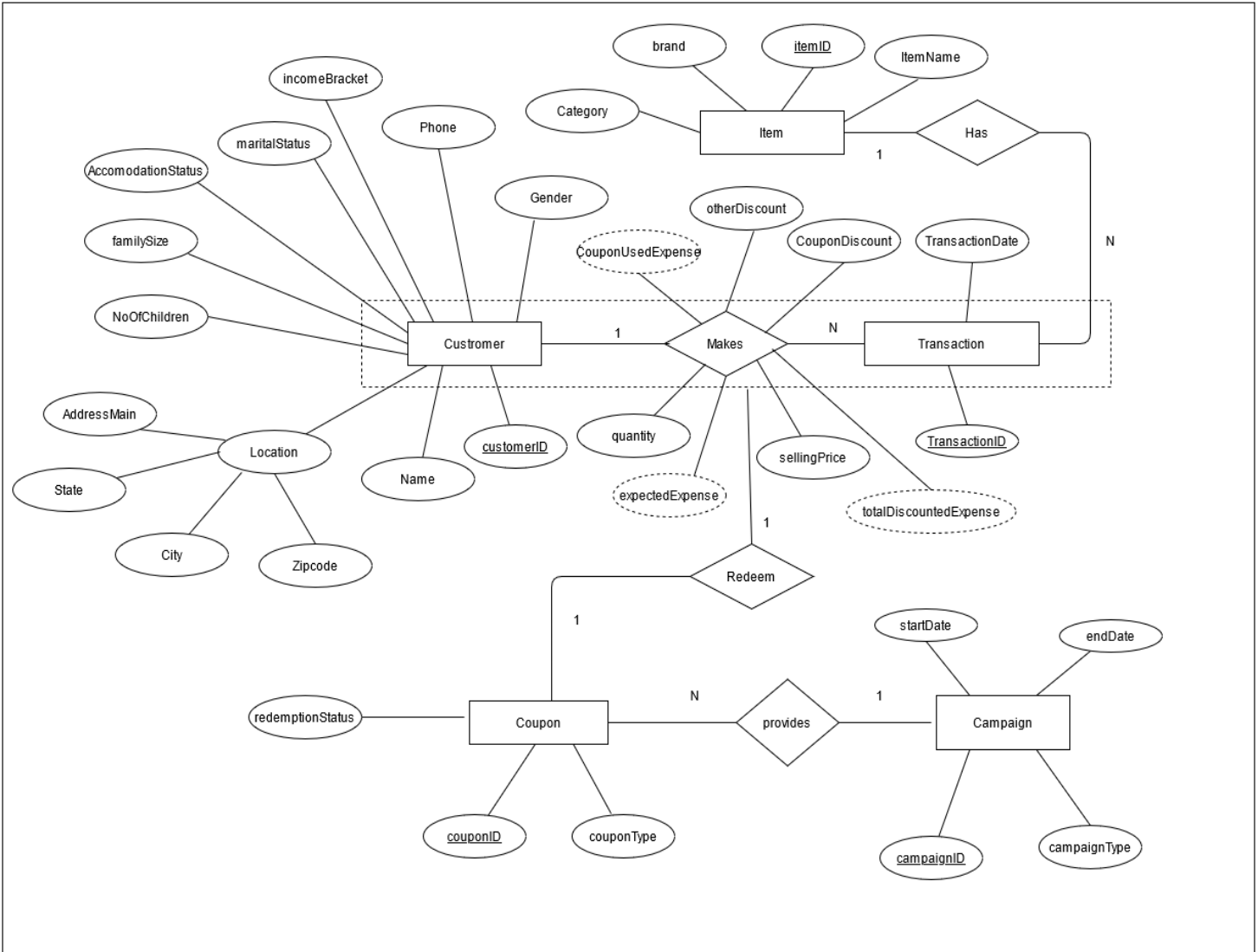


Figure 1.0 : Coupon Redemption ER diagram

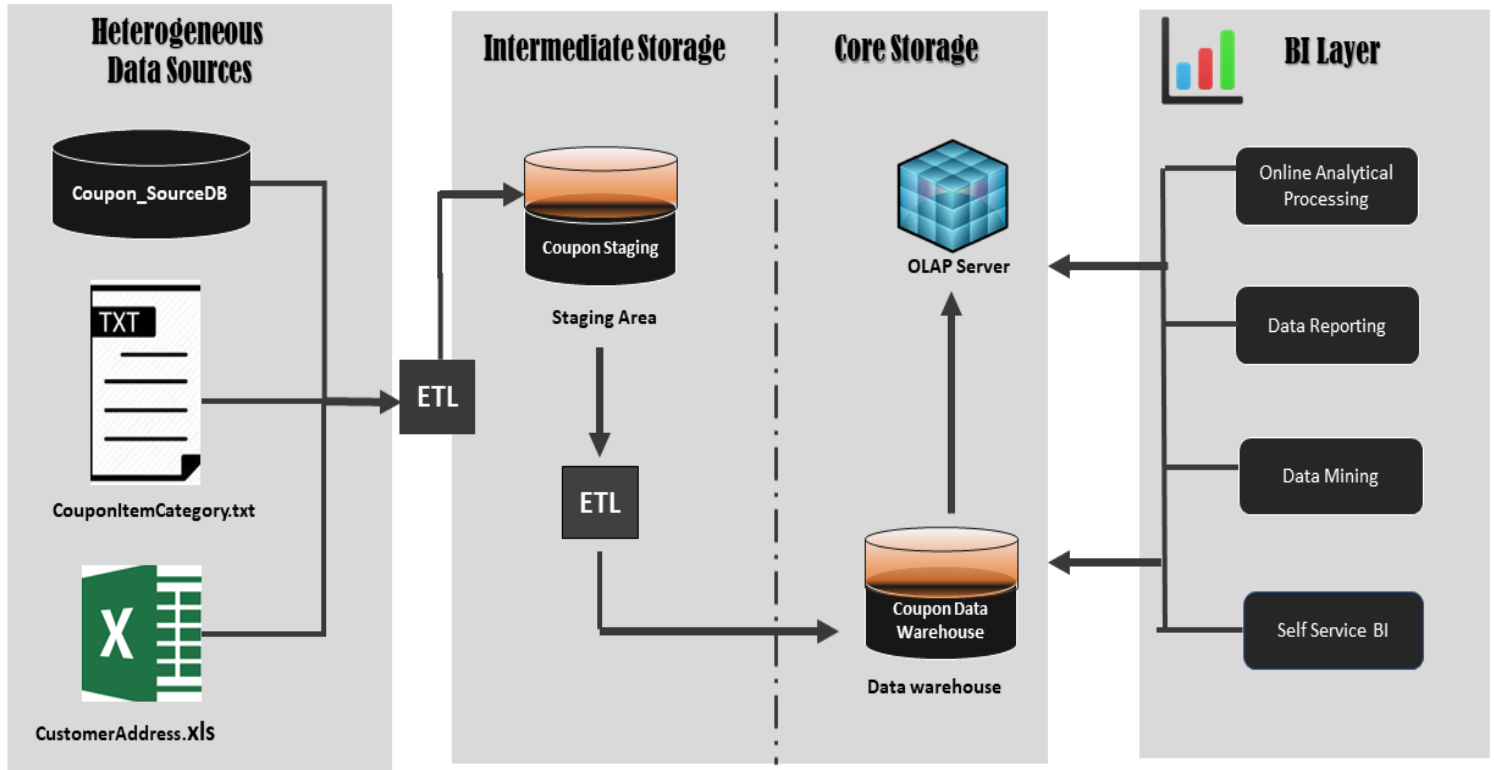
## 2.2 Data Descriptions

Source_Type	Table Name	Data																														
CouponItemCategory.txt	CouponItem category	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>CategoryID</td><td>varchar(50)</td><td>Unique Category ID.</td></tr> <tr> <td>CategoryName</td><td>varchar(50)</td><td>Name of Category.</td></tr> </table>	ColumnName	Datatype	Description	CategoryID	varchar(50)	Unique Category ID.	CategoryName	varchar(50)	Name of Category.																					
ColumnName	Datatype	Description																														
CategoryID	varchar(50)	Unique Category ID.																														
CategoryName	varchar(50)	Name of Category.																														
CouponCustomerAddresses.xls	CouponCustomer Addresses	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>CustomerID</td><td>nvarchar(255)</td><td>Unique Customer ID.</td></tr> <tr> <td>addressMain</td><td>nvarchar(255)</td><td>Customer's main Address .</td></tr> <tr> <td>city</td><td>nvarchar(255)</td><td>Customer address city.</td></tr> <tr> <td>state</td><td>nvarchar(255)</td><td>Customer address state.</td></tr> <tr> <td>zipCode</td><td>float</td><td>Customer address zipcode.</td></tr> </table>	ColumnName	Datatype	Description	CustomerID	nvarchar(255)	Unique Customer ID.	addressMain	nvarchar(255)	Customer's main Address .	city	nvarchar(255)	Customer address city.	state	nvarchar(255)	Customer address state.	zipCode	float	Customer address zipcode.												
ColumnName	Datatype	Description																														
CustomerID	nvarchar(255)	Unique Customer ID.																														
addressMain	nvarchar(255)	Customer's main Address .																														
city	nvarchar(255)	Customer address city.																														
state	nvarchar(255)	Customer address state.																														
zipCode	float	Customer address zipcode.																														
Coupon_SourceDB	Campaign Data	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>campaign_id</td><td>varchar(10)</td><td>Unique campaign ID.</td></tr> <tr> <td>campaign_type</td><td>varchar(05)</td><td>Type of campaign(X/Y).</td></tr> <tr> <td>start_date</td><td>date</td><td>Campaign start date.</td></tr> <tr> <td>end_date</td><td>date</td><td>Campaign end date.</td></tr> </table>	ColumnName	Datatype	Description	campaign_id	varchar(10)	Unique campaign ID.	campaign_type	varchar(05)	Type of campaign(X/Y).	start_date	date	Campaign start date.	end_date	date	Campaign end date.															
ColumnName	Datatype	Description																														
campaign_id	varchar(10)	Unique campaign ID.																														
campaign_type	varchar(05)	Type of campaign(X/Y).																														
start_date	date	Campaign start date.																														
end_date	date	Campaign end date.																														
	customer_Info	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>CustomerID</td><td>nvarchar(255)</td><td>Unique Customer ID.</td></tr> <tr> <td>Name</td><td>varchar(50)</td><td>Customer Name</td></tr> <tr> <td>Gender</td><td>varchar(50)</td><td>Customer Gender(M/F)</td></tr> <tr> <td>Phone</td><td>varchar(50)</td><td>Customer contact detail</td></tr> <tr> <td>marital_status</td><td>varchar(50)</td><td>Married/Single status</td></tr> <tr> <td>Accommodation_Status</td><td>varchar(50)</td><td>Rent/Landlord status</td></tr> <tr> <td>family_size</td><td>varchar(50)</td><td>Members in family</td></tr> <tr> <td>no_of_children</td><td>varchar(50)</td><td>Number of children</td></tr> <tr> <td>Income_bracket</td><td>varchar(50)</td><td>Label encoded income bracket</td></tr> </table>	ColumnName	Datatype	Description	CustomerID	nvarchar(255)	Unique Customer ID.	Name	varchar(50)	Customer Name	Gender	varchar(50)	Customer Gender(M/F)	Phone	varchar(50)	Customer contact detail	marital_status	varchar(50)	Married/Single status	Accommodation_Status	varchar(50)	Rent/Landlord status	family_size	varchar(50)	Members in family	no_of_children	varchar(50)	Number of children	Income_bracket	varchar(50)	Label encoded income bracket
ColumnName	Datatype	Description																														
CustomerID	nvarchar(255)	Unique Customer ID.																														
Name	varchar(50)	Customer Name																														
Gender	varchar(50)	Customer Gender(M/F)																														
Phone	varchar(50)	Customer contact detail																														
marital_status	varchar(50)	Married/Single status																														
Accommodation_Status	varchar(50)	Rent/Landlord status																														
family_size	varchar(50)	Members in family																														
no_of_children	varchar(50)	Number of children																														
Income_bracket	varchar(50)	Label encoded income bracket																														
	CouponRedemp	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>Coupon_ID</td><td>varchar(10)</td><td>Unique coupon ID</td></tr> <tr> <td>Coupon_Type</td><td>varchar(20)</td><td>Type of coupon</td></tr> <tr> <td>campaign_id</td><td>varchar(10)</td><td>Referencing campaign ID</td></tr> <tr> <td>Redemption_Status</td><td>varchar(05)</td><td>1 / 0 status</td></tr> </table>	ColumnName	Datatype	Description	Coupon_ID	varchar(10)	Unique coupon ID	Coupon_Type	varchar(20)	Type of coupon	campaign_id	varchar(10)	Referencing campaign ID	Redemption_Status	varchar(05)	1 / 0 status															
ColumnName	Datatype	Description																														
Coupon_ID	varchar(10)	Unique coupon ID																														
Coupon_Type	varchar(20)	Type of coupon																														
campaign_id	varchar(10)	Referencing campaign ID																														
Redemption_Status	varchar(05)	1 / 0 status																														



	productBrand	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>Brand_ID</td><td>varchar(10)</td><td>Unique brand ID</td></tr> <tr> <td>Brand_type</td><td>varchar(20)</td><td>Type of brand</td></tr> <tr> <td>CategoryID</td><td>varchar(50)</td><td>Referencing category ID</td></tr> <tr> <td>modifiedDate</td><td>date</td><td>Category modified date</td></tr> </table>	ColumnName	Datatype	Description	Brand_ID	varchar(10)	Unique brand ID	Brand_type	varchar(20)	Type of brand	CategoryID	varchar(50)	Referencing category ID	modifiedDate	date	Category modified date																											
ColumnName	Datatype	Description																																										
Brand_ID	varchar(10)	Unique brand ID																																										
Brand_type	varchar(20)	Type of brand																																										
CategoryID	varchar(50)	Referencing category ID																																										
modifiedDate	date	Category modified date																																										
	Items	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>ItemID</td><td>varchar(10)</td><td>Unique Item ID</td></tr> <tr> <td>ItemName</td><td>varchar(20)</td><td>Name of Item</td></tr> <tr> <td>BrandID</td><td>varchar(10)</td><td>Referncing brand ID</td></tr> <tr> <td>ModifiedDate</td><td>date</td><td>Brand modified date</td></tr> </table>	ColumnName	Datatype	Description	ItemID	varchar(10)	Unique Item ID	ItemName	varchar(20)	Name of Item	BrandID	varchar(10)	Referncing brand ID	ModifiedDate	date	Brand modified date																											
ColumnName	Datatype	Description																																										
ItemID	varchar(10)	Unique Item ID																																										
ItemName	varchar(20)	Name of Item																																										
BrandID	varchar(10)	Referncing brand ID																																										
ModifiedDate	date	Brand modified date																																										
	Customer_Item Transaction	<table> <tr> <th>ColumnName</th><th>Datatype</th><th>Description</th></tr> <tr> <td>TransactionID</td><td>varchar(10)</td><td>Unique Transaction ID</td></tr> <tr> <td>Transaction_Date</td><td>datetime</td><td>Date of transaction</td></tr> <tr> <td>Customer_ID</td><td>nvarchar(255)</td><td>Referencing customer ID</td></tr> <tr> <td>Item_ID</td><td>varchar(10)</td><td>Referencing Item ID</td></tr> <tr> <td>Coupon_iD</td><td>varchar(10)</td><td>Referencing Coupon ID</td></tr> <tr> <td>quantity</td><td>numeric(18,0)</td><td>Quantity of item bought</td></tr> <tr> <td>selling_price</td><td>money</td><td>Price of one item</td></tr> <tr> <td>other_discount</td><td>money</td><td>Additional item discount</td></tr> <tr> <td>coupon_discount</td><td>money</td><td>Discount from coupon</td></tr> <tr> <td colspan="3">Derived Attributes</td></tr> <tr> <td>ExpectedCustomerExpense</td><td>money</td><td>Actual transaction amount</td></tr> <tr> <td>CouponUsedExpense</td><td>money</td><td>Transaction amount after coupon usage.</td></tr> <tr> <td>TotalDiscountedExpense</td><td>money</td><td>Transaction amount after coupon usage and other discount deduction.</td></tr> </table>	ColumnName	Datatype	Description	TransactionID	varchar(10)	Unique Transaction ID	Transaction_Date	datetime	Date of transaction	Customer_ID	nvarchar(255)	Referencing customer ID	Item_ID	varchar(10)	Referencing Item ID	Coupon_iD	varchar(10)	Referencing Coupon ID	quantity	numeric(18,0)	Quantity of item bought	selling_price	money	Price of one item	other_discount	money	Additional item discount	coupon_discount	money	Discount from coupon	Derived Attributes			ExpectedCustomerExpense	money	Actual transaction amount	CouponUsedExpense	money	Transaction amount after coupon usage.	TotalDiscountedExpense	money	Transaction amount after coupon usage and other discount deduction.
ColumnName	Datatype	Description																																										
TransactionID	varchar(10)	Unique Transaction ID																																										
Transaction_Date	datetime	Date of transaction																																										
Customer_ID	nvarchar(255)	Referencing customer ID																																										
Item_ID	varchar(10)	Referencing Item ID																																										
Coupon_iD	varchar(10)	Referencing Coupon ID																																										
quantity	numeric(18,0)	Quantity of item bought																																										
selling_price	money	Price of one item																																										
other_discount	money	Additional item discount																																										
coupon_discount	money	Discount from coupon																																										
Derived Attributes																																												
ExpectedCustomerExpense	money	Actual transaction amount																																										
CouponUsedExpense	money	Transaction amount after coupon usage.																																										
TotalDiscountedExpense	money	Transaction amount after coupon usage and other discount deduction.																																										

### 3.0 High Level Design Architecture



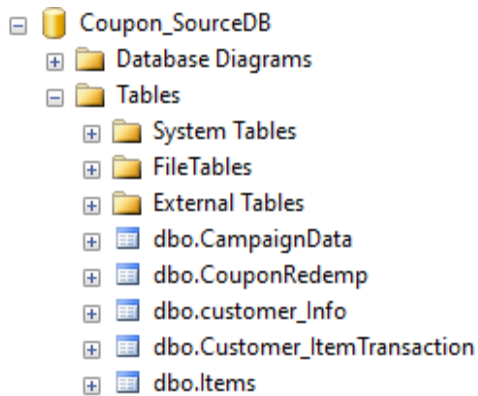
The high-level architecture solution of the Brick-and-Mortar retailer company is as given above. The entire company data is provided by three separate source formats: Source database, Text file and Excel file, respectively.

As per the requirements since we do not need any real time data capturing it was decided to load the entire source data content into an intermediate staging area. Additionally any complex queries performed will not affect the source data performance by doing so. Therefore as the initial step entire source data (Coupon\_SourceDB, CustomerAddress.xls, Category.txt) information are loaded into another schema (Coupon\_Staging), the staging database which ultimately serves as a single database containing all the data from various sources compiled at one place.

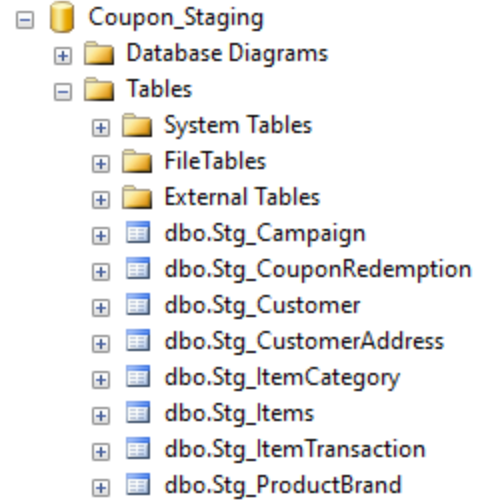
As the next step necessary transformations are done at the staging layer in order to load data from staging area to data warehouse (Coupon\_DW).

The data from data warehouse is then refreshed to create OLAP cubes which can be used by end users in order to carry out Analysis on the data set. Here data visualization can be either done through OLAP server or directly from Data warehouse which is specifically called Self service BI.

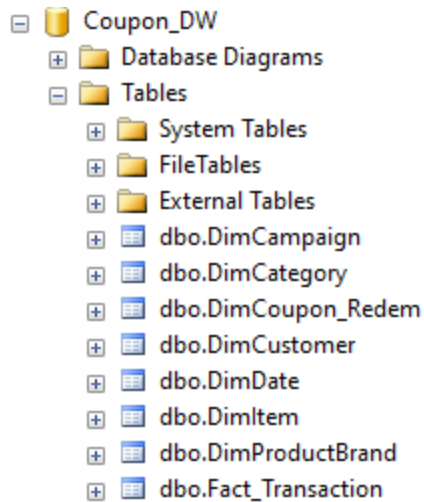
## Data Storage Snapshots from SSMS



**Figure 2.1 : Coupon Source Database**



**Figure 3.2 : Coupon Staging Database**



**Figure 3.3 : Coupon Datawarehouse**

## 4.0 Data Warehouse design and Development

### 4.1 Conceptual Data model

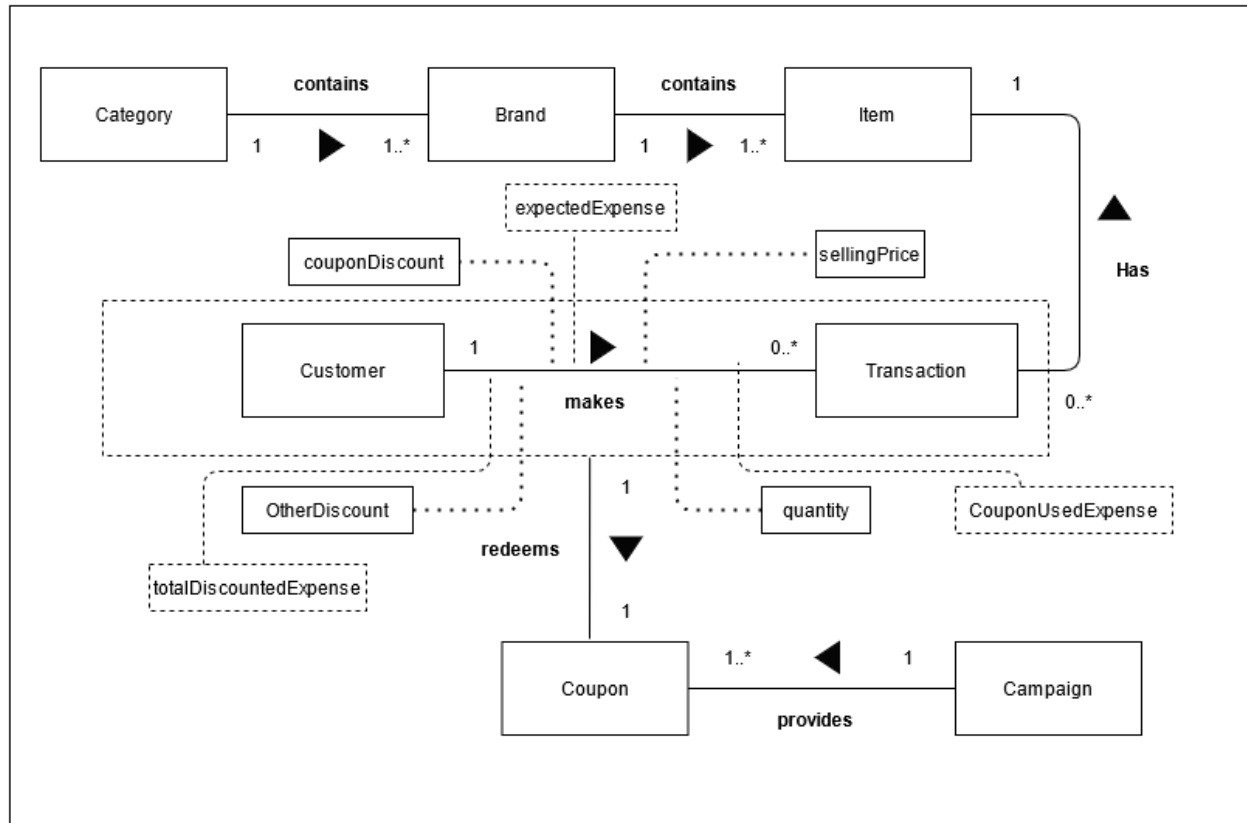


Figure 4.0 : Coupon Conceptual Diagram

A conceptual model along with an entity relationship model was designed to identify the highest-level entities and the relationships between them in the data set. Thus, the main entities identified are Category, Brand, Item, Transaction, Customer, Coupon and Campaign. Additionally, the descriptive and derived attributes are as shown in the above image via dotted lines, respectively. Ultimately by analysing this, I was able to come up with a snowflake schema for warehouse implementation.

## 4.2 Relational Schema

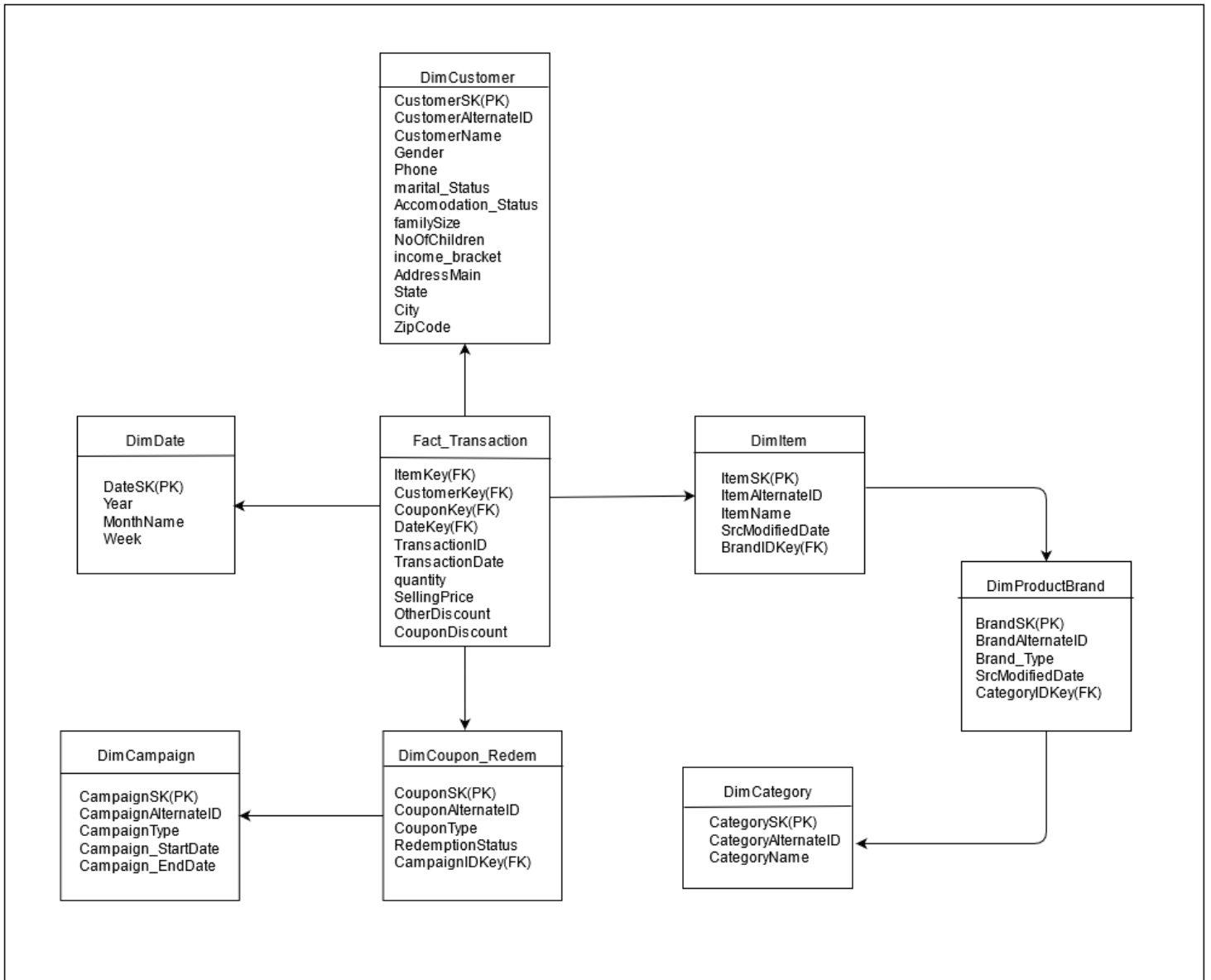


Figure 4.1 : Datawarehouse Schema

The schema designed for the Brick-and-Mortar retailer is a **snowflake schema** having one fact table and seven-dimension table including date dimension. It is observed that the entities are in a normalized form because of the snowflake design.

The dimensions are uniquely identified by the **Surrogate key**, where additionally each dimension contains the business key provided via the source data base too.

The attributes of referential integrity of the schema is indicated by the keyword **FK**.

**Hierarchical implementations** are found in this schema

1. DimProductBrand and DimCategory are hierarchical dimensions of DimItem
2. DimCustomer has a Customer address wise hierarchy.

DimCustomer table is a **slowly changing dimension** with historical attributes and changing attributes where Type 2 and Type 1 implementations are being enforced, respectively.

Transaction of a customer for a particular date is considered as the **grain** of the Fact\_Transaction fact table.

Additional **derived Calculations** are done in fact table.

1. Expected Customer Expense :  $([\text{quantity}] * [\text{SellingPrice}])$
2. Coupon used expense :  $([\text{quantity}] * [\text{SellingPrice}] + \text{CouponDiscount})$   
Note - {CouponDiscount data points are negative values}
3. Total Discounted Expense :  $(([\text{quantity}] * [\text{SellingPrice}] + [\text{quantity}] * [\text{OtherDiscount}]) + [\text{CouponDiscount}])$

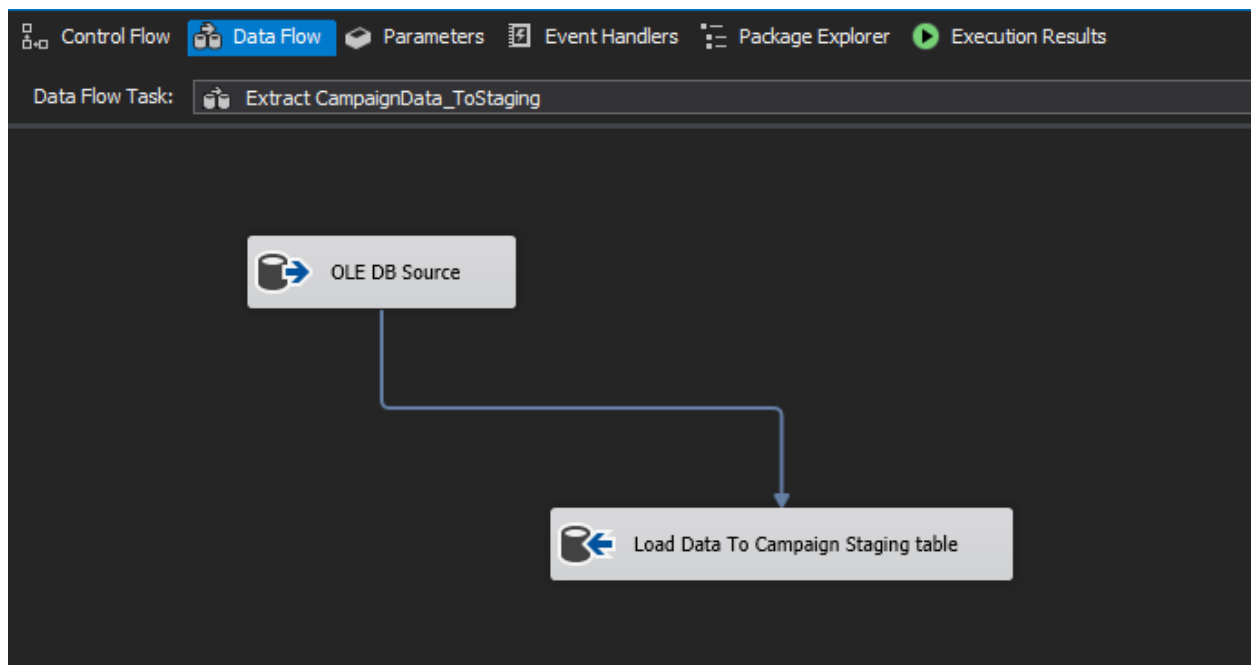
**Assumption** - (Other discount is applied to each quantity of item in a transaction, Coupon discount is applied to the entire transaction of item only once.)

## 5.0 ETL Development

### 5.1 Data Extraction & Staging Area Loading

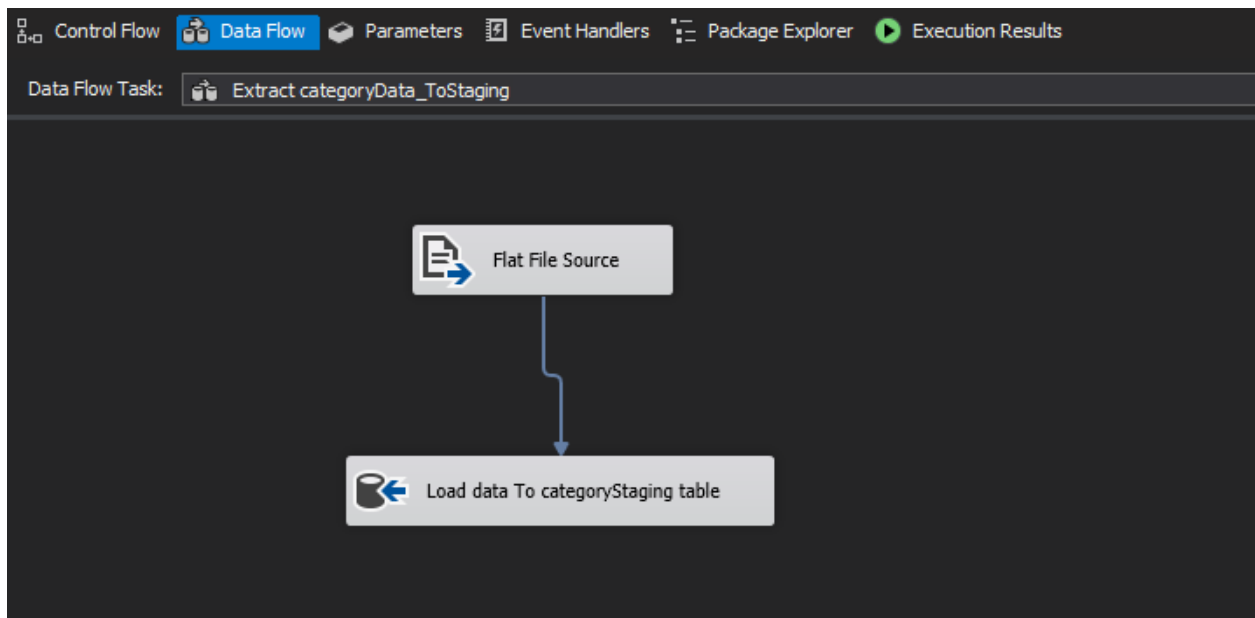
Data extraction from source data to staging area is done using SQL Server Data tool 2015 environment. The Source Database, Text file and Excel file created are used in this process. To extract data from Source Database tables, **OLE DB Source** is used in the data flow whereas text file extraction is done using **Flat file source** and Excel file extraction using **Excel Source**. From Source database to staging database lesser transformations are done, primary concern is on Data extraction and Data loading. The extracted Source data are finally loaded into the respective staging tables using **OLE DB Destination**.

Below attached images are some illustrations of different source data Extraction and Loading Process.

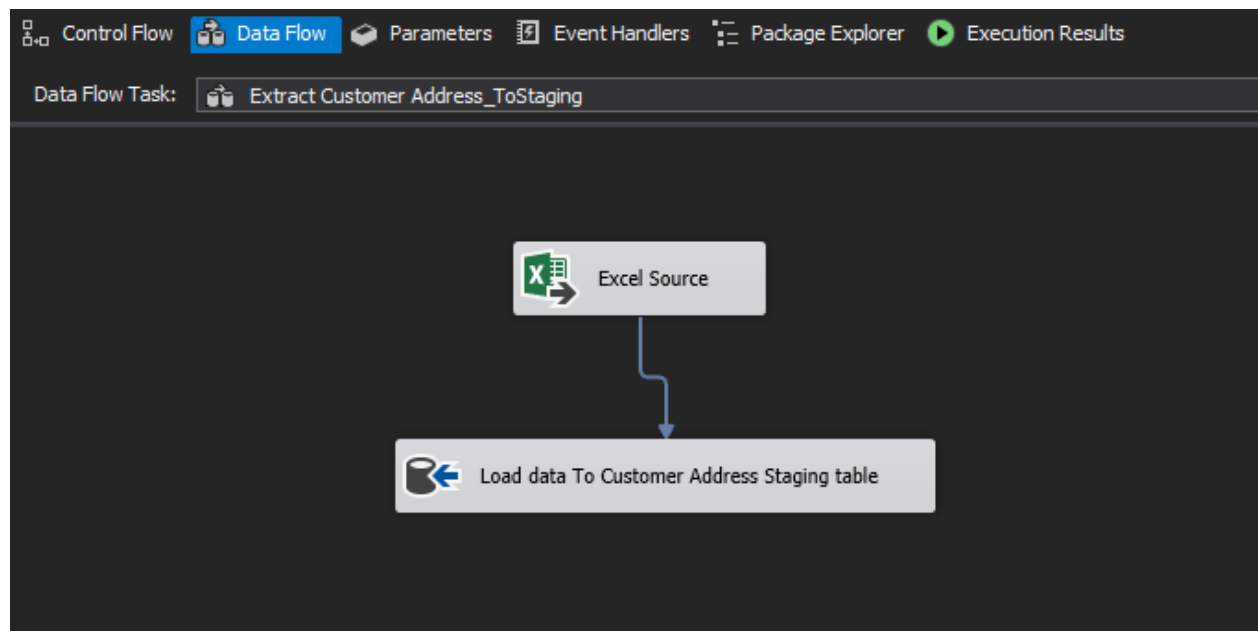


*Figure 5.0 : Data Extraction from Source database table*

Similar process to the above illustration is followed in loading all the tables of source database.

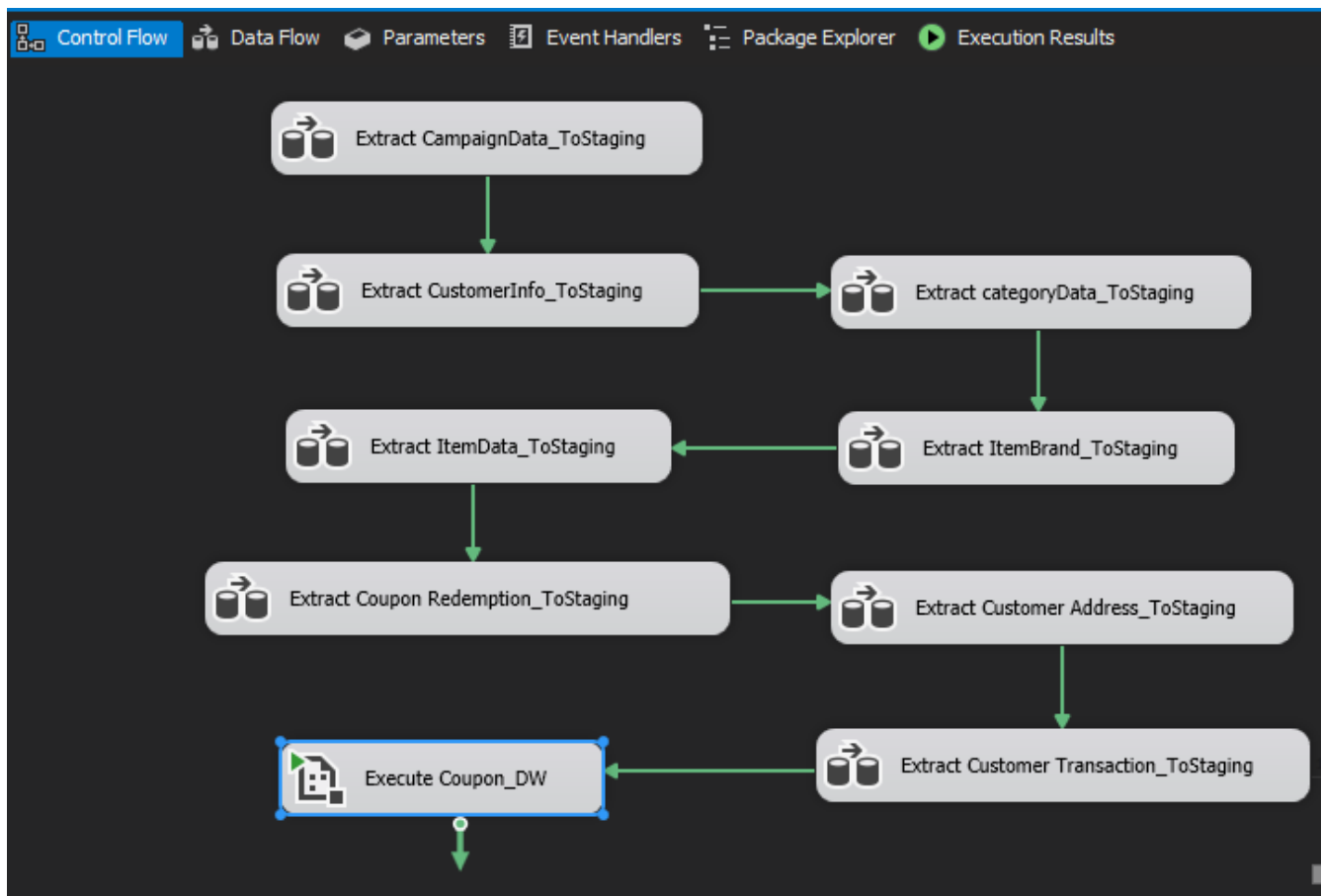


**Figure 5.1 : Data Extraction from Flat File Source**



**Figure 5.2 : Data Extraction from Excel Source**

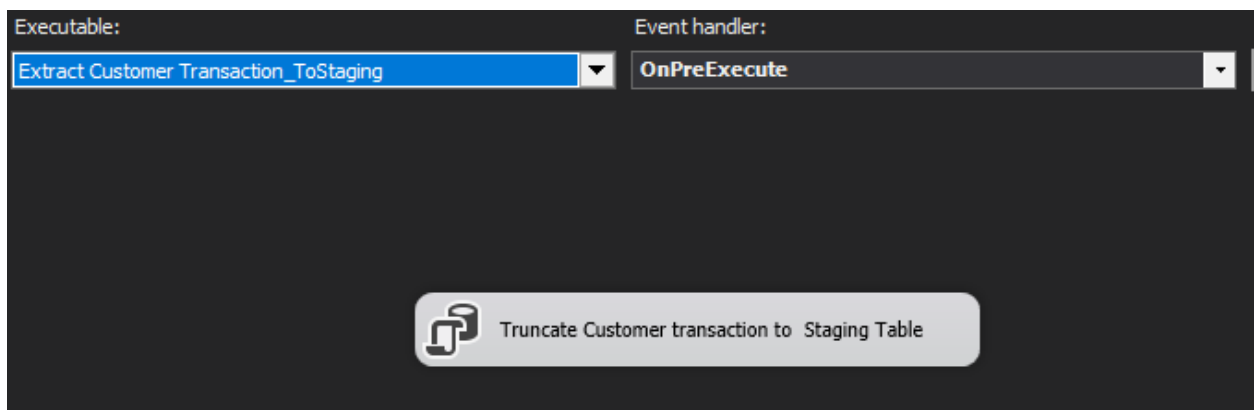




**Figure 5.3 : Coupon Staging Database Loading Orchestration**

The above order is followed in completing the orchestration of loading data to staging database. Finally to ensure that data warehouse loading is followed only after loading data to staging database an **Execute Package Task** is used.

To prevent data duplication when staging tables are repeatedly loaded , **On Pre execute event handlers** are used to truncate data before each staging table data loading is initiated.



**Figure 5.4 : Data Truncation Event Handler**

## 5.2 Data Profiling

Once the data is loaded to staging tables, data profiling can be done to analyse how the data looks like to determine the type of transformations needed to be performed on data. In this process, null value ratios, column length distributions and various other statistical information about the data can be determined.

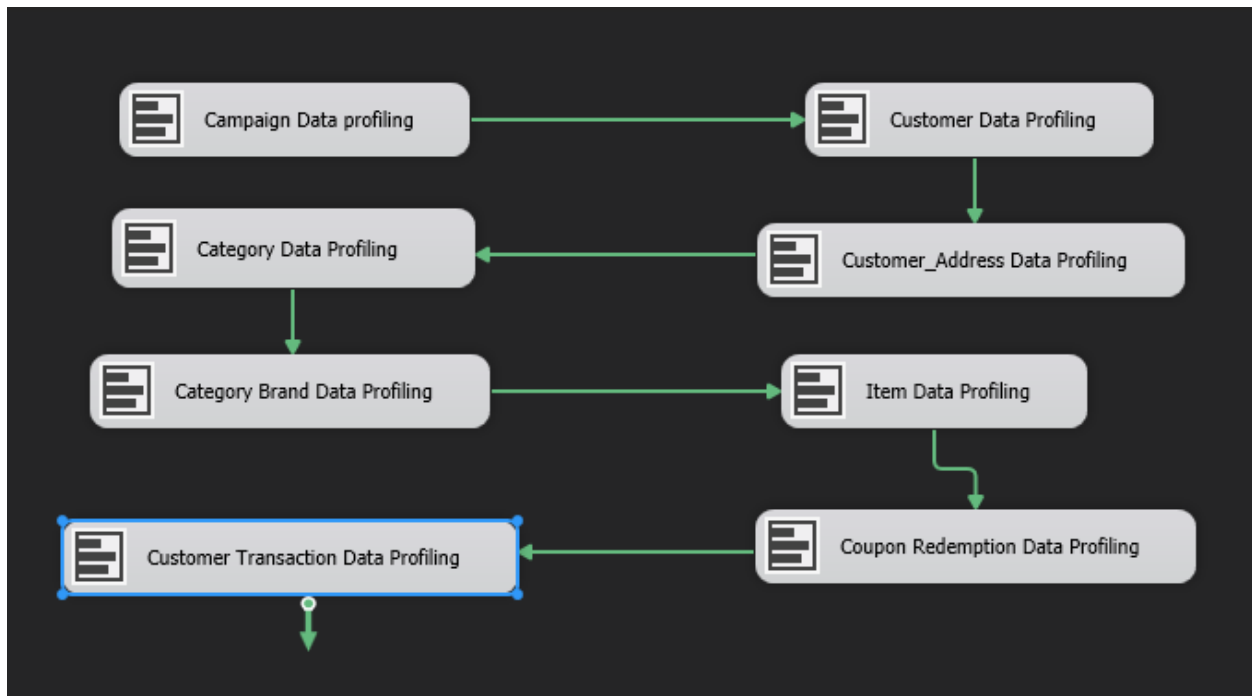


Figure 5.5 : Data Profiling on Staged tables

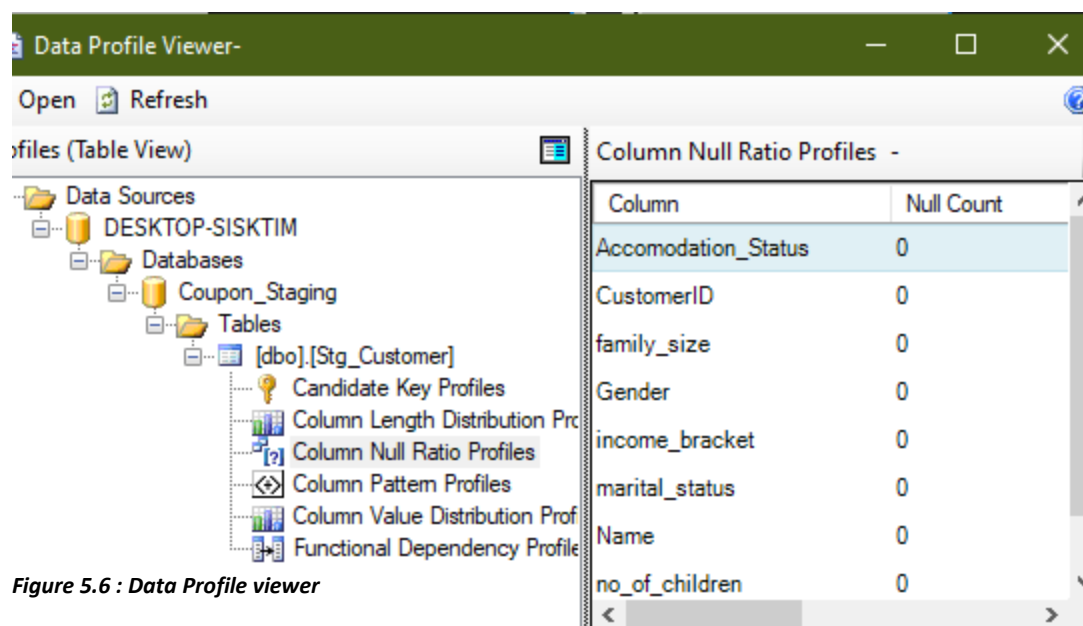


Figure 5.6 : Data Profile viewer

## 5.3 Data Extraction and Datawarehouse Loading

During the data loading process to datawarehouse , the main concern was on the order of execution of tasks. The schema was very carefully analysed in order to identify the dependencies among various tables thus deciding the order of table load executions.

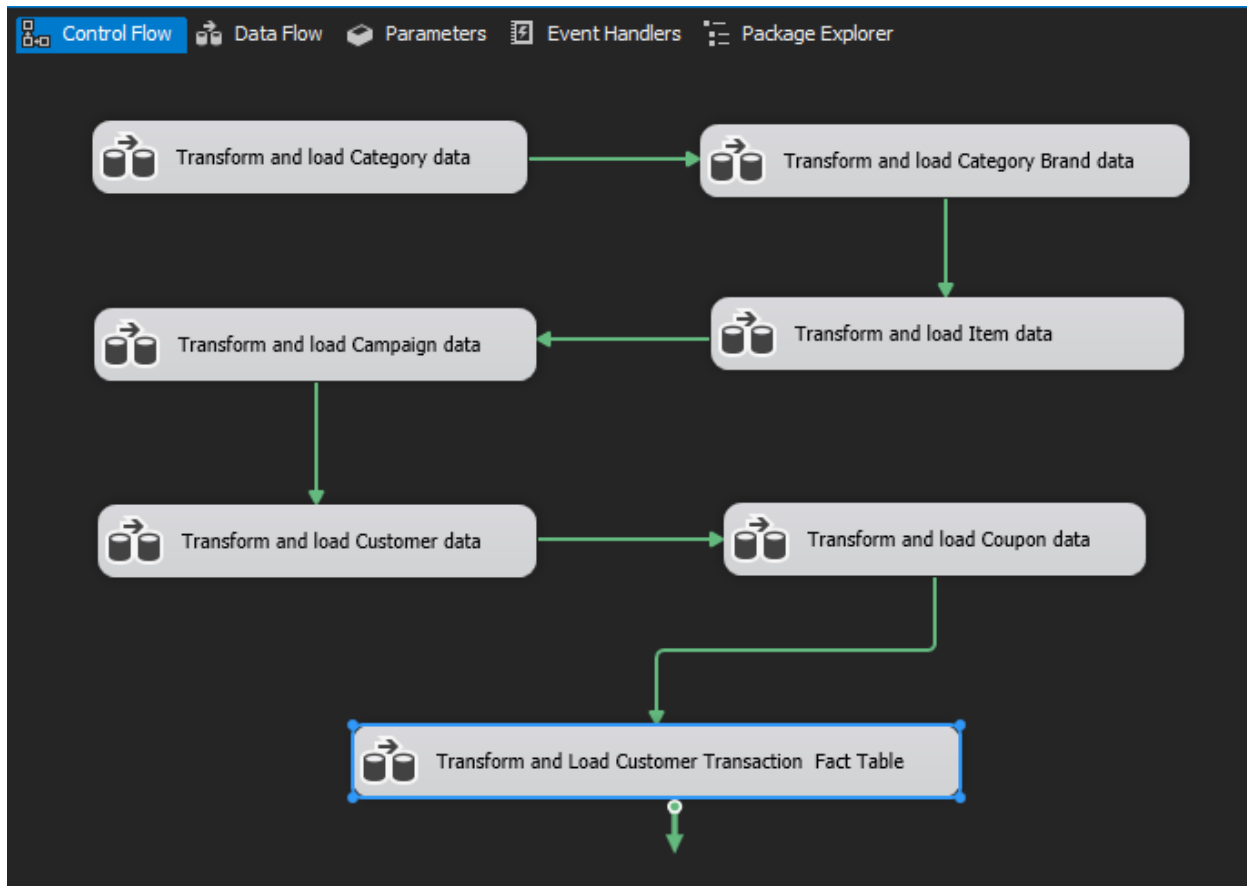
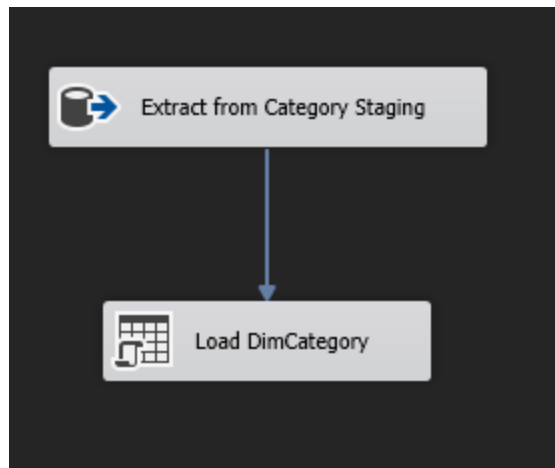


Figure 5.7 : Datawarehouse Orchestration

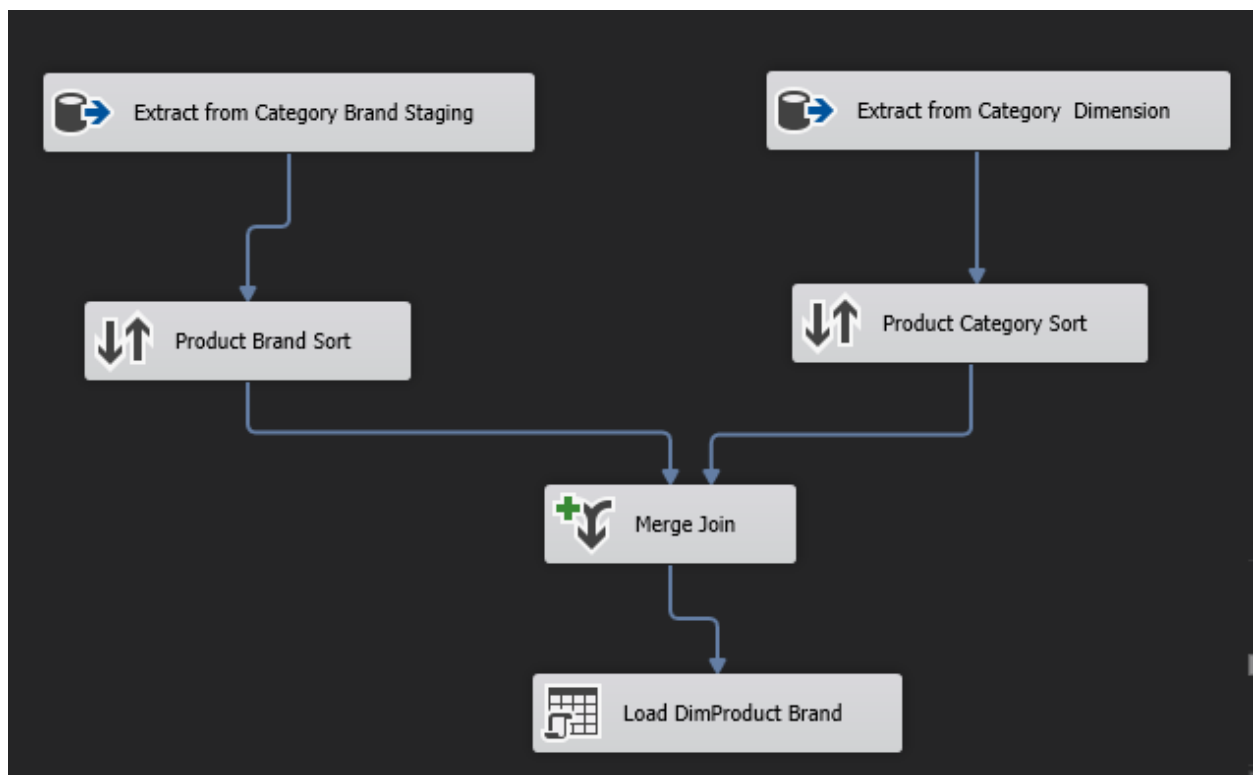
### 1. Loading Hierarchical dimensions.

By observing the Relational Schema, it is clear that ProductCategory , Brand and Item are hierarchical dimensions where Item has a reference to brand and brand has a reference to Category. Considering it Category data was loaded as the first dimension.

While loading Hierarchical dimensions **lookups, sort and merge join components** were used.



**Figure 5.8 : LoadCategoryDim**



**Figure 5.9 : LoadProductBrandDim**

While loading ProductBrand Data, ProductBrand table was sorted using the CategoryID and similarly CategoryDimension was extracted and sorted by CategoryAlternateID and was ultimately merged using merge Join component and then loaded to DimProductBrand.

**Assumption** – We load ProductBrands even though it does not contain Categories respectively therefore LEFT OUTER JOIN is used

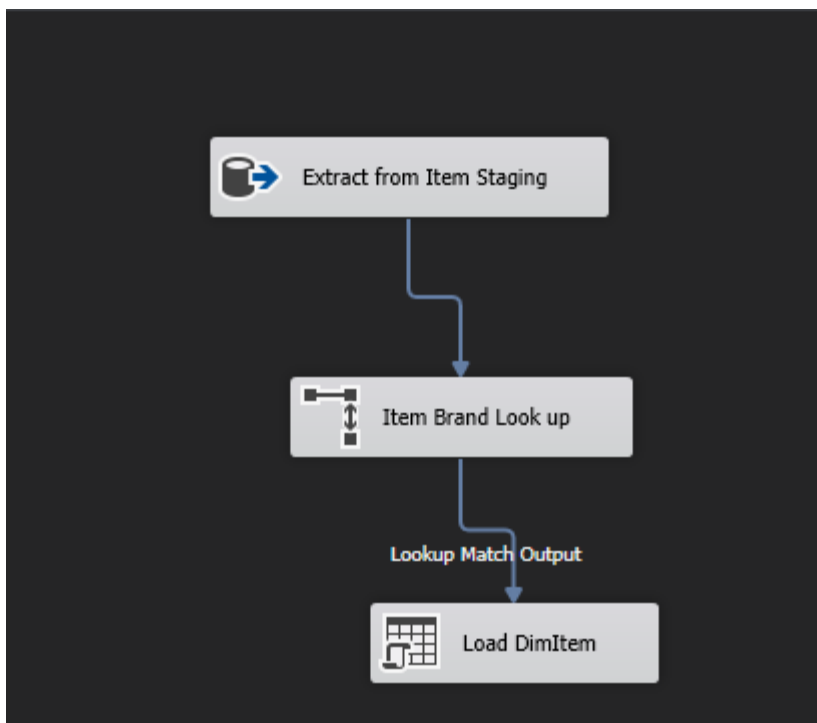
Join type: Left outer join Swap Inputs

<input type="checkbox"/>	Name	Order	Join Key
<input checked="" type="checkbox"/>	Brand_ID	0	<input type="checkbox"/>
<input checked="" type="checkbox"/>	Brand_type	0	<input type="checkbox"/>
<input type="checkbox"/>	CategoryID	1	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	modifiedDate	0	<input type="checkbox"/>

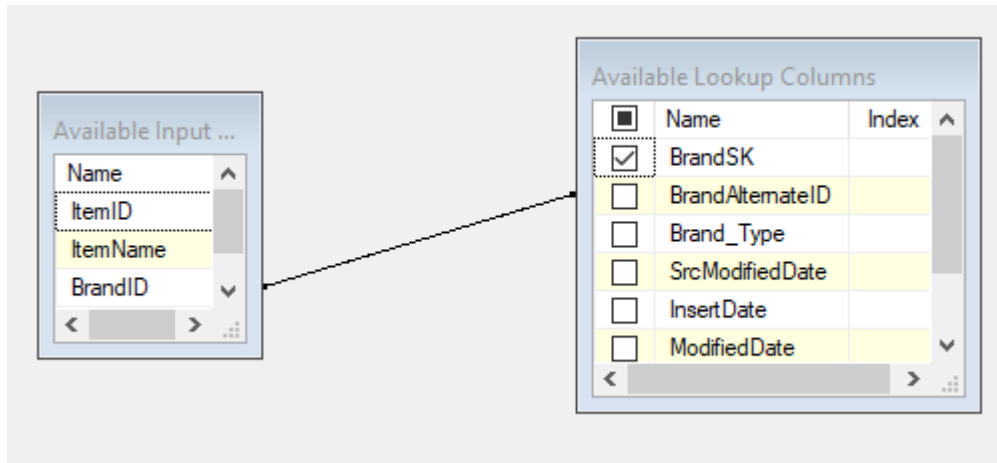
<input type="checkbox"/>	Name	Order	Join Key
<input checked="" type="checkbox"/>	CategorySK	0	<input type="checkbox"/>
<input type="checkbox"/>	CategoryAlternateID	1	<input checked="" type="checkbox"/>
<input type="checkbox"/>	CategoryName	0	<input type="checkbox"/>
<input type="checkbox"/>	InsertDate	0	<input type="checkbox"/>
<input type="checkbox"/>	ModifiedDate	0	<input type="checkbox"/>

**Figure 5.10 : Merge Illustration**

While loading Item table, since it requires BrandID reference lookup component was used to carry out the data flow. Under full cache mode, BrandID of Item table was mapped with BrandAlternateID to look up to the BrandSK of DimProductBrand

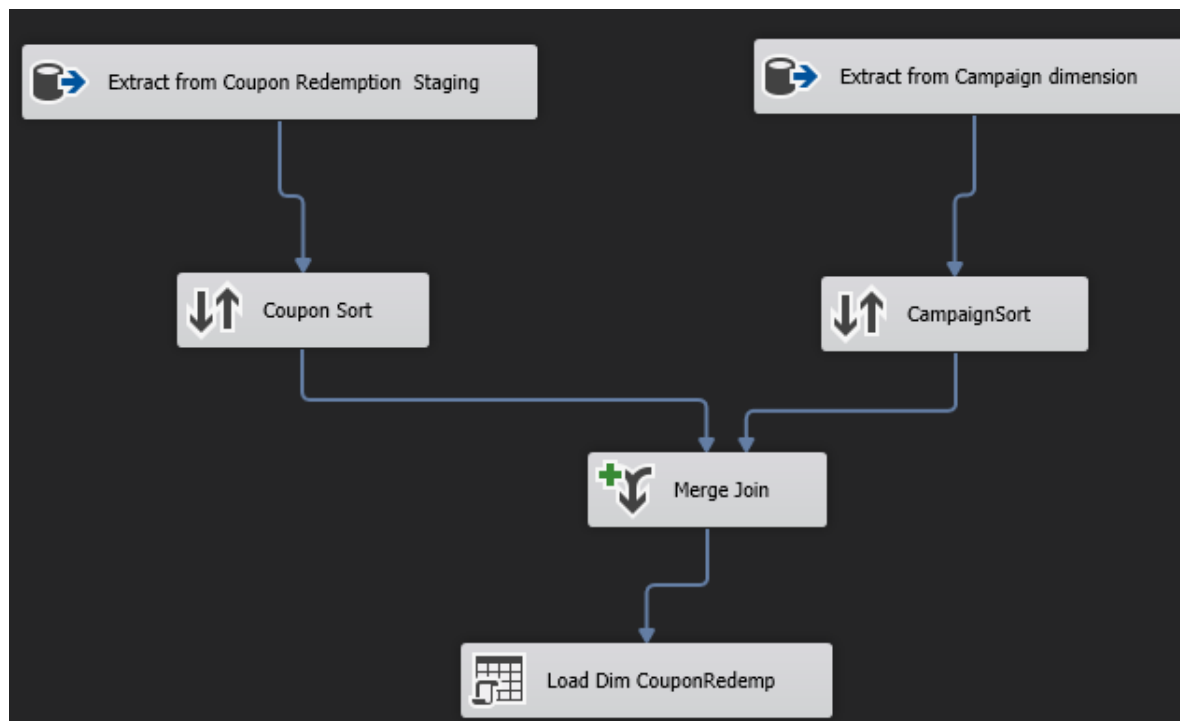


**Figure 5.11 : Dim Item Load**



**Figure 5.12 : Lookup Process**

In a similar way campaign and Coupon data were loaded into the Coupon\_DW. Since Coupon\_Redem table has a reference to campaign table sort and merge join components were used where initially Coupon data was sorted using CampaignID and CampaignData was sorted using CampaignAlternateID thus finally both were merged using a merge join in order to load data to the destination DimCouponRedem.



**Figure 5.13 : CouponDimensio loading**

## 2. Slowly Changing Dimensions

Customer Dimension is assumed to be the slowly changing dimension of the data warehouse where history management of such data are needed in order to implement specific marketing strategies thus maintaining better customer satisfaction.

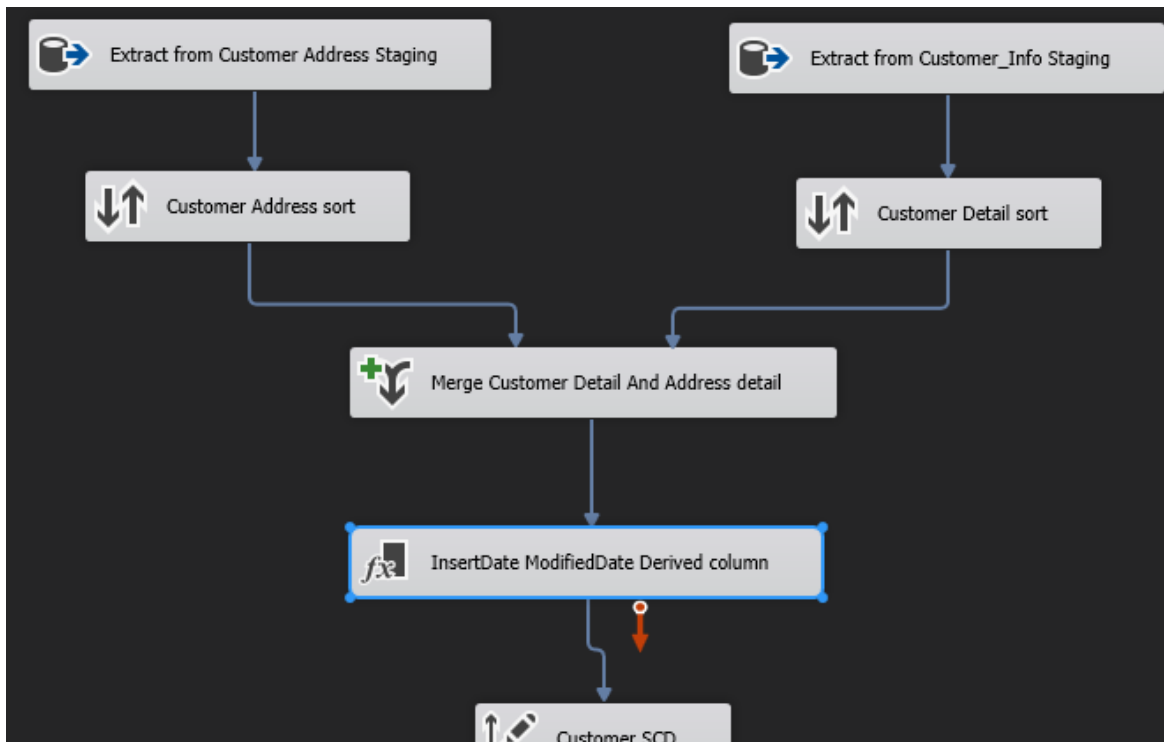


Figure 5.14 : Merging Customer Details

Initially since customer information and Customer location details are present as two separate files, they were sorted using the CustomerID and merged using a merge join component. To include the insert / modified date a **derived column component** was used.

Once this step was over, as per assumptions customer dimension was converted to a Slowly changing dimension using the **Slowly changing dimension component**.

**Note** – Start date and end date columns were incorporated in this dimension for history management purpose.

In the slowly changing dimension wizard, the attribute change types were done. The attributes Phone, marital\_Status, Accomodation\_Status, familySize, Number of Children, income\_bracket was considered as Changing Attributes basically implementing **TYPE 1** implementation where these attributes will be updated when source data values change.

The attributes AddressMain , City , State , Zipcode was considered as Historical attributes basically implementing **TYPE 2** implementation where a new record will be inserted in the target table when these attributes change in source table thus ensuring history management.

Rest of the attributes were not mentioned in the SCD wizard, therefore by default they are specified as fixed attribute basically implementing **TYPE 0** implementation where nothing happens to the target when these values are changed in the source.

Select a change type for slowly changing dimension columns:

Dimension Columns	Change Type
Phone	Changing at...
marital_Status	Changing at...
Accomodation_Sta...	Changing at...
familySize	Changing at...
NoOfChildren	Changing at...
income_bracket	Changing at...
AddressMain	Historical att...
City	Historical att...
State	Historical att...
ZipCode	Historical att...

**Figure 5.15 : Change Types for SCD**

Two additional columns start date and end date are incorporated to identify the latest record in Type 2 attribute implementation. Insertdate and modified date attributes are inserted via a derived column component.

The final data flow of the Slowly changing dimension created is as given below.



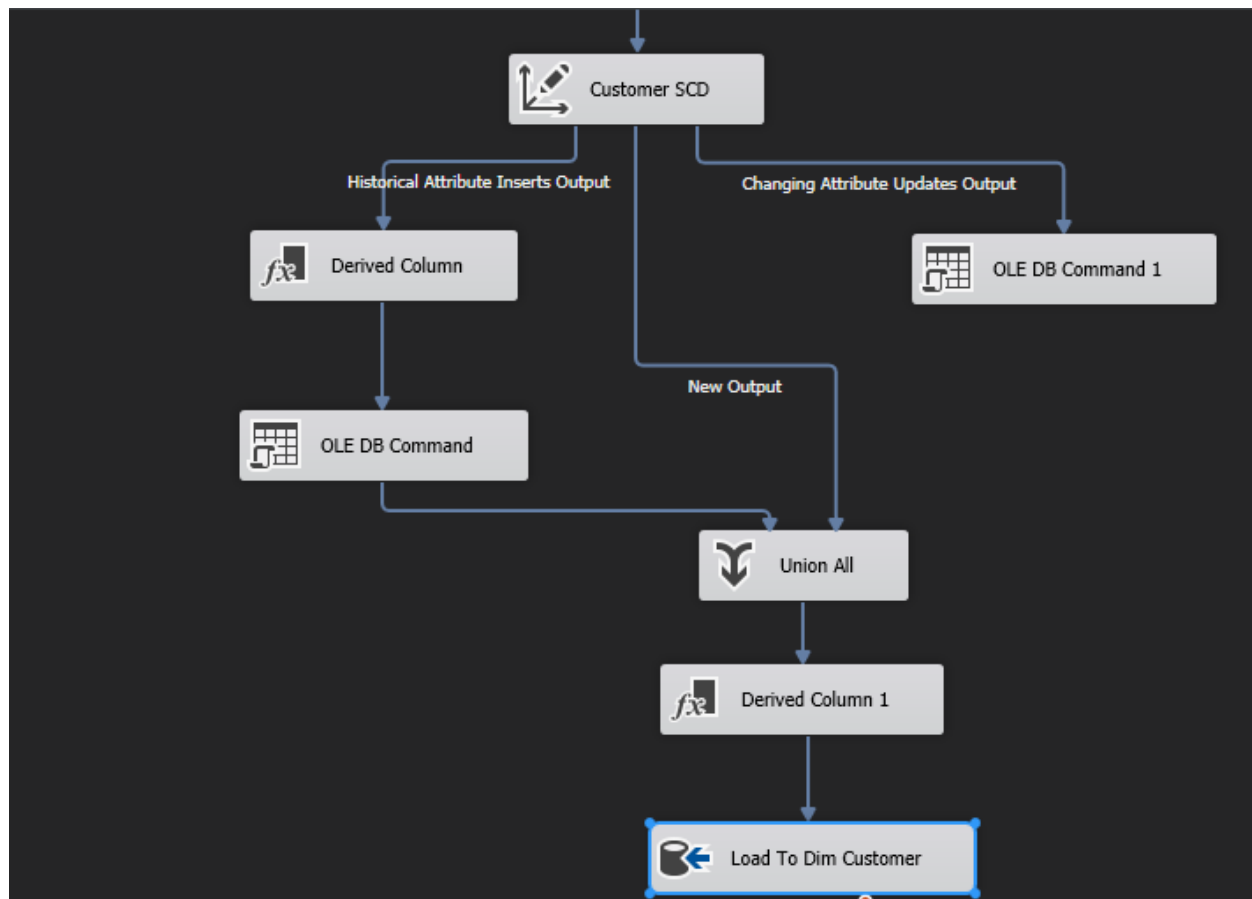


Figure 5.16 : DataFlow of SCD

### Testing Slowly Changing Dimensions.

Test cases were carried out by updating the Customer Staging table (Source) to check the data flow in the created slowly changing dimension thus analysing the modifications in DimCustomer.(Target)

#### 1. Type 0 attribute - Fixed Attributes

*/\*Update Type 0 attributes - Gender\*/*

```

update Coupon_Staging.dbo.Stg_Customer
set Gender = 'Female'
where CustomerID = 'C016'
  
```

Figure 5.17 : Type 0 Attribute test

Figure 5.18 : Type 0 Attribute test result

CustomerSK	CustomerAlternateID	CustomerName	Gender	Phone
16	C016	Michael	Male	760-882-7015

CustomerID	Name	Gender	Phone	marital_status	Accommodation
C016	Michael	Female	760-882-7015	Single	Landlord

## 2. Type 1 Attribute – Changing Attributes

```
/*Update Type 1 attributes - Phone*/  
update Coupon_Staging.dbo.Stg_Customer  
set Phone = '760-882-7000'  
where CustomerID = 'C016'
```

Figure 5.19 : Type 1 Attribute test

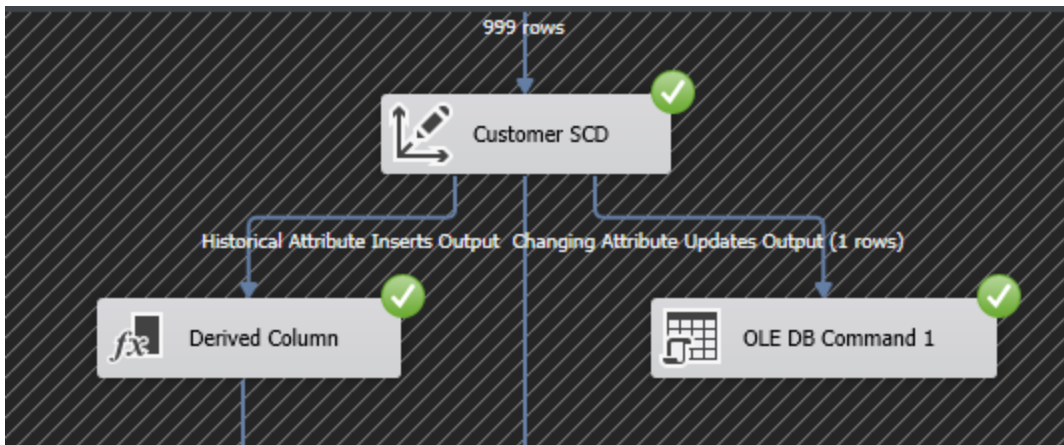


Figure 5.20 : Change Attribute Modification

## 3. Type 2 Attribute – Historical Attributes

```
/*Update Type 2 attributes - */  
update Coupon_Staging.dbo.Stg_CustomerAddress  
set state = 'NW'  
where Customer_ID = 'C016'
```

Figure 5.21 : Type 2 Attribute Test

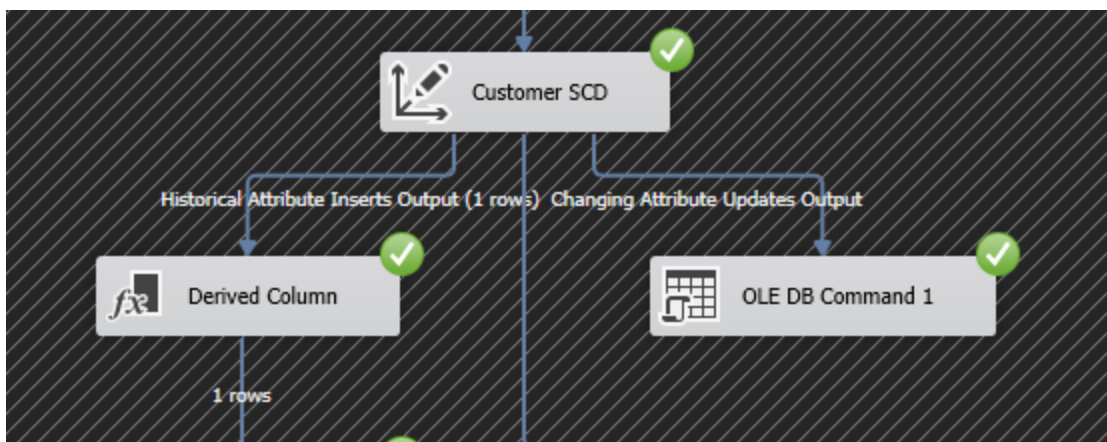


Figure 5.23 : Historical Attribute modification

### 3. Fact table

The final step of data extraction and loading to data warehouse is the process of loading data to Transaction Fact table. To complete it lookup components were used to obtain the key columns in the Transaction fact table.

Transaction fact table consists of four look up components referring DimItem , DimCustomer , DimCoupon\_Redem and DimDate, respectively. The data flow for Transaction Fact table loading is as given below.

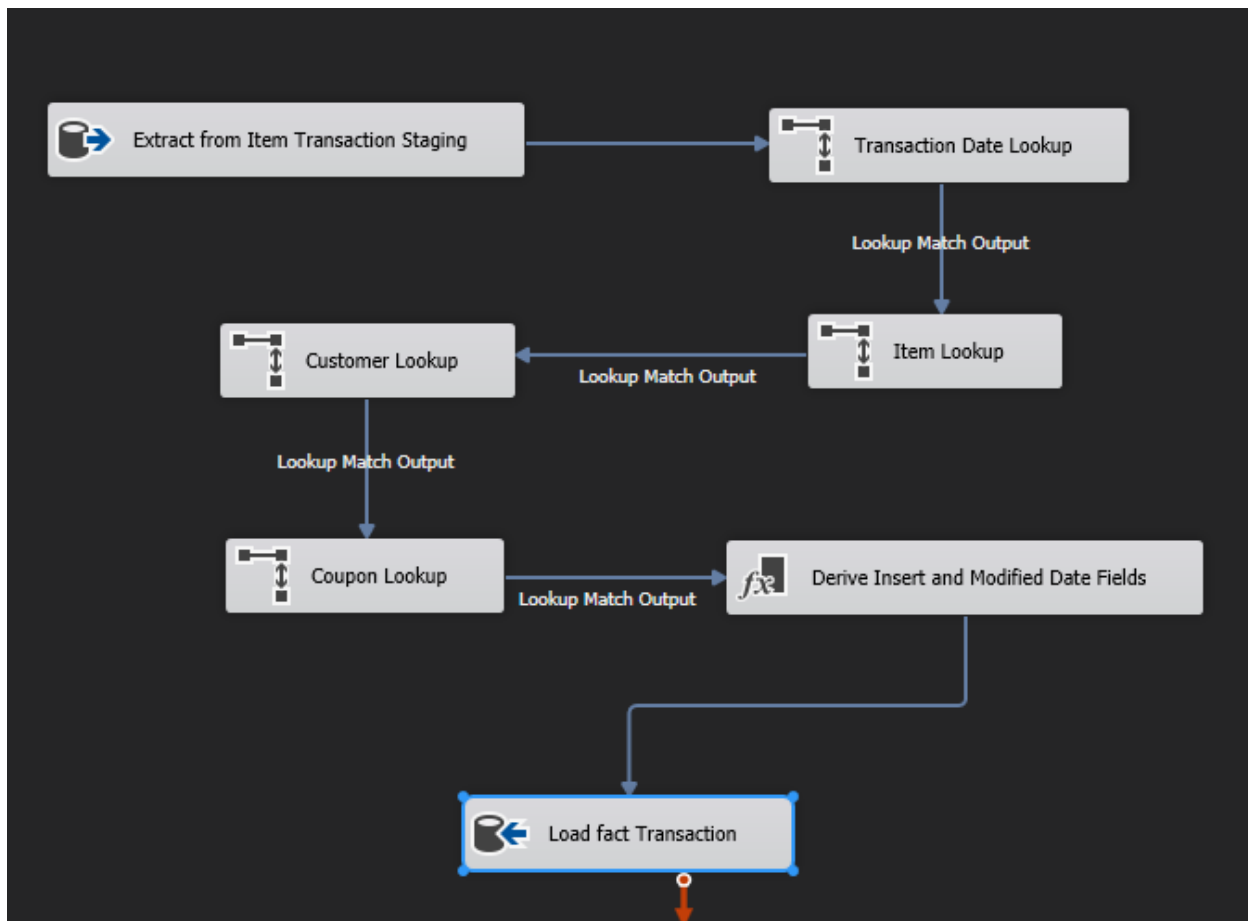


Figure 5.24 : Transaction Fact data flow

## Steps in Transaction Fact loading

1. Customer Transaction details from Stg\_ItemTransaction are extracted from Staging table using an OLE DB Source component.
2. A lookup component is added to extract the Date SK from Dimdate by mapping Transaction date in Stg\_ItemTransaction table with Date column in order to look up to the Surrogate key of DimDate.
3. Continuing from step 2 another look up component is added to extract the Item SK from DimItem by mapping Item\_ID in Stg\_ItemTransaction table with ItemAlternateID in DimItem in order to look up to the Surrogate key of DimItem.
4. A third look up component is added to extract the SK from DimCustomer by mapping Customer\_ID in Stg\_ItemTransaction table with CustomerAlternateID in DimCustomer in order to look up to the Surrogate key of DimCustomer.
5. The final look up component is added to extract the Coupon SK from DimCoupon\_Redem by mapping Coupon\_ID in Stg\_ItemTransaction table with CouponAlternateID in DimCoupon\_Redem in order to look up to the Surrogate key of DimCoupon\_Redem.
6. Then Derived Column components are added to incorporate Insert date / Modified date into Transaction fact.
7. Finally, the combined data is loaded to the Fact\_transaction in the datawarehouse with the usage of OLE DB Destination component.

As we are not maintaining history of other dimensions except the Customer Dimension , we should implement a method to have the updated latest record in the data warehouse. In order to do that implementation stored procedures were added in the target.(Data warehouse). The logic implemented within the stored procedure is that , if a tuple is already existing then we can update that record in the target else if it's a new record it can be inserted freshly into the target table.

Below attached are some snapshots of implemented Stored procedures.

```

CREATE PROCEDURE [dbo].[UpdateDimCoupon_Redem]
@CouponID varchar(10),
@CouponType varchar(20),
@CampaignIDKey int,
@redemptionStatus varchar(5)
AS
BEGIN
if not exists (select CouponSK
from dbo.DimCoupon_Redem
where CouponAlternateID = @CouponID)
BEGIN
insert into dbo.DimCoupon_Redem
(CouponAlternateID, CouponType, RedemptionStatus, CampaignIDKey, InsertDate , ModifiedDate)
values
(@CouponID, @CouponType, @redemptionStatus, @CampaignIDKey, GETDATE(), GETDATE())
END;
if exists (select CouponSK
from dbo.DimCoupon_Redem
where CouponAlternateID = @CouponID)
BEGIN
update dbo.DimCoupon_Redem
set CouponType = @CouponType,
RedemptionStatus = @redemptionStatus,
CampaignIDKey = @CampaignIDKey
where CouponAlternateID = @CouponID
END;
END;

```

Figure 5.25 : Coupon Dimension procedure

Figure 5.26 : ProductBrandDimension procedure

```

CREATE PROCEDURE [dbo].[UpdateDimProductBrand]
@ProductBrandID varchar(10),
@ProductBrandType varchar(20),
@ProductCategoryKey int,
@ModifiedDate date
AS
BEGIN
if not exists (select BrandSK
from dbo.DimProductBrand
where BrandAlternateID =@ProductBrandID)
BEGIN
insert into dbo.DimProductBrand
(BrandAlternateID, Brand_Type, SrcModifiedDate, InsertDate, ModifiedDate , CategoryIDKey)
values
(@ProductBrandID, @ProductBrandType, @ModifiedDate, GETDATE(), GETDATE(),@ProductCategoryKey)
END;
if exists (select BrandSK
from dbo.DimProductBrand
where BrandAlternateID = @ProductBrandID)
BEGIN
update dbo.DimProductBrand
set CategoryIDKey = @ProductCategoryKey,
Brand_Type = @ProductBrandType,
SrcModifiedDate = @ModifiedDate,
ModifiedDate = GETDATE()
where BrandAlternateID = @ProductBrandID
END;
END;

```