



DESIGN AND IMPLEMENTATION OF A DATA WAREHOUSE

Joint Consulting Project



Section 602 - Group 6

Team Members
Naresh Choudhary
Pallavi Tiwari
Chetan Joshi

MAYS BUSINESS SCHOOL, TEXAS A&M UNIVERSITY

TABLE OF CONTENTS

1. Introduction	2
i. Details about DFF data	
ii. Domain understanding	
2. Business Questions and Dimension Models	7
i. Business questions and corresponding models	
3. Data Marts and mapping tables	26
i. Kimball's Matrix for Data Marts	
ii. Data Marts	
iii. Mapping tables	
4. ETL Development Plan and Implementation	32
i. Data Quality	
ii. Development of ETL plan	
iii. Data mapping	
iv. Data Extraction rules	
v. Data transformation and cleansing rules	
vi. Aggregate tables	
vii. Organization of data staging area	
viii. Procedures for data extraction and loading	
ix. ETL for dimension tables	
x. ETL for fact tables	
xi. List of all temporary tables in the Staging area	
xii. Data Granularity at the independent data mart level	
5. Physical design	77
6. BI Reporting	80
i. Reporting Plan	
ii. Cube Generation using SSAS	
iii. Report using SSRS (Business Question 1)	
iv. Report using Report Builder (Business Question 2)	
v. Report using SSAS & SSRS (Business Question 3)	
vi. Report using SSAS (Business Question 4)	
vii. Report using Report Builder (Business Question 5)	
7. References	109

1. Introduction

Dominick's Finer Food chain, founded in 1918, was a Chicago-area grocery store chain which operated over 30+ locations during 1990 to 2000. DFF operated in sales of wide range of products including dairy, meat, cosmetics, alcohol, pharmacy and fresh produce. DFF was known for its experimentation over market to generate revenue. It experimented with 'food and drug' combo, walkie-talkie for communication, the ways through which the announcements were made to the customer and so on. After running for more than 20 years, on Dec 2013, its parent company announced to close all its stores in Chicago. [1]

To create a data warehouse for DFF, major concern is the complex data set. It contains files from 5 critical tables of DFF recording customer in-traffic, product details and inventory count over 10 years. Complexity is magnified by erroneous and incomplete information in the tables resulting in multiple interpretation of data, for example there are negative values in quantity of a commodity. It can be interpreted as junk or loss incurred to store due to breakage or spoilage leading to uncertainty.

Beside complex dataset, lack of subject knowledge of Retail and Marketing might hinder the development of a data warehouse which actually brings value to the business.

1.1 Details about DFF Data

DFF majorly consist of 4 data files: CCOUNT, DEMOGRAPHICS, UPC and MOVMENT.

Understanding the data:

CCOUNT: It is in-store traffic information file which contains number of customers visited and sales of each product in dollars segregated by date and stores. Information is also grouped by weeks. It also stores the coupons redeemed for various products on daily basis for each store. Data is dirty comprising of missing primary columns such as store id and date and includes unknown characters and negative values. Each store's demographic information is further described in Store-Specific Demographics excel file.

Store-Specific Demographics: It contains demographic details containing location of each store. For example, % of household with 1 person or % of College Graduates. Many tables hold repetitive data (i.e. contains same information) and can be removed/ignored.

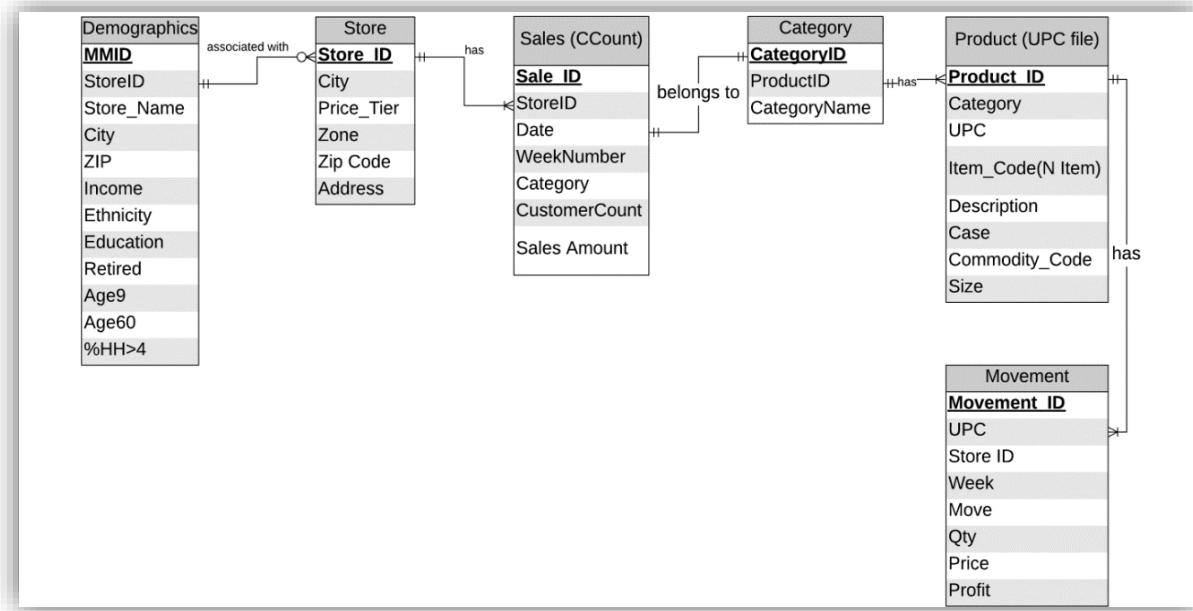
UPC: UPC files contain the unique key to identify products which are followed throughout market i.e. among different vendors, distributors and manufacturers. UPC key contains the product key associated with manufacturer key. Each file in UPC corresponds to single category which in turn has multiple commodities (identified by commodity code). DFF uses Item Code which uniquely defines a product within DFF database. Same key also helps to identify if the product was delivered to the Dominick's warehouse or directly to the store.

Movement: Movement file stores number of units, profit and Quantity segregated by week, UPC number and stores.

OLTP Metadata

File	Metadata	Description
CCount	Store, Date, Week, CustCount, Grocery, Dairy, Frozen, Bottle, Meat, MeatFroz, MeatCoup, Fish, FishCoup, Beer, Wine, Spirits, Pharmacy, DairyCoup, PharCoup, etc.	CCount has customer count data for a store for a particular date. It also consists of coupons redeemed and sales data for categories such as fish, dairy, etc.
Demographics	Age9, age60, ethnic, educ, nocar, income, incsigma, hsizeavg, hsize1, hsize2, hsize34, hsize567, Hh3plus, hh4plus, hhsingle, hhlarge, workwo, retired, unemp, wrkch17, nwrkch5, nwrkch17, wrkch, nwrkch, wrkwch, wrkwnch, telephn, mortgage, nwhite, poverty, etc.	This file consists of demographics data which is distributed store wise. It provides with data based on age, income, gender, family size etc.
Movement	Store, UPC, Move, Profit, Qty, Price, Sale, Week	Movement file stores the data of number of items that were sold for a store in a week
UPC	Com_Code, UPC, Description, Case, Size, Nitem	Product data from with description and commodity codes

Entity-Relation Diagram



Database Representation of Tables

CCount

Sales ID	StoreID	Date	Category	Sales	Customer Count
1	47	880101	GROCERY	14900	1546
2	47	880101	DAIRY	3321	1546
3	47	880101	FROZEN	2625	1546
4	47	880101	BOTTLE	0	1546
.
100	47	880102	GROCERY	15881	1749
101	47	880102	DAIRY	3321	1749
102	47	880102	FROZEN	2621	1749
103	47	880102	BOTTLE	0	1749
.

Store

StoreID	City	Price_Tier	Zone	ZipCode	Address
2	Chicago	High	B	60439	5400 N. Lakewood Avenue

Demographic

MMID	StoreID	Store_Name	City	Zip	Income	Ethnicity	Education	Retired	Age9	Age60	%HH>4
12345	2	Dominick Chicago	Chic ago	60439	10.2	0.6	0.8	0.67	0.54	0.46	0.33

Category

CategoryID	CategoryName	ProductID
WTPA	toothpaste	100002
WTPA	toothpaste	100001

Product

ProductID	UPC	Category	Item_code	Description	Case	CommodityCode	Size
1	1192603016	ANA	7342431	CAFFEDRINE CAPLETS 1	6	953	16 CT
2	1192662108	ANA	7333311	SLEEPINAL SOFTGEL	6	953	8 CT
3	1192603016	ANA	7342431	CAFFEDRINE CAPLETS 1	12	953	16 CT

Movement

MovementID	UPC	StoreID	Week	MOve	QTY	DiscountType	Price	Profit
1	1060831115	5	298	7	1		0.59	15.25
2	1060831115	5	335	2	1	B	0.51	50.98

1.2 Domain Understanding

“Retail is the process of selling consumer goods or services to customers through multiple channels of distribution to earn a profit. Retailers satisfy demand identified through a supply chain” [2]. There are four core principles to be followed in retail [3]:

1. Customer is most important
2. Understanding your customer
3. 4 Ps - Product, Price, Promotion, Place
4. Location

Current trends, Strategies & problems in Retails Market

Digital Disruption

The rise of e-commerce has disruptive effect on retail market. Currently, more than 10% of US retail market is dominated by e-commerce. According to Credit Suisse, more than 8640 stores with 147m sq. feet of space closed in 2018 [4]. Below graph supports the underlying trend:

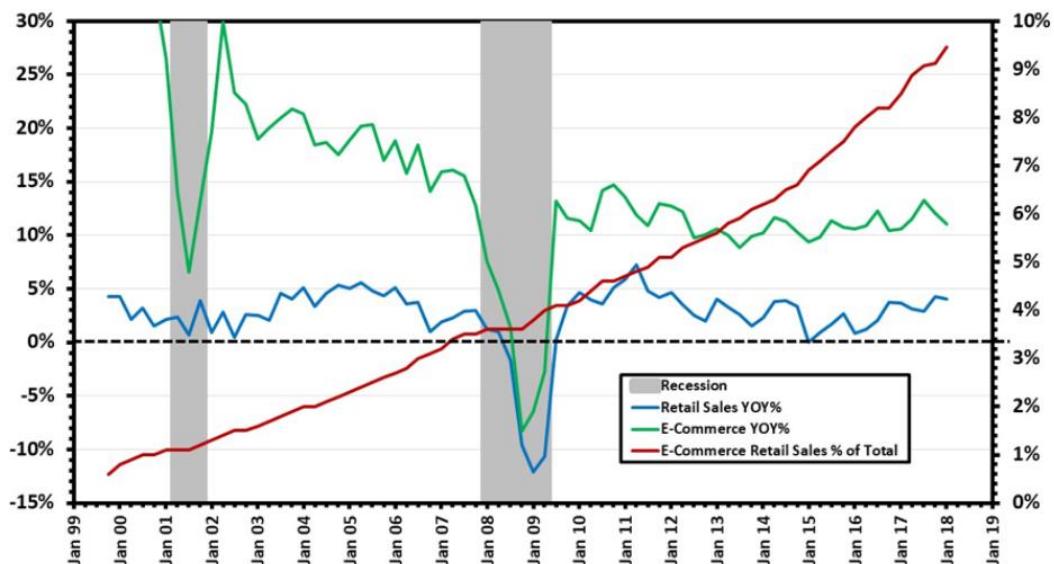


Figure 1) Retail and E-Commerce Year-Over-Year Sales Growth Rate (Left) and E-Commerce Retail Sales as Percent of Total Retail Sales (Right); Recessions are Shown in Grey (<https://fred.stlouisfed.org/>)

Retail chains should come up with ideas to annul the loss in revenue due to e-commerce by providing in-home service and delivery.

Customer Customization according to Demographics and Psychographics

Customers change their shopping pattern frequently and look for personalized experience. Currently retailers analyze demographic data to understand their customer and provide variety in products in which those customers are interested in. Steps involve analyzing customer base, checking competition, understanding product/service, choosing specific demographic to target and

finally expanding product base [4]. However, targeting a specific population doesn't mean you are excluding people who doesn't fit criteria. [5]

Marketing Trends by One of the Smartest Supermarket in the United States

Texas based super mart H-E-B is notorious for promoting local produce by associating them with symbol of pride towards their state. Products are branded like Houston Blend Coffee and Hill Country, which trigger the sentiments in the customers and lead to similar amount of sales as compared to branded products.

Another strategy they use is Neuro marketing by utilization of yellow coupons spread across the aisles. "Put a big yellow price sign next to something, and it will light up people's brains - even if it's not actually a great deal" [6]. In H-E-B's case, they offer flagged coupons to tear off and to be carried to the checkout counter for scanning which takes Cialdini's principles of persuasion in account by influencing the customers to engage rather than just pick up a product with discount pasted on the product label.

Utilizing this strategy can lead to promotion of certain products which may have been failing to grasp customer's attention. Furthermore, by putting these strategies in action, Dominick's can promote in-house products and utilize coupons in the stores.

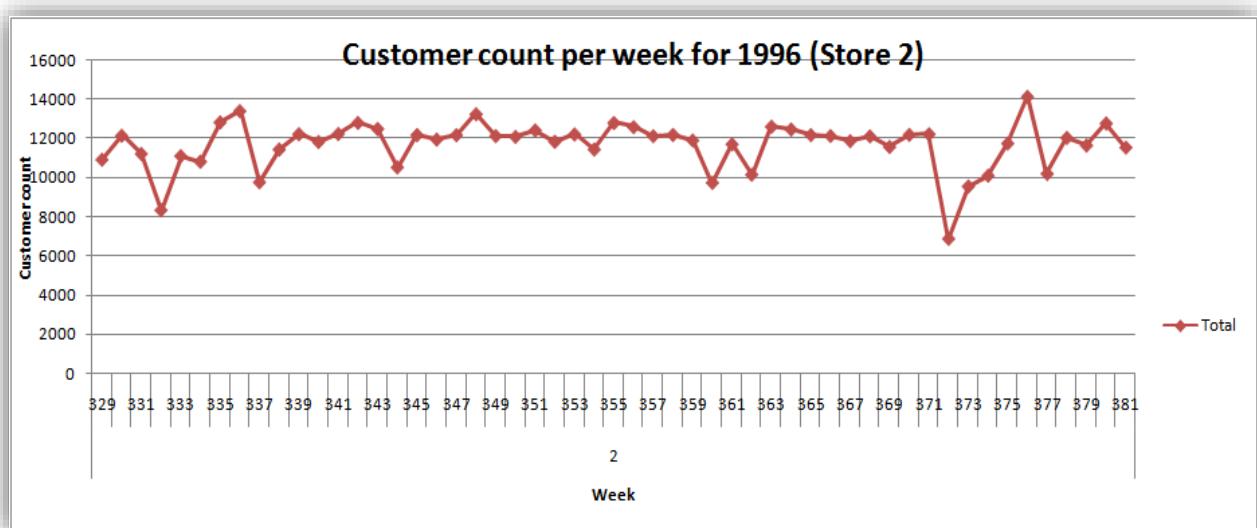
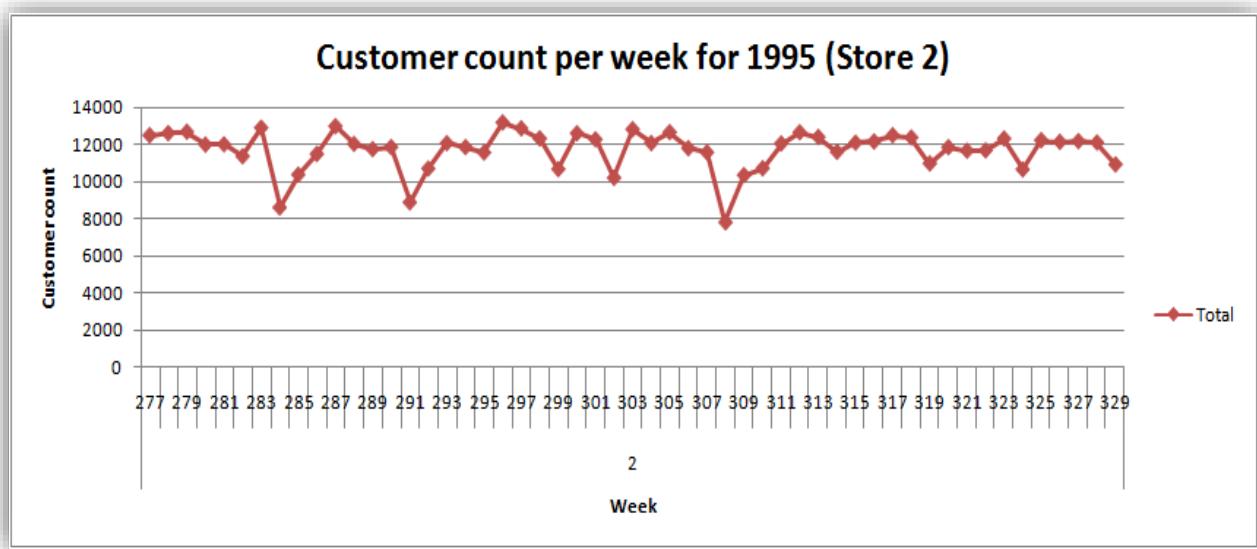
Revenue Vs Profitability

Retailers face several challenges, the biggest one is to increase profitability. Customers have ever changing demands which are difficult to track. This causes problems in managing the supply chain. There may be products which are always in demand but there is low availability due to supply chain inefficiencies or lack of labor. Retailers also fail to evaluate the potential customers and markets which leads to lower sales. The inventory management system especially during the holiday season is not efficient to manage the inventory across the all the stores. Customers are ever demanding of good quality products which are cheap. To do this, they resort to multiple channels such as online shopping. Retailers are expected to be consistent across all the channels and provide convenience to the customers. Often, retailers fail to estimate their competitors' capabilities. Small retailers cannot compete with big retailers on price. Conclusively, they need to focus on something other than price to create a good shopping experience. [7]

2. Business questions

2.1 What is weekly trend of customer in traffic for store 2 between 1995 and 1996?
(Selected)

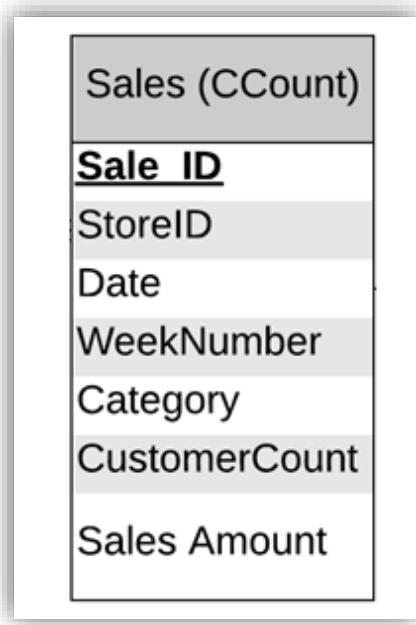
Pivot:



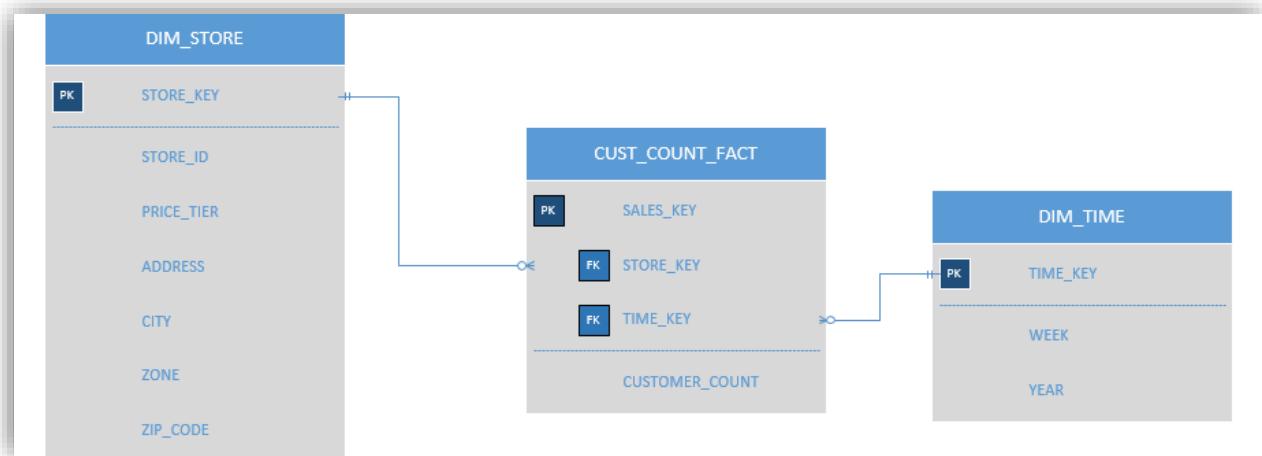
Business Justification: For a sample of stores, we have taken data for year 1995 and 1996 to find out the number of customers visiting the store weekly. Using this data, we have identified a general trend of reduction in customer count during the 1st week of months. During these trough points in graph i.e. the weeks with reduced customer count, we can reduce the staff or let the staff members

plan their vacations. This will further lead to reduction of over-stocking and employees count can be reduced to save salary expenses.

ERD:



Dimensional Model:



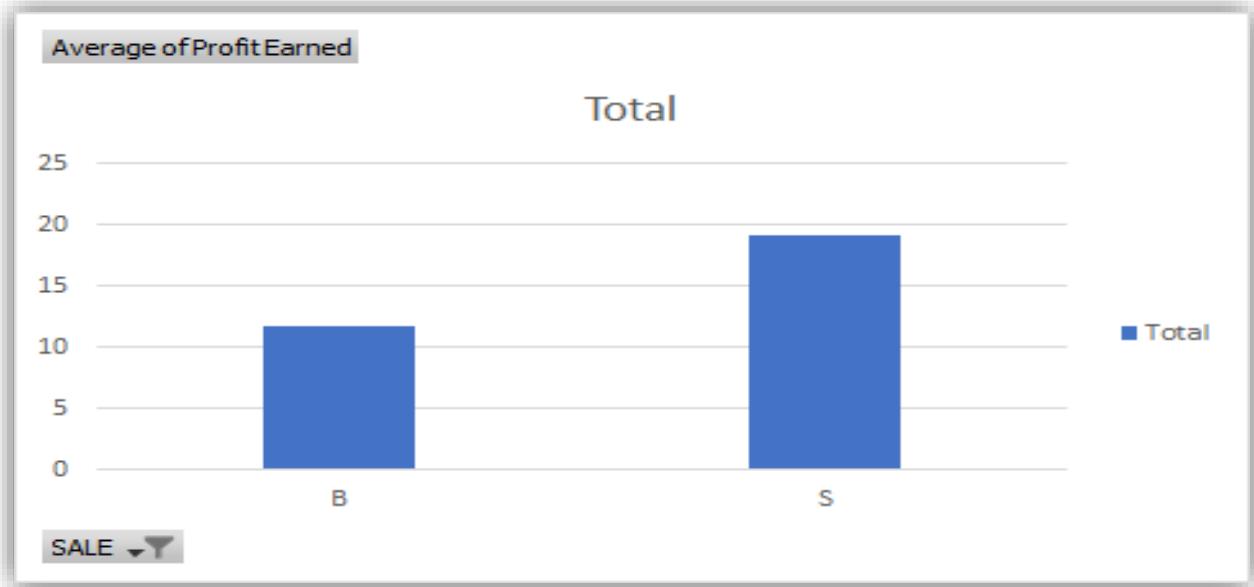
Schema Justification: To calculate customer count on a specific store on weekly basis, we need two dimensions namely *DIM_TIME* and *DIM_STORE*. Time dimension will aggregate the data on weeks and year. Similarly store dimension will help to find customer count on different stores.

CUST_COUNT_FACT will have auto incrementing primary key and foreign key to other dimensions. *CUST_COUNT_FACT* will have only one attribute *CUSTOMER_COUNT* which

corresponds to customer in traffic on a particular week on a particular store (summing daily customer count to aggregate it week wise: $\text{sum}(\text{customer_count_daily}) \text{ from sales_table}/CCount$ group_by week, group_by store)

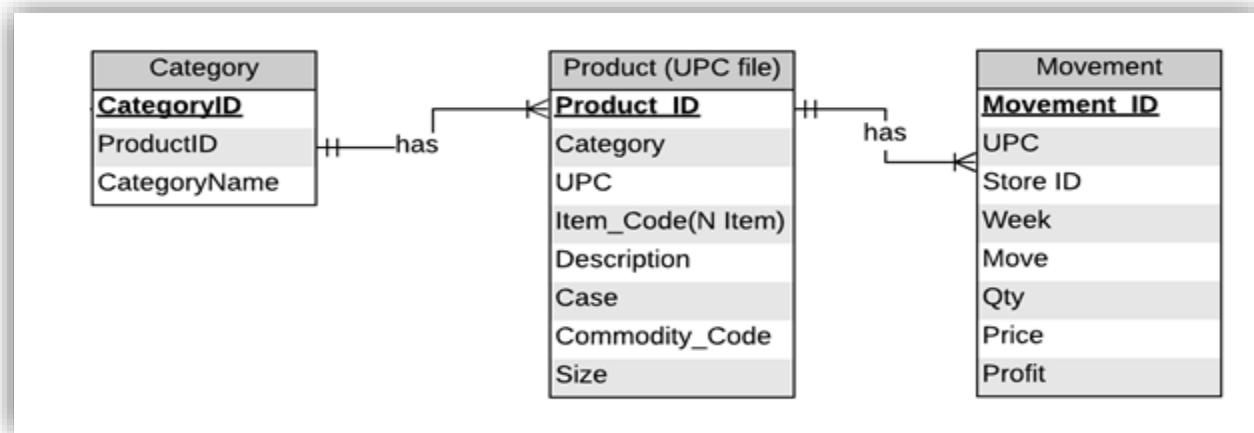
2.2 In category of frozen products, which method of discount (Coupon or price reduction) gives more profit? (Selected)

Pivot:

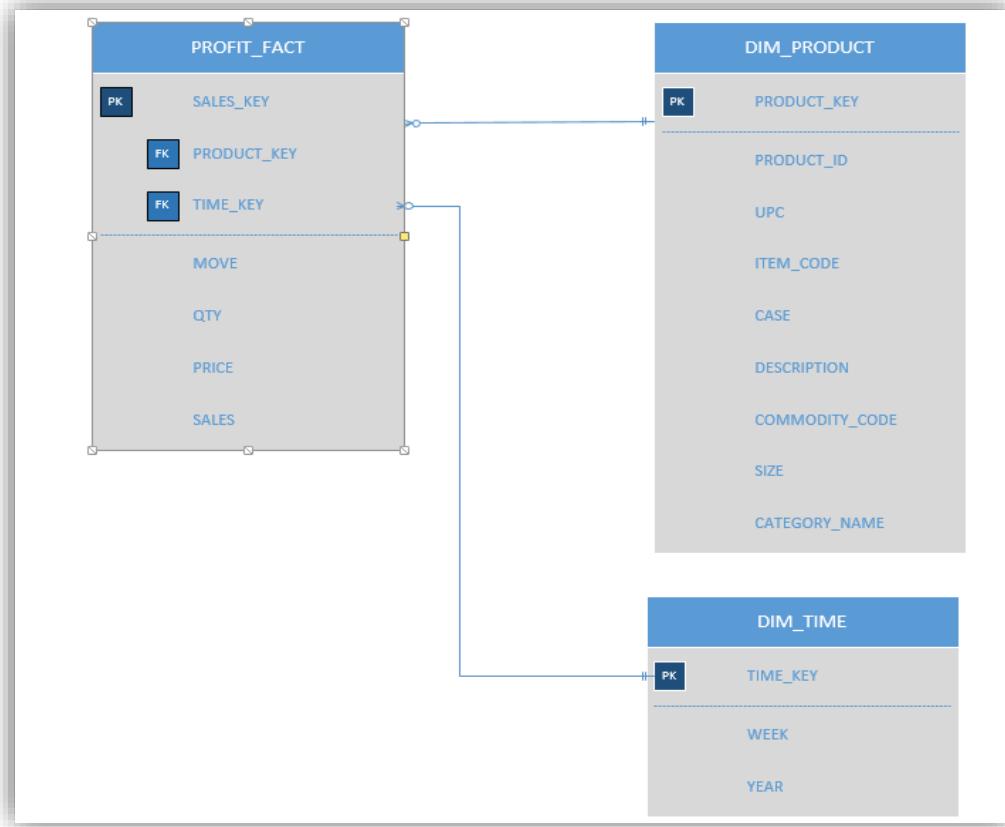


Schema Justification: Due to lower shelf life of frozen foods they are often sold on discounted rates to avoid loss. Analysis of profit earned in category frozen entree shows that when simple price reduction is offered, sale and profit is more. Dominick can consider offering simple price reduction rather than offering coupons.

ERD:



Dimensional Model:

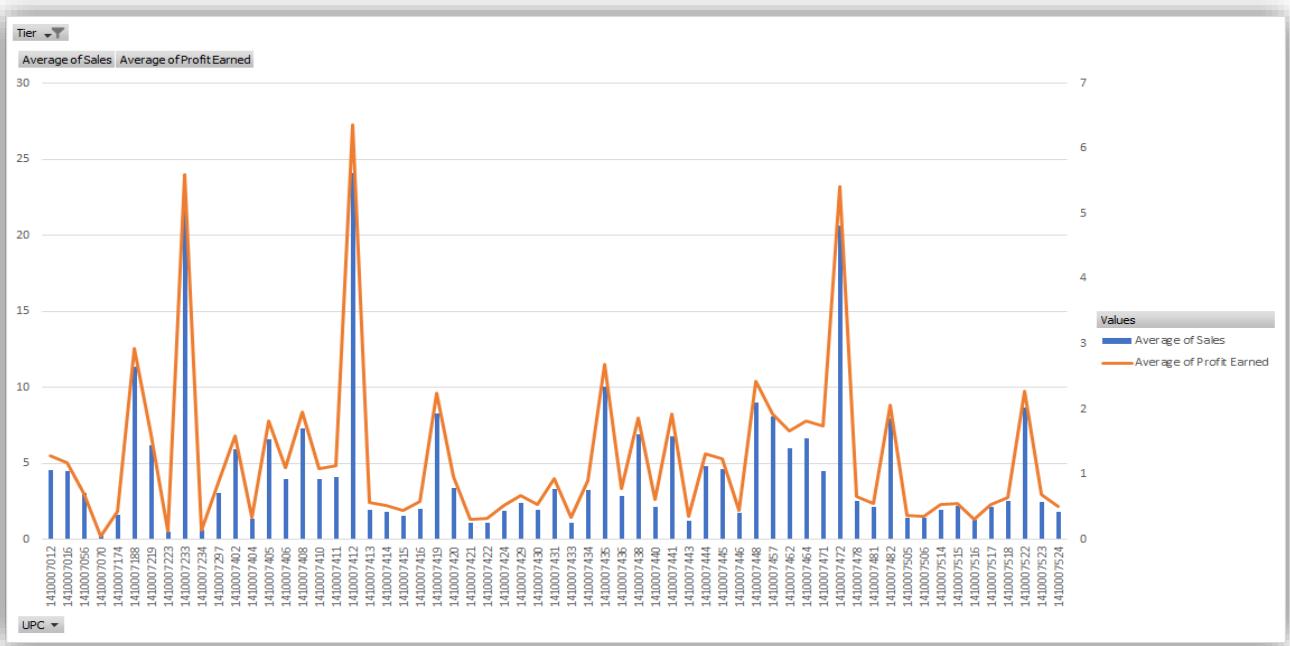


Schema Justification: For this question, we propose a schema which consists of a fact table PROFIT_FACT and two-dimension tables as DIM_PRODUCT and DIM_TIME. The basic requirement of this question is to calculate the sales of frozen products when products had simple discount vs same product when they had bonus buy offer. This schema holds these sales details in PROFIT_FACT as ‘sales’ attribute. Since BQ asks for frozen products, PROFIT_FACT is linked to DIM_PRODUCT in which PRODUCT_KEY is the surrogate key & PRODUCT_ID is the OLTP database’s ID. Further, to figure out discount type product dimension for any week or year (coming from the DIM_TIME) table will be used.

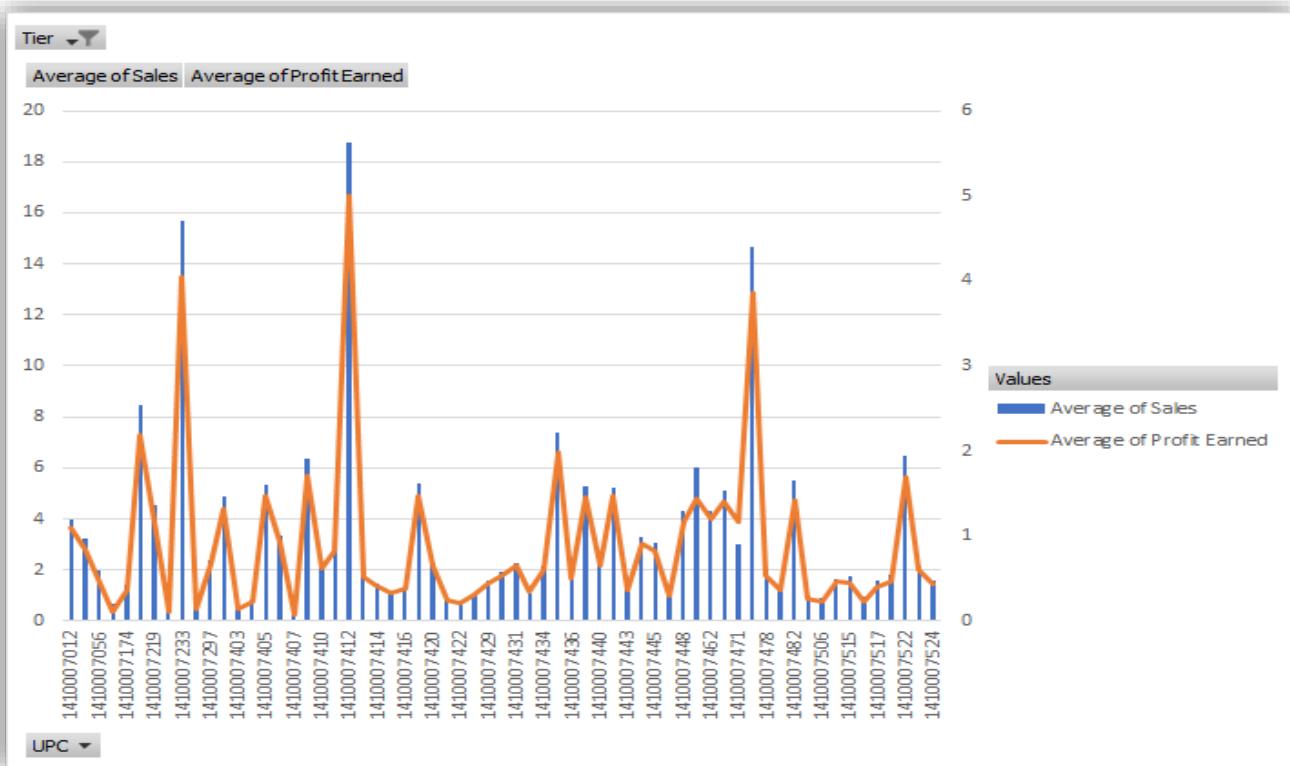
Data in PROFIT_FACT table comes from movement table/movement file. Since ERD movement table (or movement file from Dominick) records data on a daily basis, sales will be aggregated week wise.

2.3 Which UPCs had most sale in high, medium and low tier for cookies? (Selected)

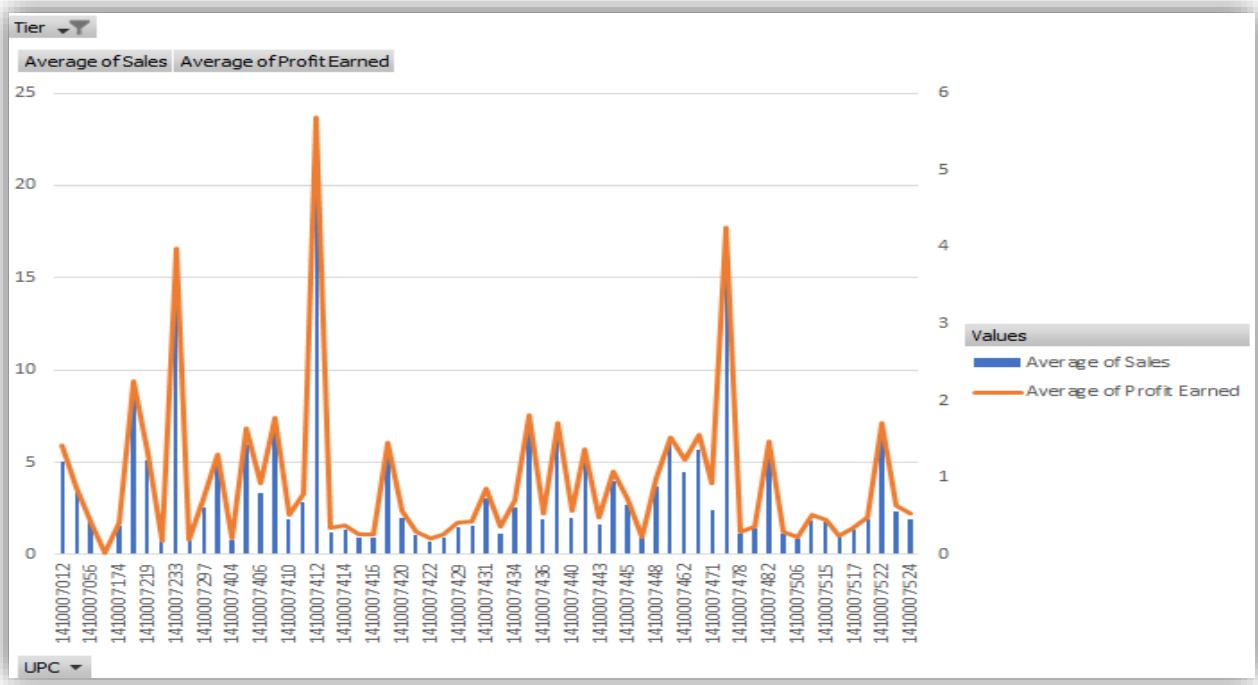
Pivot (High tier stores):



Pivot (Low tier stores):

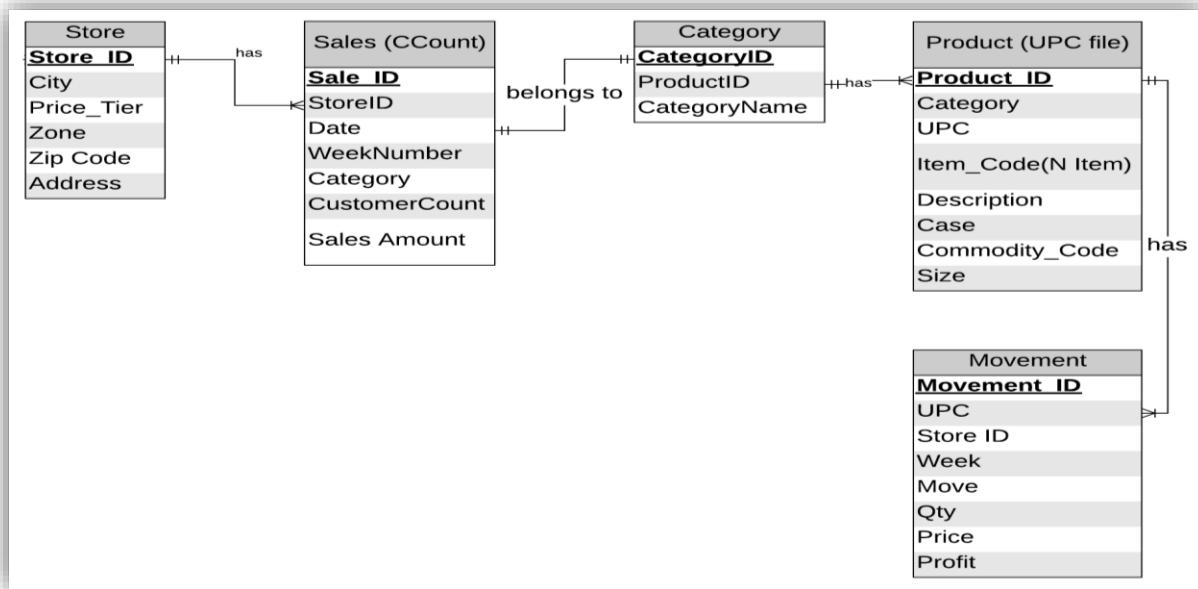


Pivot (Medium tier stores):

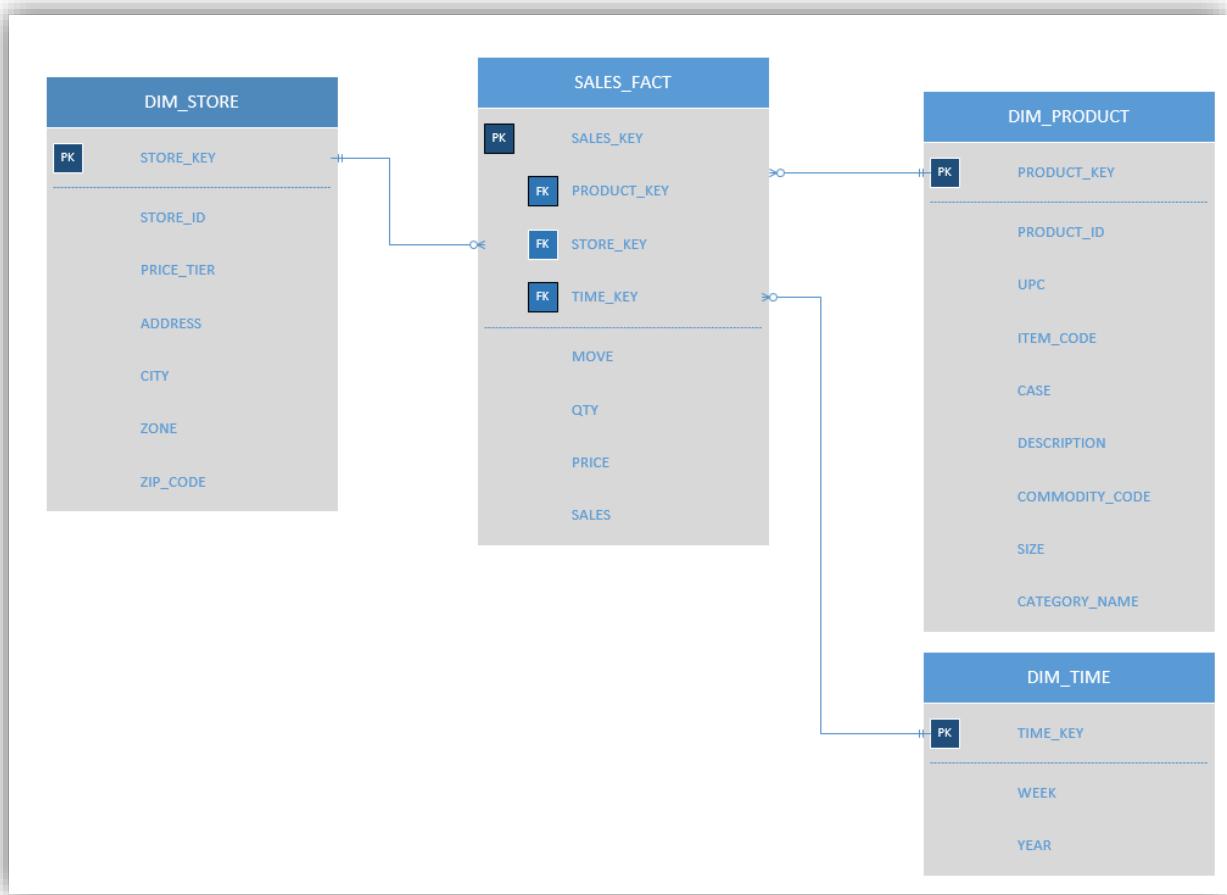


Schema Justification: For cookies, the pivot charts show data for different tiers i.e. high, medium and low. We observed that sale of certain UPCs was very high as compared to the others. For example, the sale and profit of UPC is high across all the tiers. This analysis can help the business to promote certain UPCs more than the others. Our analysis on different tiers for the cookies shows that known products, especially high and medium tiers one, are often bought. The business can then focus on the sale of these products by introducing coupons or discounts. They should particularly aim at the high/medium tier to increase profit margins substantially.

ERD:



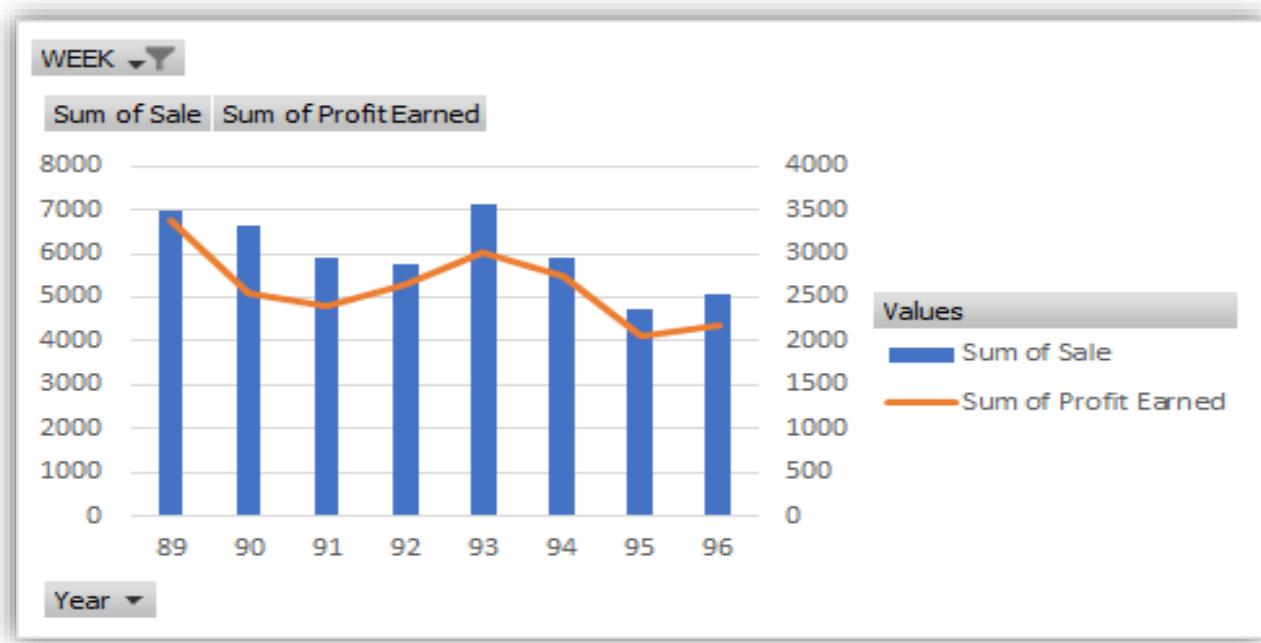
Dimensional Model:



Schema Justification: For this question, we propose a schema which consists of a fact table SALES_FACT and three-dimension tables as DIM_PRODUCT, DIM STORE and DIM_TIME. We need to find the UPCs (Cookie type) that had highest sales in different tiers. For this, we have designed the SALES_FACT table to store the movement, quantity, price and sales data (from movement table). The UPC data comes from the Product dimension. DIM_STORE will store the store details which will provide the tier data. In this way, we can capture the sale of a UPC in a particular store/tier.

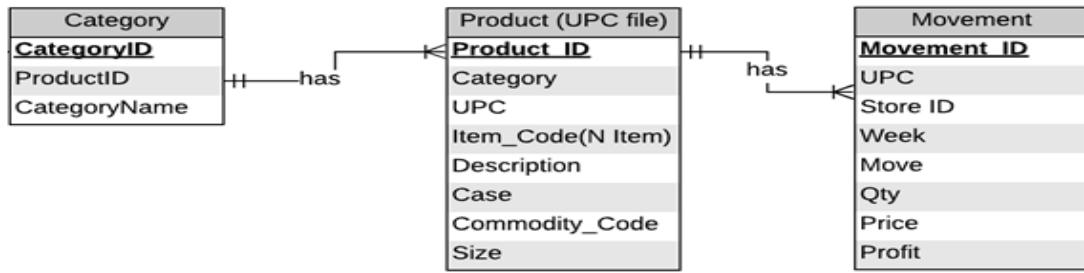
2.4 What is the trend of candies' sale during Halloween year by year? (Selected)

Pivot:

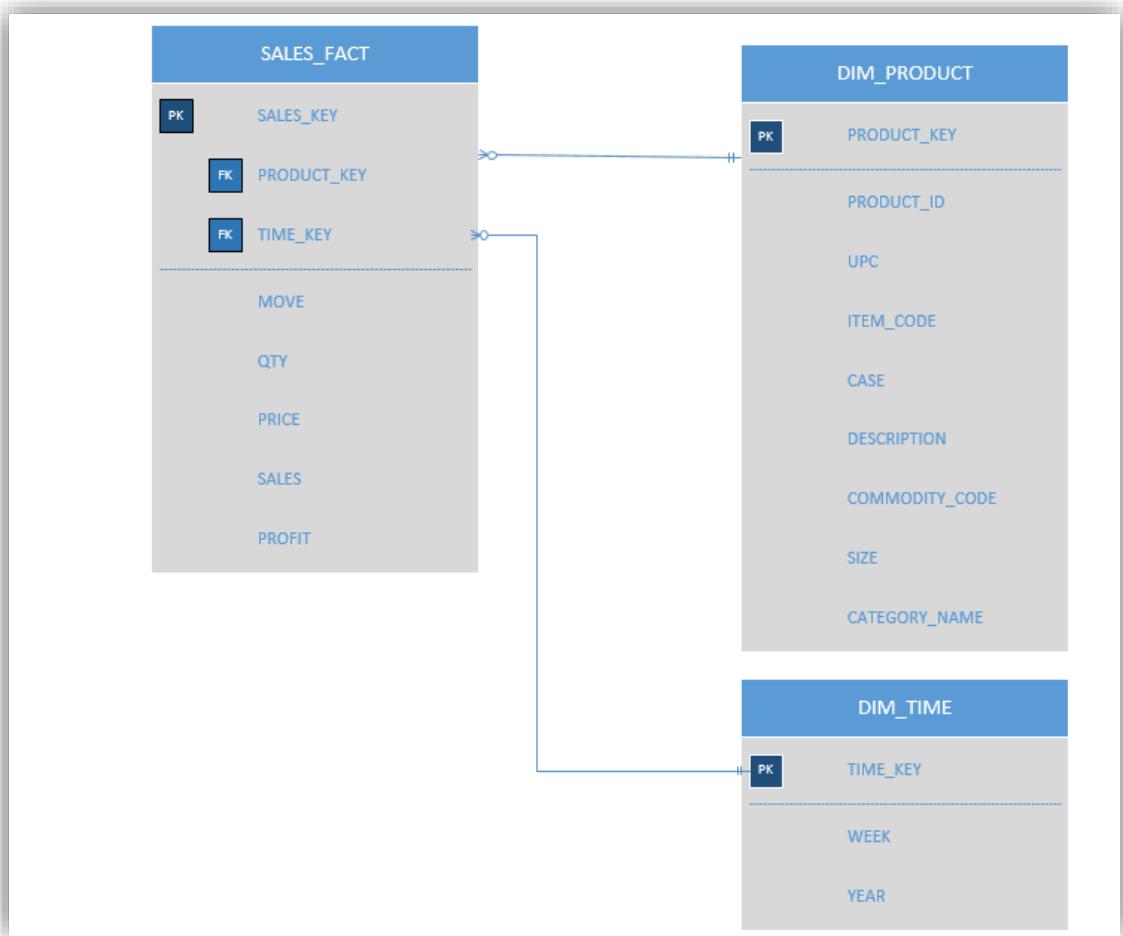


Schema Justification: Graph shows the trend of candies during Halloween season from 89 to 96. It can help to predict sales of next year which could range between 4000-5000. This trend can be validated with data from years to come.

ERD:



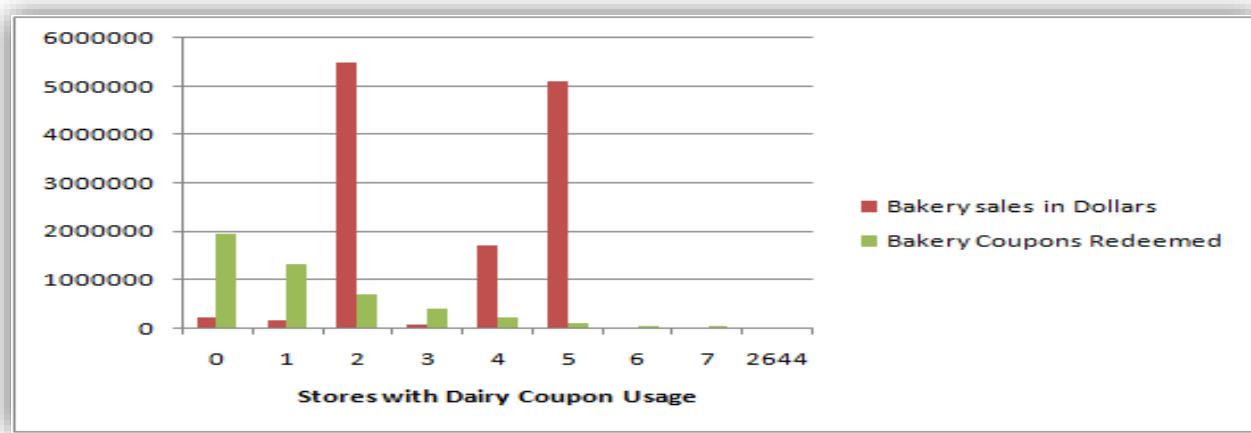
Dimensional Model:



Schema Justification: To find the sales of candies again uses SALES_FACT which is linked to DIM_TIME to find Halloween weeks and DIM_PRODUCT to find select only candies products.

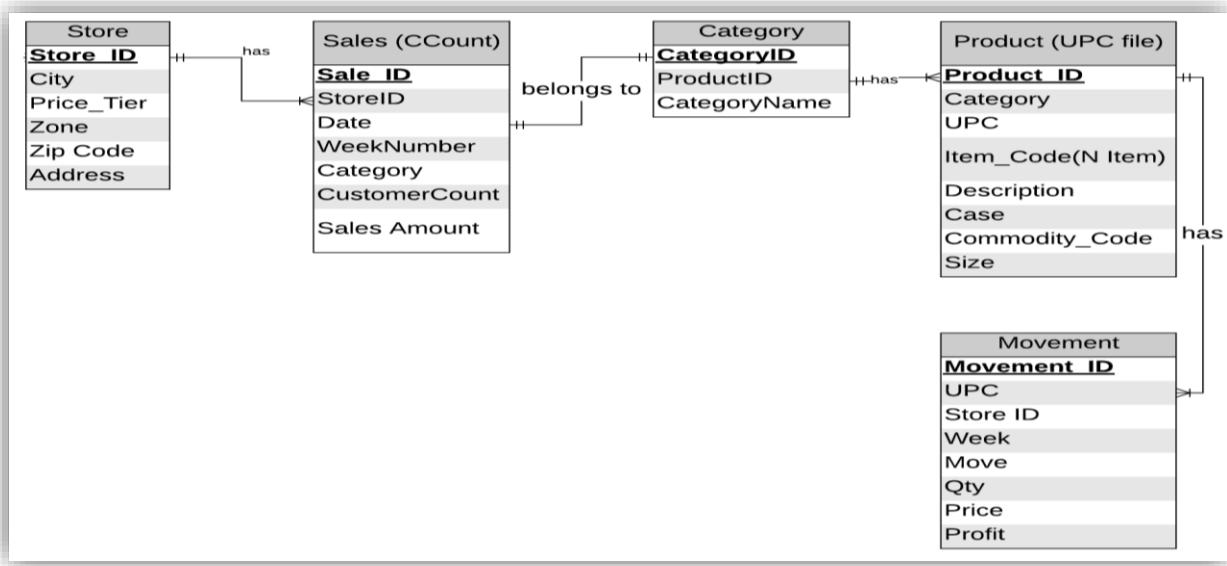
2.5 Find out the relation between Bakery sales and bakery coupons redeemed for all stores? (Selected)

Pivot

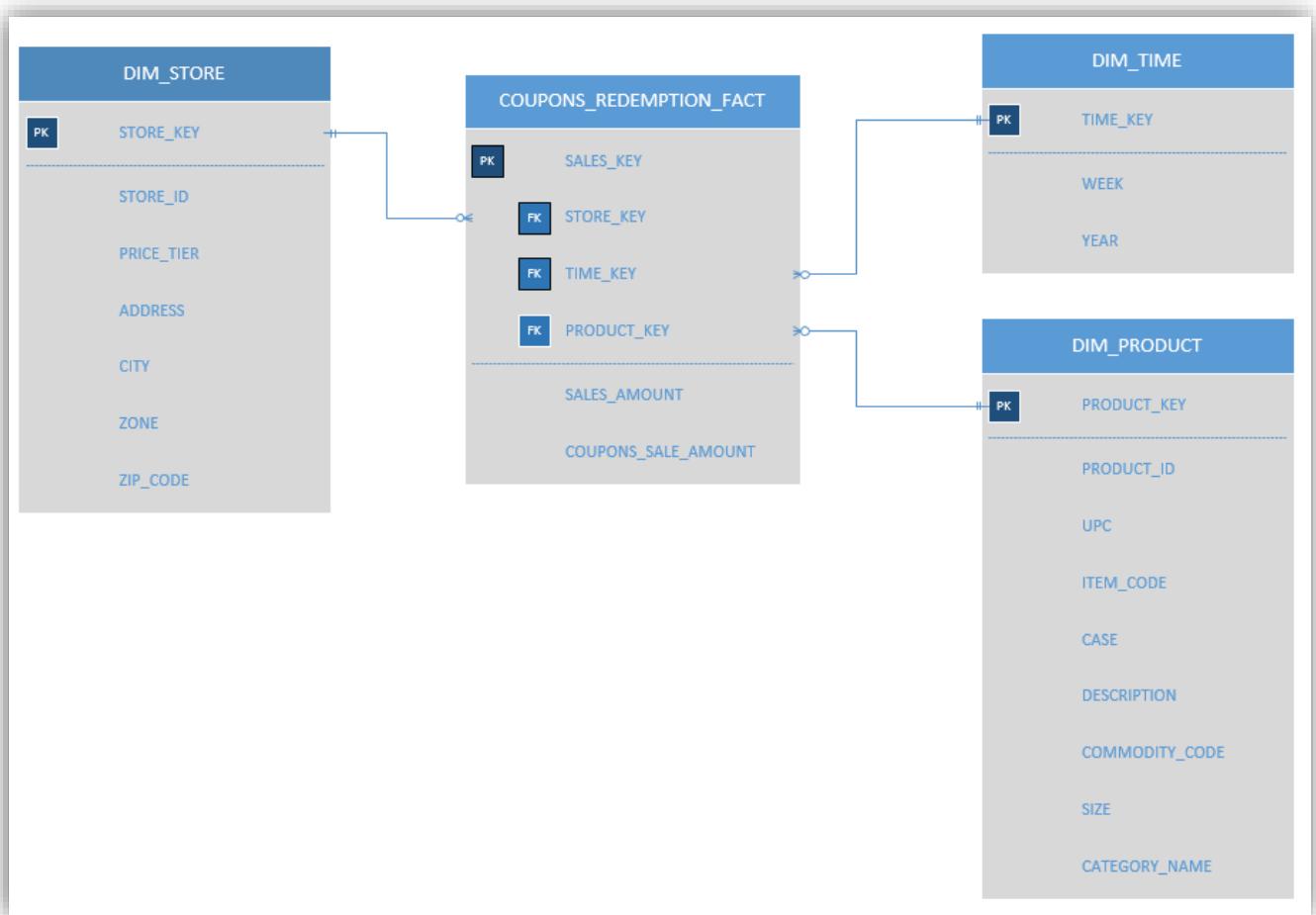


Schema Justification: In this business question, we looked at the data given in the CCount table for bakery sales. We wanted to evaluate this amount store wise to see if there is any relevance of the coupons redeemed for bakery items. We noticed that for certain stores such as 0, 1, 2 as shown above, the coupons redemption was high. In contrast, the sale of bakery items was very low. If we see the sale for store 2, 4 and 5, we notice that the bakery sales were extremely high. In this case, the coupons redeemed was not so substantial. This analysis can be further explored to evaluate the difference. One of the strategies by DFF can be to give out more options in coupons or discounts to attract customers in store 2, 4, 5 to eventually increase the sales exponentially. DFF can also try to understand the trend of sales for coupons in stores 0, 1, 2 and strategize to increase the bakery sales in those stores.

ERD:



Dimensional Model:



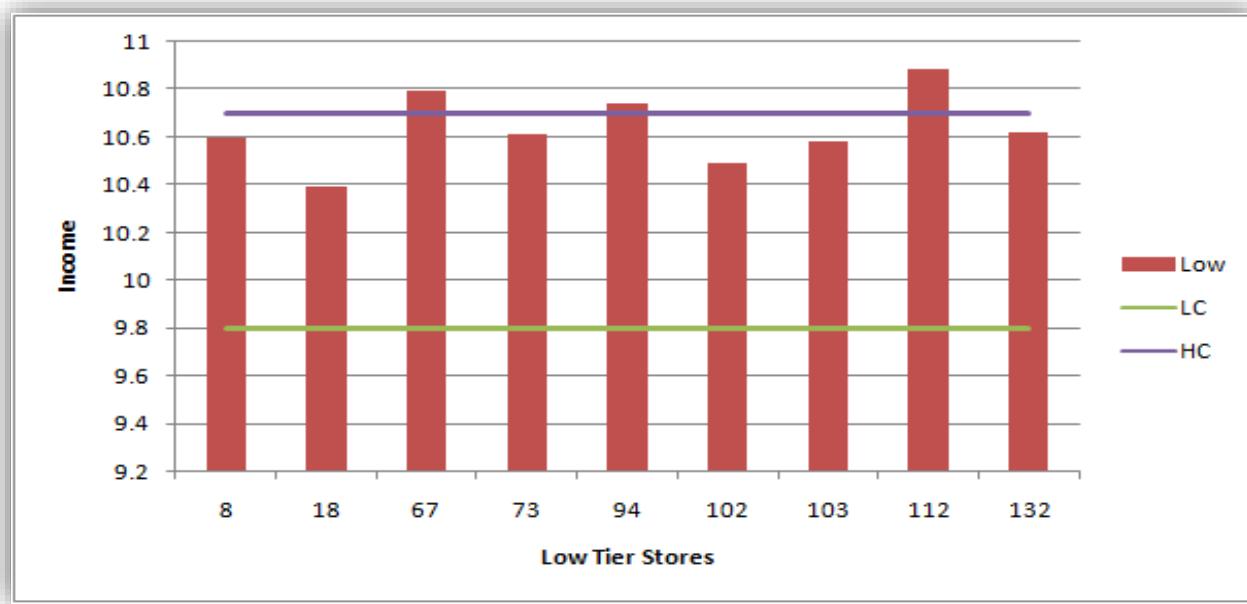
Schema Justification: To find the relation between bakery sales and coupons redeemed for all stores, we propose a schema which consists of the COUPONS_REDEMPTION_FACT fact table. This table stores the SALES_AMOUNT for regular sales and COUPONS_SALE_AMOUNT for the coupons redeemed sales amount from the CCount table data. The DIM_STORE table stores the store details and the DIM_PRODUCT stores the product details along with the CATEGORY_NAME. The DIM_TIME dimension table stores the week at the lowest granularity, moving up to year.

2.6 Find out stores which are wrongly categorized as Low, Medium or high based on the demographic income?

It is assumed that tier of a store depends on the income of people of that area. A high tier store's product and their prices are relatively high as average income in that area is above median income. Overall income of Chicago demographic ranges from 9.8 to 11.5. A low tier store refers to a zone where income lies between 9 and 9.8. A store is considered as medium tier if income lies between 9.8 and 10.7. A high tier store has population whose income lies between 10.8 & above. Apart from income other factors contribute too to decide the tier. However, there are facts represented by our analysis that strongly question the tier of stores.

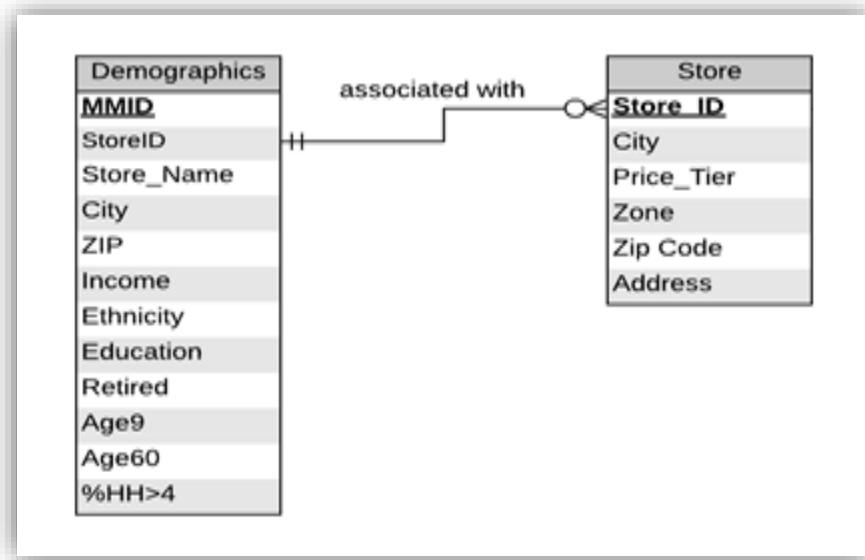
Low Tier:

Pivot



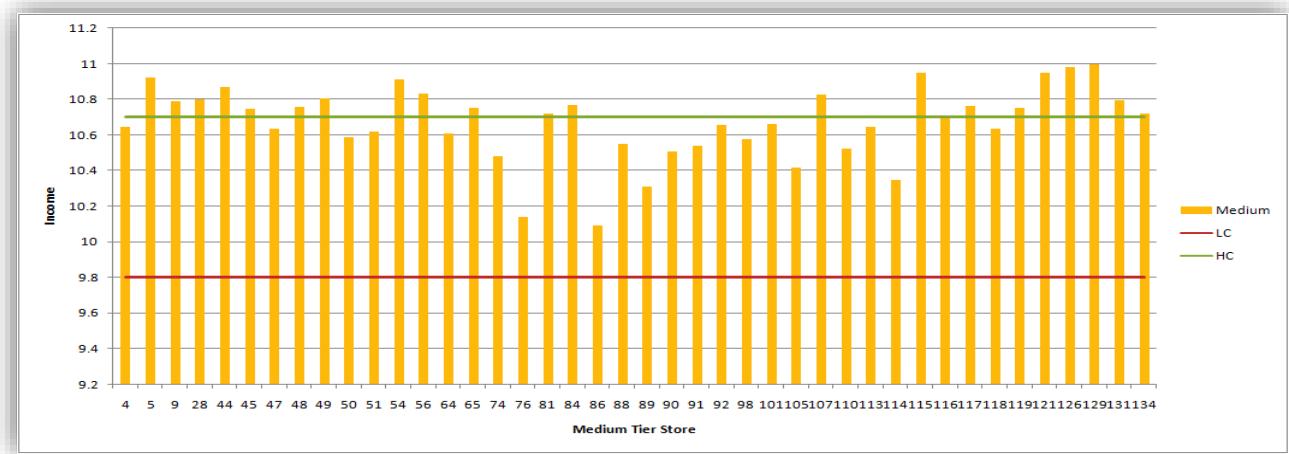
Justification: For example, graph shows that average income of families around store 67 and 112 is more than average income of high tier stores (10.8). Store should reconsider tier of these stores.

ERD:



Medium Tier:

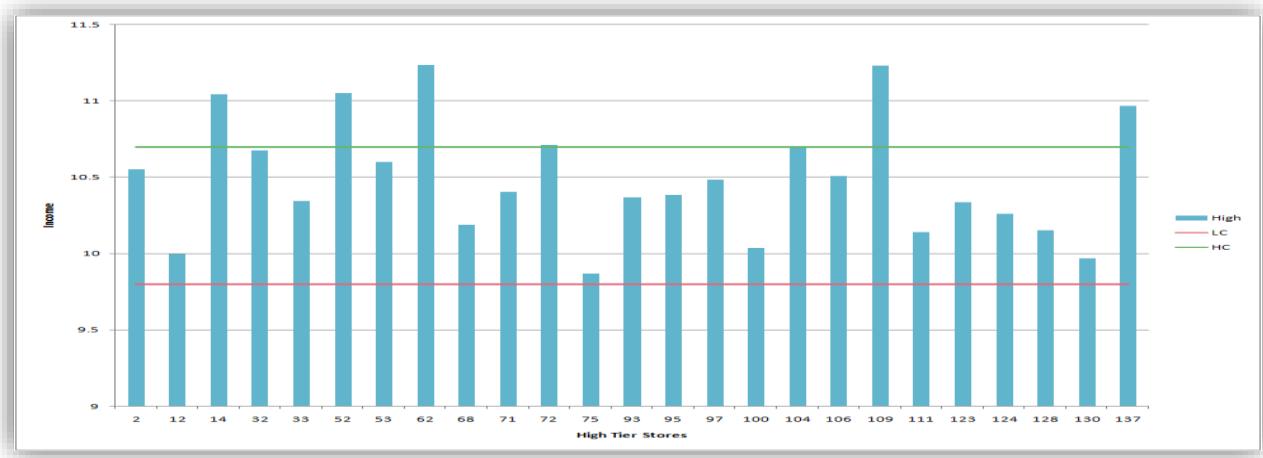
Pivot



Justification: Income of populations around medium tier stores show that there are many stores that cross 10.8 (lower range of high tier stores). These stores should be reclassified to high tier if other factors are minute. This will change the product price and product range according to high tier leading to increase in revenue.

High tier:

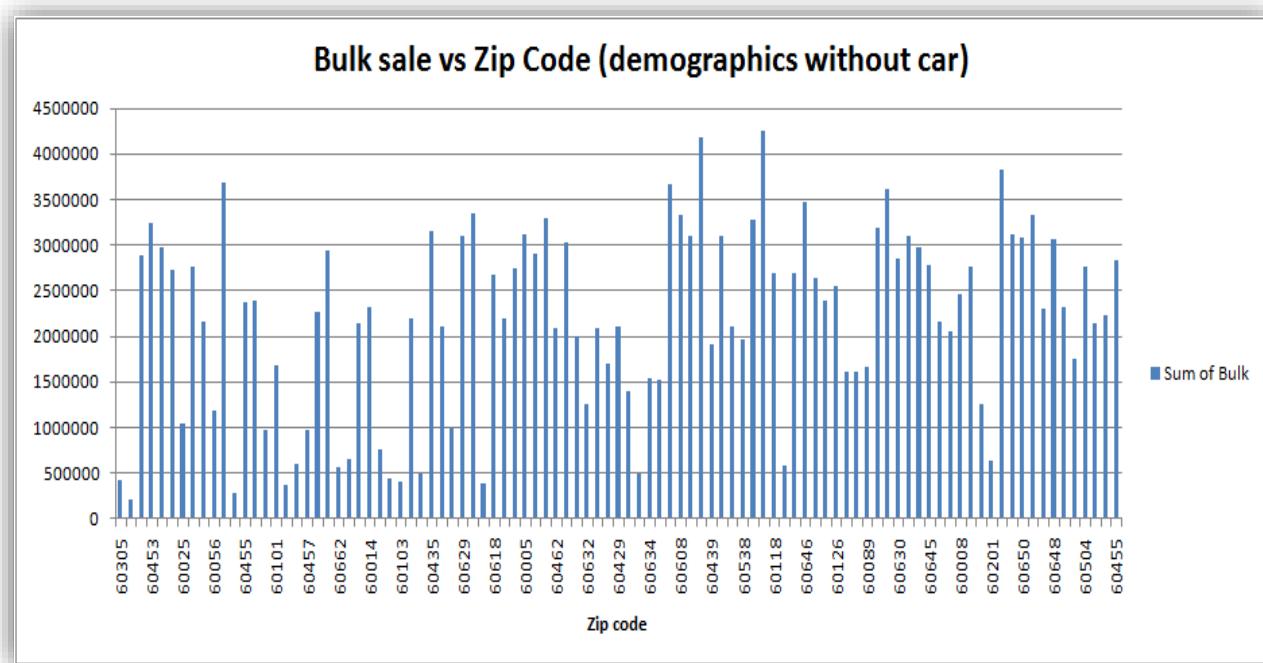
Pivot



Justification: Graph shows that store 75 and 130 are near 9.8 but are considered as high tier store. These should be re-categorized as low or medium stores.

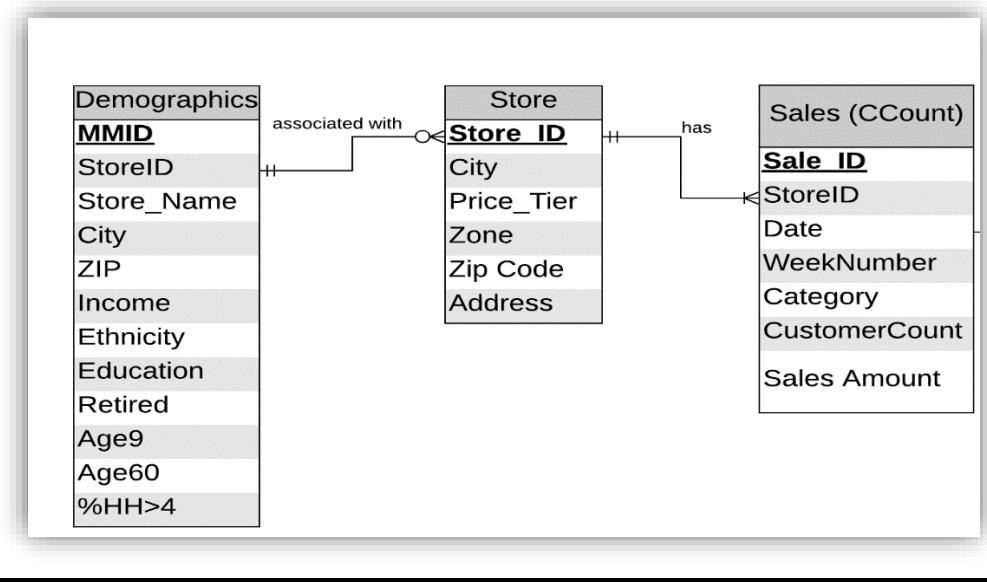
2.7 For the stores where the customer population without car is high, find out the areas in which sale of bulk products is high?

Pivot



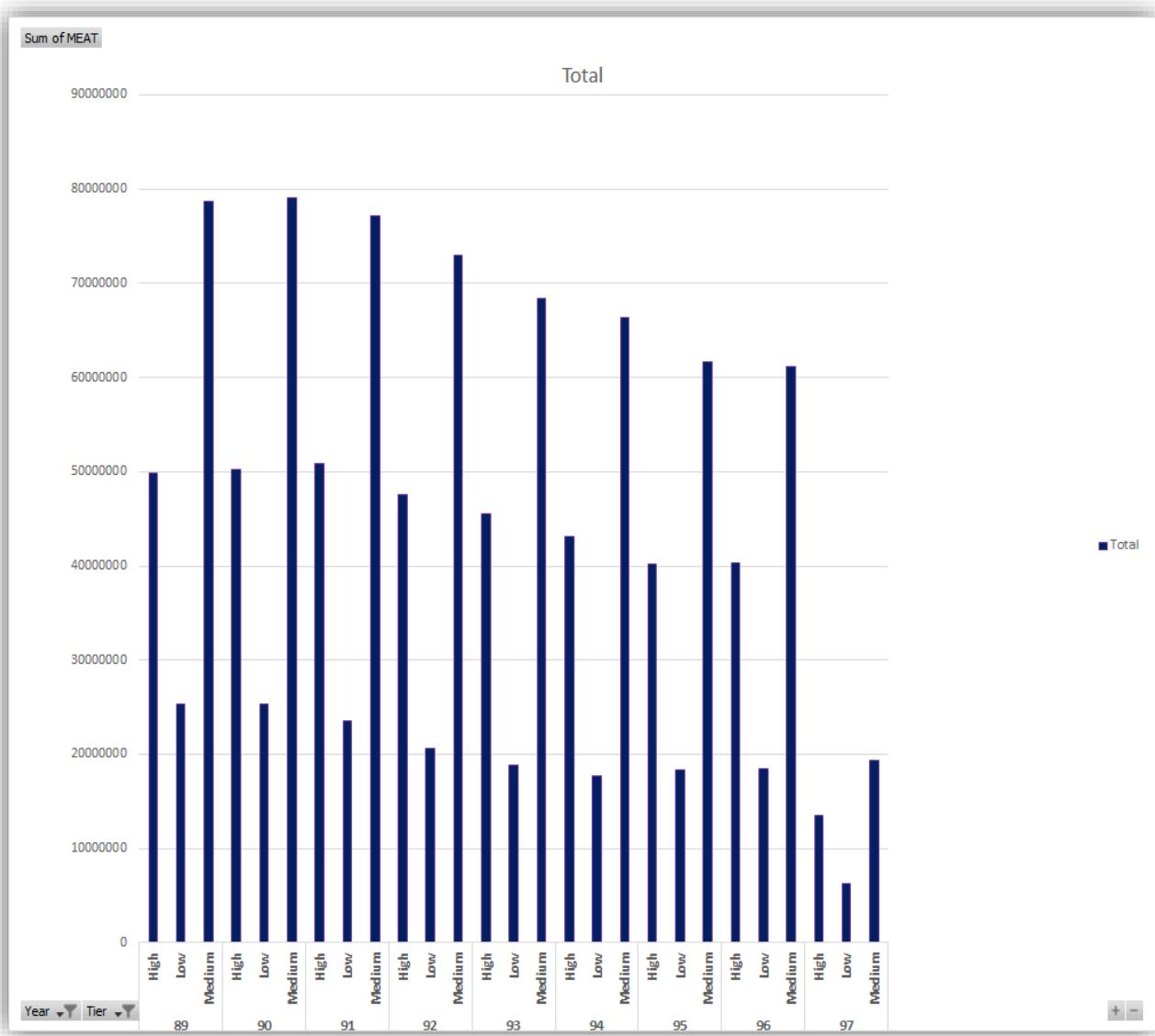
Justification: To compete with e-commerce Dominick should start home delivery options. To start with home delivery options, Dominick chain should start with the areas where number of people with no car are more and stores where bulk product sale is more. This data indicates that stores in zip code 60056, 60608, 60118, 60648 have high sales of bulk product with maximum population without car. Enabling home delivery will have positive impact on sales of these area.

ERD



2.8 What is the effect of bonus buy and simple discount on the sale of meat based on tiers?

Pivot

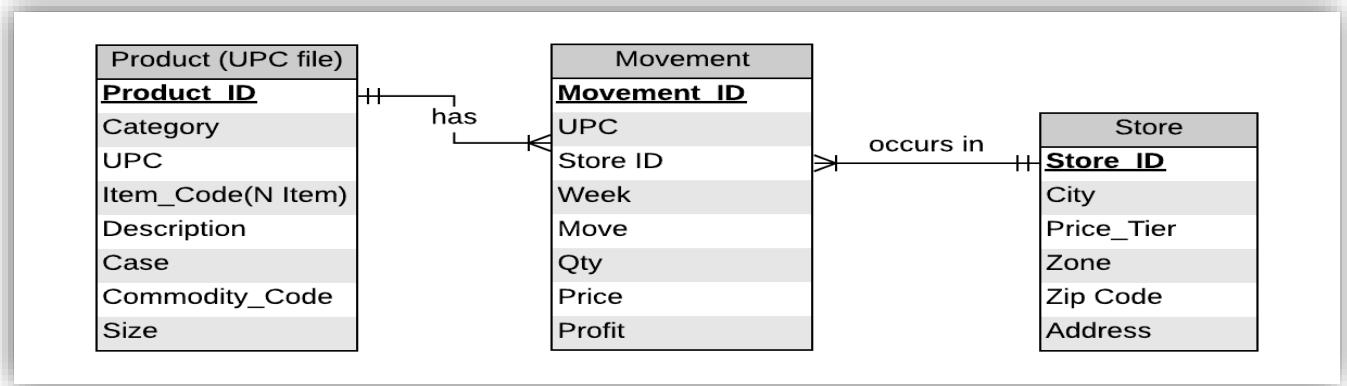


Justification:

The CCount file has data which tells us about the meat sales date-wise and week-wise. Our rationale to evaluate meat sales was to understand the impact of tiers on sale based on yearly data. We wanted to evaluate if there is any difference in the sales for high, medium tiers as meat is a popular category. On analysis, we found that the medium tier especially had high net sales as seen above. We expected the sales to be greater for high tier too. But compared to the high tier for all years, medium tier showed relatively high sales. Noticeably, we can see that there is a decline in

the sale over the years from 1998-1997. DFF can benefit by analyzing the cause of this decline and coming up with measures to make it an increasing trend over the next few years.

ERD



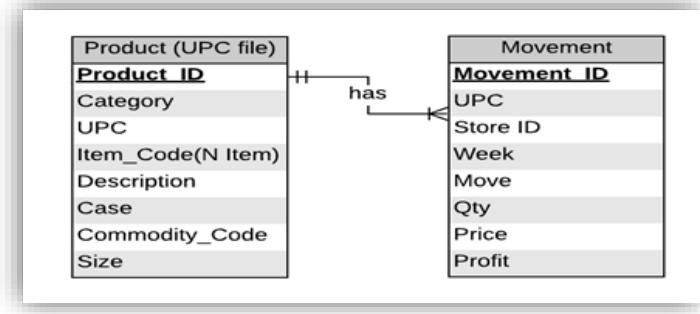
2.9 When toilet papers are bundled, then is there any difference in the sales and profit because of bundling?

Pivot



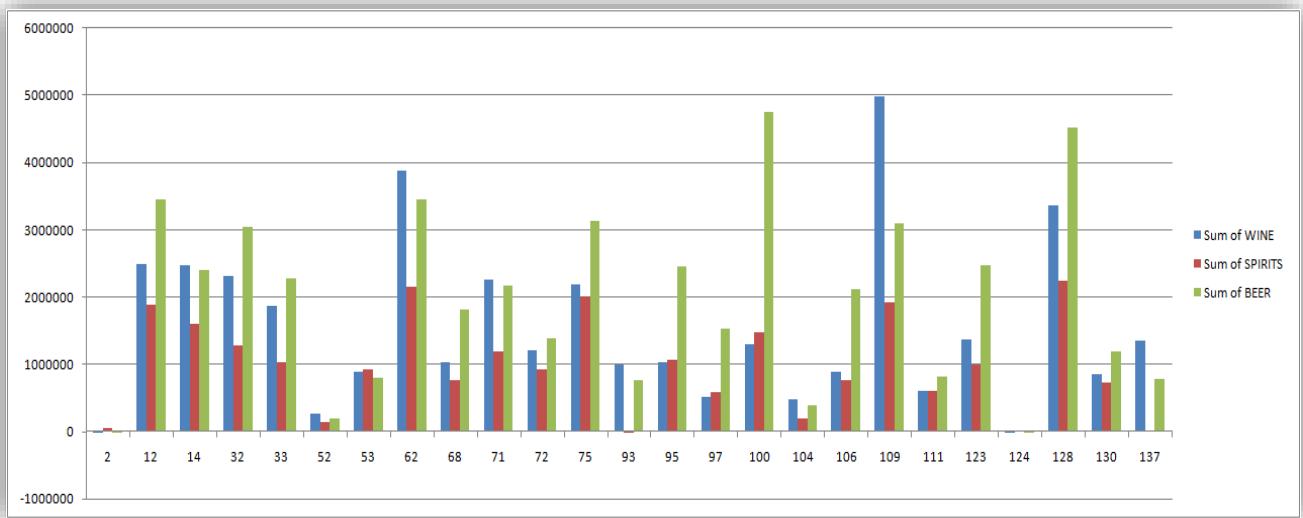
Justification: We have data for each category and bundles in which they are sold, and we think that bundling will have an impact on certain products. We took the example of toilet paper as usually customers tend to buy bundles of toilet paper in one go. When we performed analysis on bundled data for toilet papers, we discovered a trend which suggests that when toilet paper is bundled in a group of 3 then it has the highest sales and profits. DFF can increase their sale even more by focusing on the bundles of 3. Moreover, we observed that the bundles of 7 have relatively lower sales but high profits. So DFF can also focus on these bundles furthermore to increase their profitability.

ERD



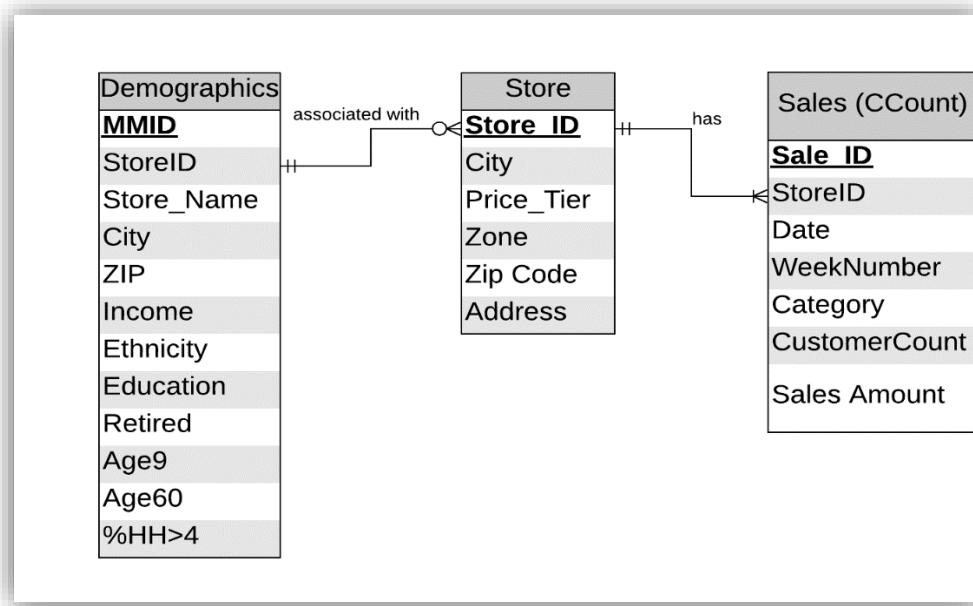
2.10 In high tier stores, find out which of the alcohol-based products dominate sales?

Pivot



Justification: In high tier stores, a major part of sales is due to beer (green) as compared to Wine & Spirit. In order to increase sales from high tier stores, where median income of population is high, stocks of high quality and expensive beer can be increased. This can increase revenue significantly.

ERD



3. Data marts and mapping tables

3.1 Kimball's Matrix for Data Marts:

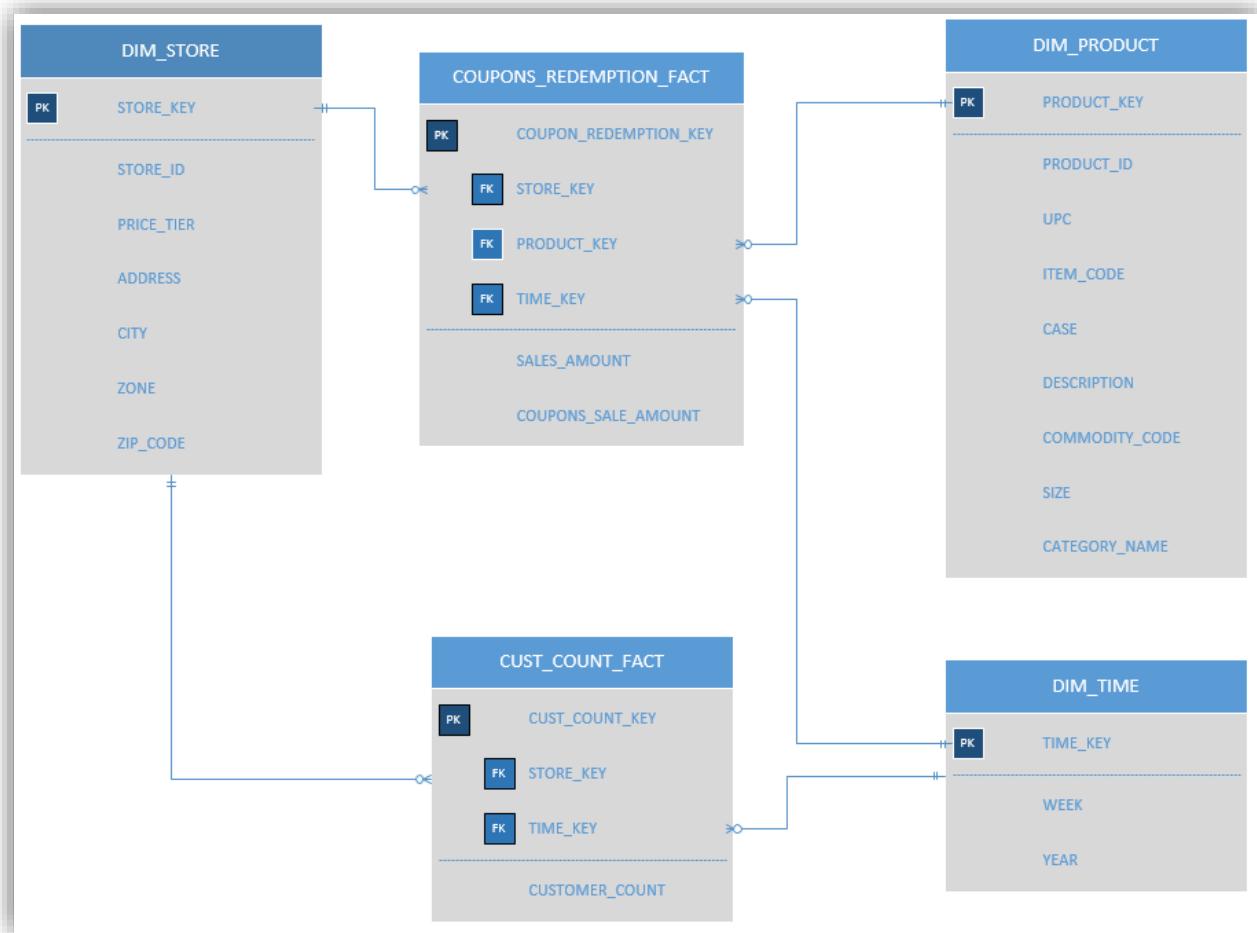
Data Marts	Time	Product	Store Demo	Demographic
Customer Count & Coupon Redemption Sales	X	X	X	
Product Sales	X	X	X	

The Data Mart Matrix

3.2 Data marts

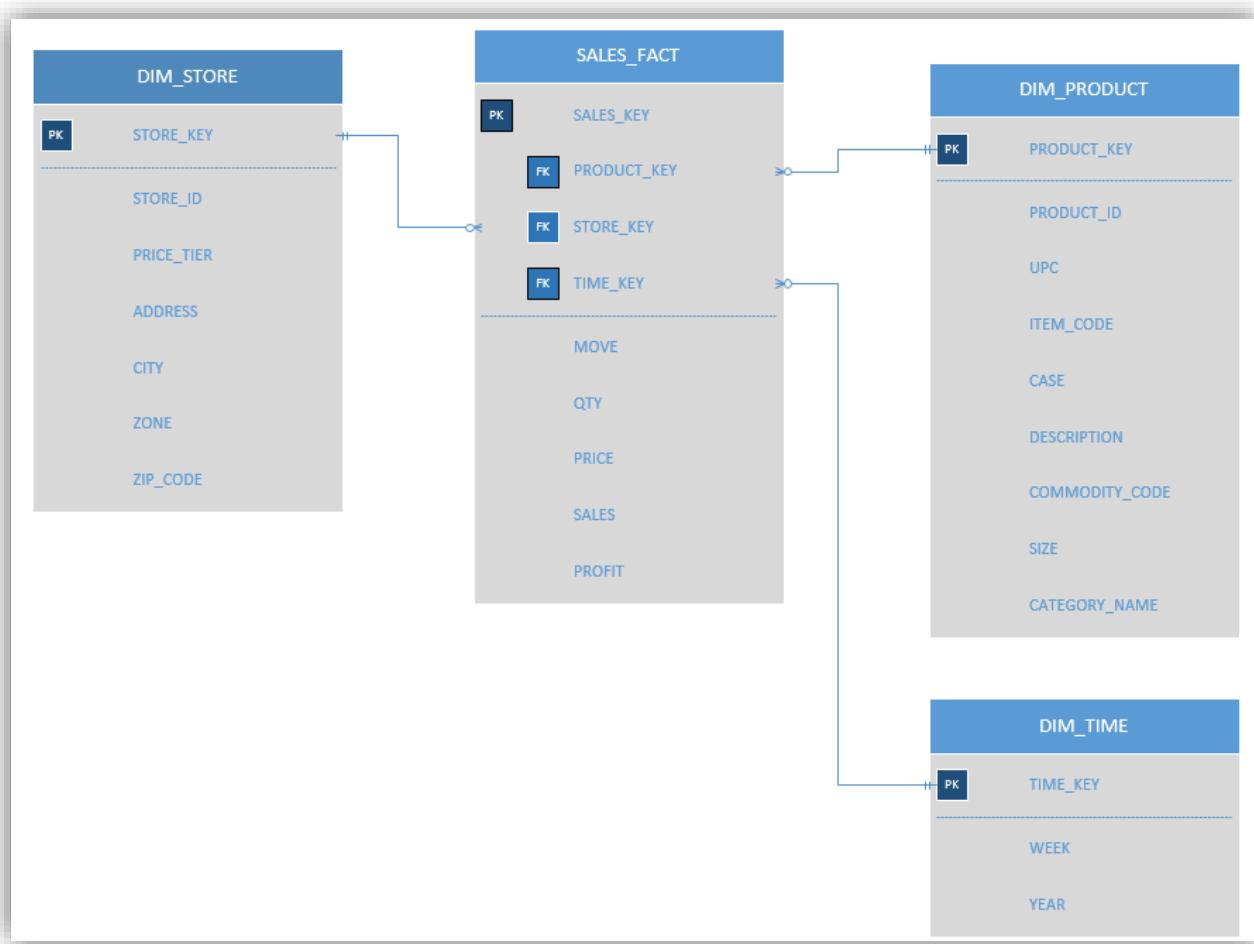
(Please refer to the business question 1-5 for the approach to develop below marts)

3.2.1 Coupon sales & Customer Count Data Mart



Schema Justification: The first data mart is created using the models created for Q1 & Q5. We combined the models to have two fact tables, one for the Customer Count related facts and the other for Sales amount related to coupons. Rationale behind keeping two fact table is that one question demands fact measured at customer level while other business question demands facts at store level. The other dimensions Time is constant.

3.2.2 Sales & Profit Data Mart:



Schema Justification: The remaining questions Q2, Q3, Q4 require us to calculate sales or profits for products or for a category. In essence, we need a sales fact table in all three questions. Since granularity and other factors were same, we merged all 3-fact table to create our second data mart with the fact table as **SALES_FACT** and related dimension tables: **DIM_STORE**, **DIM_PRODUCT** & **DIM_TIME**.

3.3 Mapping Tables:

STORE:

DW Target Table	DW Target Attribute	Target Datatype	Source Table	Source Column	Mapping Function	Others
DIM_STORE	STORE_KEY	Numeric				Surrogate key
DIM_STORE	STORE_ID	Numeric	Dominick's dataset-Store	STOREID		
DIM_STORE	PRICE_TIER	String	Dominick's dataset-Store	TIER		
DIM_STORE	ADDRESS	String	Dominick's dataset-Store	ADDRESS		
DIM_STORE	CITY	String		CITY		
DIM_STORE	ZONE	String		ZONE		
DIM_STORE	ZIP_CODE	Numeric		ZIP_CODE		

PRODUCT:

DW Target Table	DW Target Attribute	Target Datatype	Source Table	Source Column	Mapping Function	Others
DIM_PROD_UCT	PRODUCT_KEY	Numeric				Surrogate key
DIM_PROD_UCT	PRODUCT_ID	Numeric		Generated in ERD		Primary key
DIM_PROD_UCT	UPC	String	Dominick's dataset-Store	UPC		
DIM_PROD_UCT	ITEM_CODE	String	Dominick's dataset-Store	ITEM_CODE		
DIM_PROD_UCT	CASE	Numeric	Dominick's dataset-Store	CASE		
DIM_PROD_UCT	ZONE	String	Dominick's dataset-Store	ZONE		
DIM_PROD_UCT	DESCRIPTION	String	Dominick's dataset-Store	DESCRIPTION		
DIM_PROD_UCT	COMMODITY_CODE	String	Dominick's dataset-Store	COMMODITY_CODE		
DIM_PROD_UCT	SIZE	String	Dominick's dataset-Store	SIZE		
DIM_PROD_UCT	CATEGORY_NAME	String	Dominick's dataset-Store	CATEGORY_NAME		

TIME:

DW Target Table	DW Target Attribute	Target Datatype	Source Table	Source Column	Mapping Function	Others
DIM_TIME	TIME_KEY	Numeric				Surrogate key
DIM_TIME	WEEK	Numeric	Dominick's dataset-Store	WEEK_NUMBER		
DIM_TIME	YEAR	Numeric	Dominick's dataset-Store	YEAR		

SALES_FACT:

DW Target Table	DW Target Attribute	Target Datatype	Source Table	Source Column	Mapping Function	Others
SALES_FACT	SALES_KEY	Numeric				
SALES_FACT	PRODUCT_KEY	Numeric	DIM_PRODUCT	PRODUCT_KEY		
SALES_FACT	STORE_KEY	Numeric	DIM_STORE	STORE_KEY		
SALES_FACT	MOVE	Numeric	Dominick's dataset-Store	MOVE	sum(move) from movement group by week	
SALES_FACT	QTY	Numeric	Dominick's dataset-Store	QTY	sum(qty) from movement group by week	
SALES_FACT	PRICE	Numeric	Dominick's dataset-Store	PRICE	sum(price) from movement group by week	
SALES_FACT	SALES	Numeric	Dominick's dataset-Store	SALES	sum(sales) group by week	

COUPONS_REDEMPTION_FACT:

DW Target Table	DW Target Attribute	Target Datatype	Source Table	Source Column	Mapping Function	Others
COUPONS_R_EDEEMPTION_FACT	COUPON_REDEMPTION_KEY	Numeric				Primary key
COUPONS_R_EDEEMPTION_FACT	PRODUCT_KEY	Numeric	DIM_PRODUCT	PRODUCT_KEY		
COUPONS_R_EDEEMPTION_FACT	STORE_KEY	Numeric	DIM_STORE	STORE_KEY		
COUPONS_R_EDEEMPTION_FACT	TIME_KEY	Numeric	DIM_TIME	TIME_KEY		
COUPONS_R_EDEEMPTION_FACT	SALES_AMOUNT	Numeric	Dominick's dataset-Store	SALES_AMOUNT	Sum (sales_amount) from sales_table/ccou	

					nt group by week	
COUPONS_R EDEEMPTIO N_FACT	COUPON_SAL E_AMOUNT	Numeric	Dominick's dataset-Store	SALES_AMOUNT	Sum (sales_amount_ from sales_table/ccou nt where category group by week	

CUST_COUNT_FACT:

DW Target Table	DW Target Attribute	Target Datatype	Source Table	Source Column	Mapping Function	Others
CUST_COU NT_FACT	CUST_COUNT_KEY	Numeric				Primary key
CUST_COU NT_FACT	STORE_KEY	Numeric	DIM_STORE	STORE_KEY		
CUST_COU NT_FACT	TIME_KEY	Numeric	DIM_TIME	TIME_KEY		
CUST_COU NT_FACT	CUSTOMER_COUNT	Numeric	SALES/ MOVEMENT	SALES		

4. ETL Development Plan and Implementation

4.1 Data Quality

Data which is fit for the purpose it is intended for is called data quality. The data in DFF data sets had many quality issues. The data after being extracted into the staging area had data type varchar (255)/nvarchar (255). There were lot of junk data cells which has special characters in columns which should be numeric in value. There were few missing values such as week 0 had to be added later on to make the data consistent with the data in Movement and CCount file data. The data format in the input file is '890301'. This had to be made consistent throughout to a standard date format '1989-03-01'. There was missing data with many of the fields blank in the files. For example, the first 1155 rows of CCount consists of gibberish values or blank rows. The data in Movement and CCount has values for StoreId which are greater than 139. For this case, the Store table has no data after StoreId after 139. So, all the data in CCount and Movement which has StoreId>139 will not have referential integrity with the Store table. Few StoreIds, zip code and price tiers were also missing in the Store table. The information in Store table is valuable for business and any missing values definitely reduces the data quality.

4.2 Development of ETL Plan

Our development plan includes target data, all data sources, data mappings, extraction rules, transformation rules, cleaning rules and aggregated rules.

4.2.1 Target Data

Our solution for the data warehouse consists of 3 fact tables and 3 dimension tables. The target data is shown below:

Dimension Tables:

Target table	Target Column	Target Column Datatype
602_Group6_Warehouse.DIM_STORE	STORE_KEY	INT
	STORE_ID	INT
	PRICE_TIER	VARCHAR
	ADDRESS	VARCHAR
	CITY	VARCHAR
	ZONE	VARCHAR
	ZIP_CODE	VARCHAR

Target table	Target Column	Target Column Datatype
602_Group6_Warehouse.DIM_WEEK	WEEK_KEY	INT
	WEEK_NUMBER	INT
	START	DATE
	END	DATE
	SPECIAL_EVENTS	VARCHAR

Target table	Target Column	Target Column Datatype
602_Group6_Warehouse.DIM_PRODUCT	PRODUCT_KEY	INT
	PRODUCT	INT
	ITEM_CODE	INT
	UPC	INT
	SIZE	VARCHAR
	CASE	INT
	CATEGORY	VARCHAR
	DESCRIPTION	VARCHAR
	COMMODITY_CODE	INT

Fact Tables:

Target table	Target Column	Target Column Datatype
602_Group6_Warehouse.SALES_MOVE_FACT	SALES_MOVE_KEY	INT
	PRODUCT_KEY	INT
	WEEK_KEY	INT
	STORE_KEY	INT
	MOVE	INT
	QTY	INT
	PRICE	FLOAT
	SALE	VARCHAR
	PROFIT	FLOAT
	CALCULATED_SALES_AMOUNT	FLOAT
	CALCULATED_PROFIT	FLOAT

Target table	Target Column	Target Column Datatype
602_Group6_Warehouse.CUST_COUNT_FACT	CUST_COUNT_KEY	INT
	CUSTOMER_COUNT	INT
	WEEK_KEY	INT
	STORE_KEY	INT

Target table	Target Column	Target Column Datatype
602_Group6_Warehouse.COUPON_REDEMPTION_FACT	COUPON_REDEMPTION_KEY	INT
	STORE_KEY	INT
	WEEK_KEY	INT
	PRODUCT_KEY	INT
	SALES_AMOUNT	FLOAT

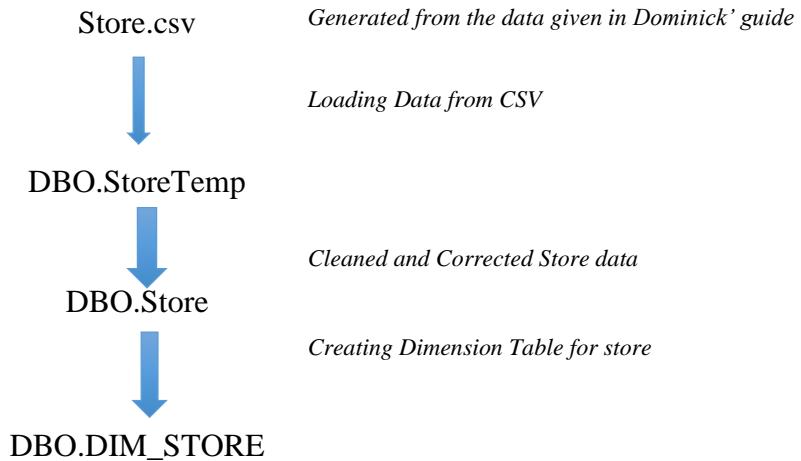
Data Sources

The data sources are the files given in the Dominick data set. For the business questions that we need to focus on, we used the CSV files provided which are CCount, Movement, UPC and Store files.

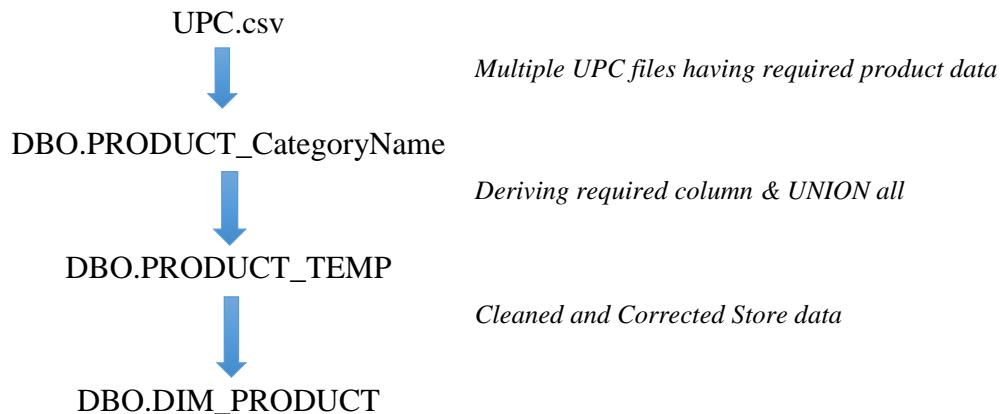
For one business question, we require data of customer count. This data is available in the CCount file. Another question required us to import data from CCount because we needed to find the sales data for coupons redemption for a category. We will use the columns needed for these questions from CCount file. For three questions, we need the sales and profit, so we will make use of the Movement and UPC files for this. Additionally, we will use the Store table given in the Dominick data manual to create a store dimension as we have questions which use the store data.

4.3 Data Mapping

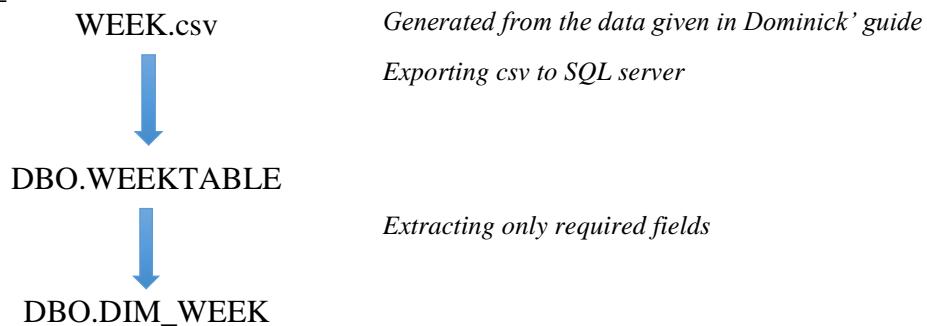
DIM_STORE



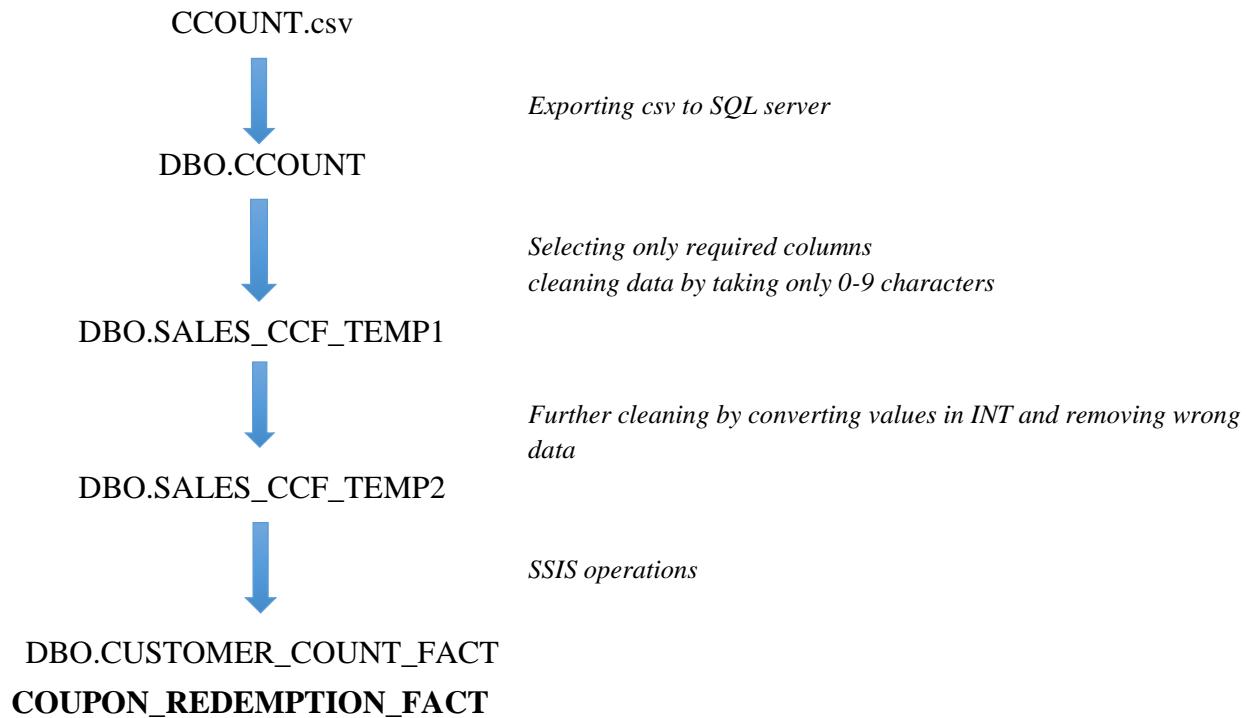
DIM_PRODUCT



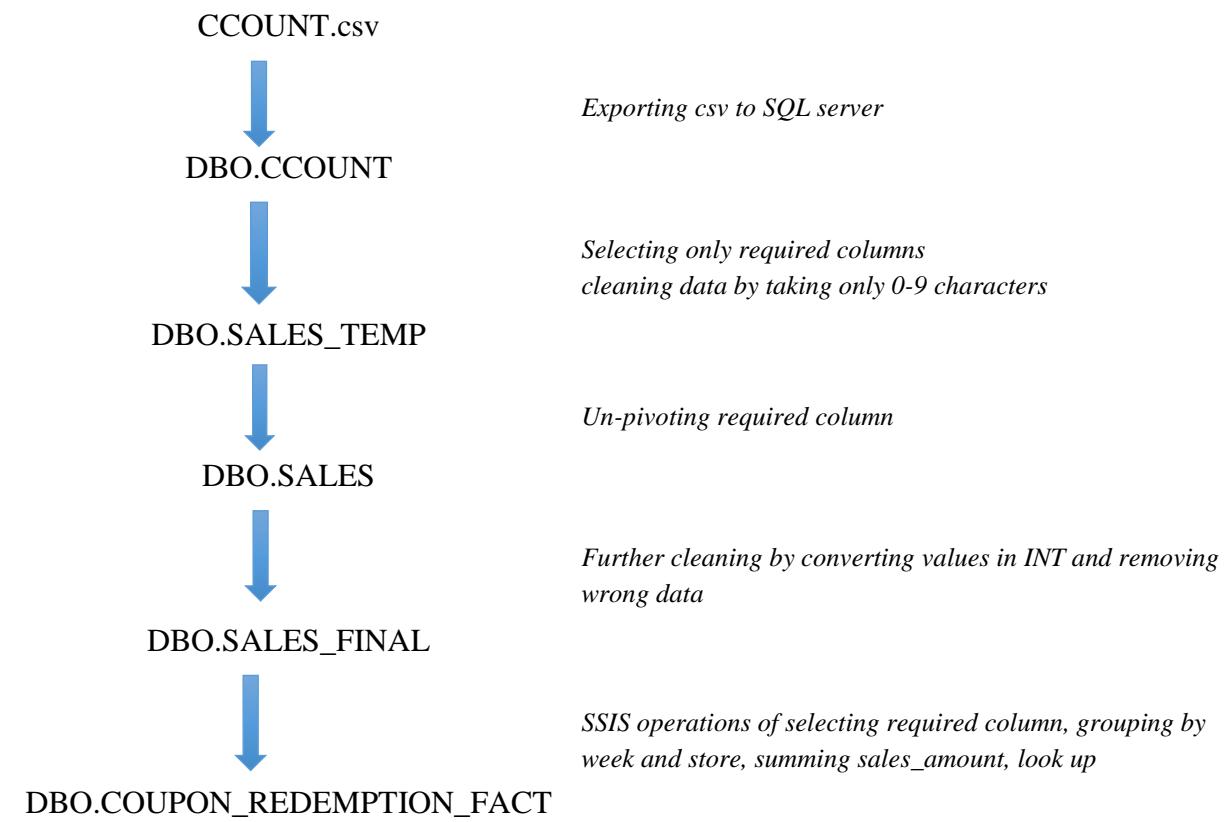
DIM_WEEK



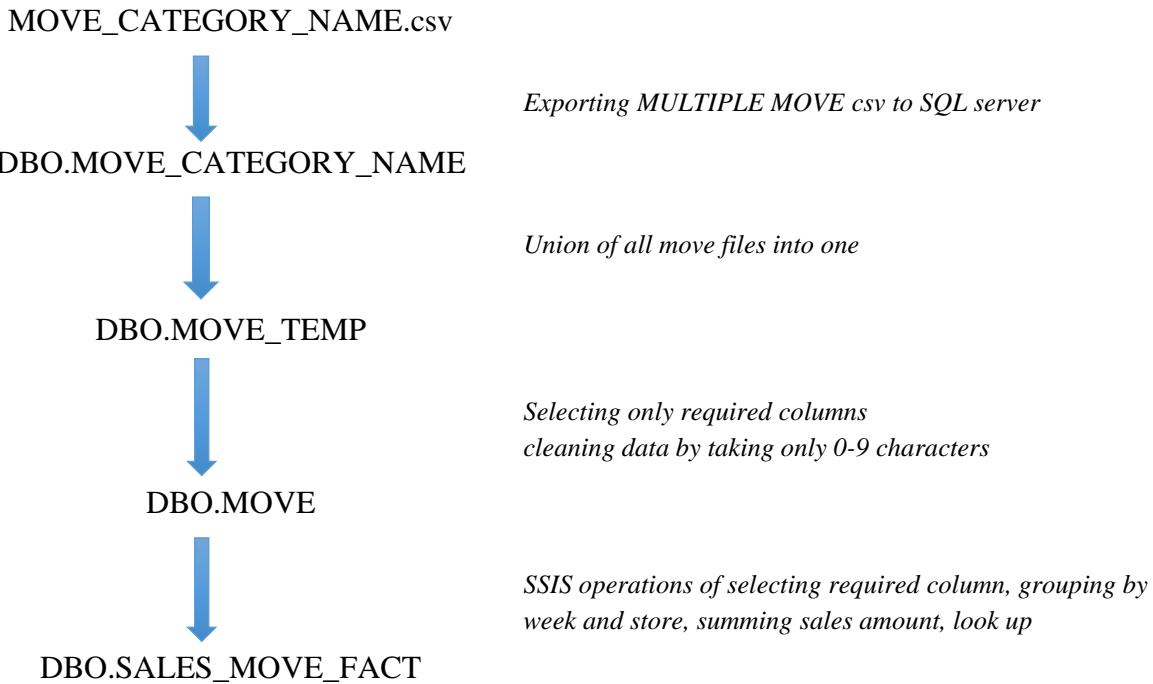
CUSTOMER_COUNT_FACT



COUPON_REDEMPTION_FACT



SALES_MOVE_FACT



4.4 Data Extraction rules

The extraction process will involve loading data from the source files to the staging area. Since we are not going to take care of updates to the data, we are going to use the Full Extraction process. We will take the CCOUNT file and directly import it to the staging area as CCOUNT. For the UPC and Movement files, we will take the files we need for our questions and load them into the staging area. For example, for frozen entrees we will load the UPC file into a table called *product_frozenentree* and movement table for frozen entrée products into the *move_frozenentree* table into the staging area. After this, we will combine all such *product_xxx* tables into a final PRODUCT table and all such *move_xxx* tables into final SALES table. The extraction of all the store details will be stored in the STORE table. The week and date table provided to us will be loaded into the week table in the staging area.

Additionally, while exporting data from CSV we made sure that the column names are not in “ ”.

Following is the raw data after loading CSVs into Staging area:
CCount

	DATE	GROCERY	DAIRY	FROZEN	BOTTLE	MVPCLUB	GROCCOUPL	MEAT	MEATFROZ	MEATCOUP	FISH	FISHCOUP	PROMO	PROMOCOUP	PRODUCE	BULK	SALADBAR	PRODCOUP	BULKCOUP	SALCOUP	FLORAL	FLO
1	920712	14347.01	4201.57	2996.52	-1.6	11.98	-100.85	3849.39	272.54	-6.57	259.86	0	230.39	-72.7	4963.31	237.01	1.21	0	0	0	230.97	0
2	920713	12309.58	3335.8	2367.1	-10.1	5.99	-123.02	3272.53	156.04	0	266.22	0	54.59	-17.8	3797.05	202.25	17.31	0	0	0	95.78	0
3	920714	10454.18	2979.68	2344.59	0	21.87	-170.83	3034.77	153.32	-2.79	363.4	0	148.89	-38.1	3364.55	318.44	7.09	0	0	0	149.09	0
4	920715	11245.74	3110.77	2379.47	-0.8	19.96	-130.48	3175.7	180.71	-5.58	380.53	0	98.47	-47.89	3530.08	186.79	15.58	0	0	0	163.72	0
5	920716	12216.94	3378.75	2507.6	-0.7	29.95	-75.93	3579.46	246.83	0	729.27	0	249.54	-97.9	3764.57	391.32	7.59	0	0	0	260	0
6	920717	13195.79	3607.79	2702.63	0	92.99	-87.25	3658.12	180.66	0	668.13	0	268.27	-63.6	4230.81	343.99	5.83	0	0	0	433.55	0
7	920718	16207.97	4347.12	3223.69	-4.8	81.99	-91.34	4904.44	234.73	-2.18	524.49	0	62.01	-9.9	4955.34	435.78	9.05	0	0	0	380.85	0
8	920719	14581.12	4118.43	2992.9	-2.4	81.41	-65.45	3936.24	281.79	-2.18	312.95	0	37.64	0	4890.12	366.32	14.21	0	0	0	382.79	0
9	920720	11606.58	3435.3	2092.76	-5.6	134.81	-56.71	3319.94	193.7	0	406.05	0	44.82	-14.4	3824.7	254.9	5.49	-0.9	0	0	284.98	0
10	920721	11744.11	3607.61	2445.6	-1.5	10.99	-48.56	3452.33	225.75	0	419.84	0	150.19	-57.2	3488.18	268.7	5	0	0	0	274.03	0
11	920722	10533.25	2924.68	2063.4	-1.6	18.47	-44.2	2724.83	174.86	-4.36	478.94	0	102.74	-43.5	3057.63	261.59	11.42	0	0	0	186.38	0
12	920723	13613.08	3603.18	2450.4	-1.8	21.98	-82.13	4759.88	297.31	-8	629.45	0	214.15	-58.8	4501.6	349.57	12.61	0	0	0	368.33	0
13	920724	14302.29	3512.18	2681.97	-0.8	37.05	-110.64	4798.56	299.84	-9	794.37	0	123.25	-32.4	4718.09	303.8	7.83	0	0	0	225.24	0
14	920725	19007.5	4733.75	3483.44	-4.3	55.94	-118.96	5887.19	477.87	-5	777.28	0	157.89	-26.4	5792.8	372.8	7.29	0	0	0	330.92	0
15	920726	15544.37	4091.81	3019.99	-3.9	12.49	-91.87	4172.79	302.66	-4	236.82	0	81.17	-20.8	4995.65	285.58	7.82	0	0	0	368.8	0
16	920727	12347.84	3381	2400.59	0	17.36	-72.39	3679.27	231.67	-8	246.68	0	87.96	-20.8	4075.39	252.82	13.59	0	0	0	182.48	0
17	920728	11051.37	3092.88	2508.87	-4.7	67.17	-56.95	3039.53	259.45	-7.79	568.32	0	32.68	-14.4	3934.39	199.46	6.9	0	0	0	205.71	0
18	920729	10572	2643	330.0	0	0	0	3167	0	0	811	0	0	0	2778	0	0	0	0	0	169	0
19	920730	11936.06	3131.25	2345.58	0	21.87	-104.94	3600.69	224.75	-2	854.35	0	39.77	-14.8	3665.88	216.24	5.84	0	0	0	196.61	-51
20	920731	14683.97	3464.56	2611.78	0	14.58	-159.74	3741.54	265.7	0	932.37	0	148.72	-10.4	4556.2	286.12	6.18	0	0	0	373.11	-88
21	920801	18081.58	4204.34	3191.35	-4.1	21.87	-141.24	5523.07	268.22	-4	770.22	0	130.43	-37.1	5139.35	306.01	19.25	0	0	0	533.88	-13
22	920802	16676	3982	4231	0	0	0	4103	0	0	142	0	0	0	3873	0	0	0	0	0	215	0
23	920803	12514.36	3164.93	2155.72	-1.6	7.29	-104.34	3299.55	209.22	0	496.35	0	42.89	-25.5	3948.05	220.12	8.56	0	0	0	253.49	-76
24	920804	11279	2776	3297	0	0	0	3285	0	0	400	0	0	0	2933	0	0	0	0	0	147	0
25	920805	11626.82	2847.31	2136.72	-1.6	11.98	-53.21	3318.23	213.56	-2	586.78	0	62.91	-10	3512.3	303.22	8.34	0	0	0	201.57	-25
26	920806	12147.67	4346.25	2371.89	0	0	-705.57	3666.33	157.25	-2.79	749.26	0	600.47	-297.9	4270.81	278.17	12.14	0	0	0	165.54	0
27	920807	12544.97	4229.57	2415.15	-0.8	0	-625.58	3760.69	198.84	0	566.53	0	574.74	-347.4	4335.91	266.42	17.87	0	0	0	166.17	0
28	920808	18011.41	5716.3	3714.65	0	0	-716.12	5510.15	297.83	-13.95	739.76	0	650.42	-339.5	5925.87	276.61	13.19	0	0	0	325.84	0
29	920809	12568.89	4260.32	2992.76	-2.4	19.17	-413.07	3437.99	253.37	0	157.11	0	264.9	-161.6	4830.1	199.39	12.53	0	0	0	248.85	0
30	920810	11663.66	3875.41	2387.86	-0.2	6.39	-395.26	3012.17	229.02	-2.79	288.73	0	472.35	-264.8	3906.37	232.81	24.27	0	0	0	213.24	0
31	920811	11716.51	3753.71	2309.91	-0.2	0	-345.42	3134.59	231.05	-5.58	341.72	0	497.18	-334.9	3631.84	229.67	5.95	0	0	0	115.89	0
32	920812	11677.72	3860.84	2355.61	-0.6	19.17	-399.32	3101	195.36	-2.79	201.76	0	446.84	-338.7	3327.86	249.4	6.61	0	0	0	198.34	0
33	920813	13170.15	3366.49	2443.31	0	6.39	-70.17	4072.16	248.97	-2	634.97	0	103.31	-36	5360.36	265.18	2.85	0	0	0	268.72	0
34	920814	14489.94	3566.31	2716.32	0	6.39	-84.14	3921.16	319.45	0	719.59	0	105.41	-36.6	4880.32	261.69	10.84	0	0	0	275.43	0

move_frozenentree

STORE	UPC	WEEK	MOVE	QTY	PRICE	SALE	PROFIT	OK
1	134	1380010154	362	4	1	2.89	32.99	1
2	134	1380010154	363	1	1	2.89	32.76	1
3	134	1380010154	364	4	1	2.89	34.75	1
4	134	1380010154	365	29	1	1.93	S 12.02	1
5	134	1380010154	366	1	1	2.89	41.24	1
6	134	1380010154	367	3	1	2.89	41.23	1
7	134	1380010154	368	7	1	1.93	S 11.96	1
8	134	1380010154	369	3	1	2.89	41.21	1
9	134	1380010154	370	1	1	2.89	32.56	1
10	134	1380010154	371	2	1	2.89	32.56	1
11	134	1380010154	372	2	1	2.89	43.66	1
12	134	1380010154	373	2	1	2.89	43.66	1
13	134	1380010154	374	3	1	2.89	36.85	1
14	134	1380010154	375	6	1	2.89	36.85	1
15	134	1380010154	376	3	1	2.89	33.42	1
16	134	1380010154	377	0	1	0	0	1
17	134	1380010154	378	3	1	2.89	33.07	1
18	134	1380010154	379	2	1	2.89	33.07	1
19	134	1380010154	380	2	1	2.89	38.23	1
20	134	1380010154	381	24	1	1.93	B 17.11	1
21	134	1380010154	382	14	1	1.93	B 17.15	1
22	134	1380010154	383	19	1	1.98	B 19.26	1
23	134	1380010154	384	4	1	2.89	B 44.67	1
24	134	1380010154	385	1	1	2.89	44.67	1
25	134	1380010154	386	23	1	2.89	44.67	1
26	134	1380010154	387	5	1	2.89	44.67	1
27	134	1380010154	388	3	1	2.89	44.67	1
28	134	1380010154	389	3	1	2.89	44.67	1
29	134	1380010154	390	8	1	2.89	44.67	1
30	134	1380010154	391	30	1	2.5	S 36.04	1
31	134	1380010154	392	2	1	2.89	S 44.67	1
32	134	1380010154	393	0	1	0	0	1
33	134	1380010154	394	2	1	2.89	44.67	1
34	134	1380010154	395	2	1	2.89	44.67	1

product_frozenentree

	COM_CODE	UPC	DESCRIP	SIZE	CASE	NITEM
1	105	1242409401	WOLFGANG PUCK 4 CHS	13.5	8	9310050
2	105	1242409402	WOLFGANG PUCK MSH/SP	13.5	8	9310060
3	105	1242409403	WOLFGANG PUCK CHN RA	13.5 O	8	9310010
4	105	1242409404	WOLFGANG PUCK SWTPPO	13.5 O	8	9310020
5	105	1242409406	WOLFGANG PUCK MSH SP	15 OZ	8	9310090
6	105	1242409407	WOLFGANG PUCK EGPLNT	15 OZ	8	9310080
7	105	1242409408	WOLFGANG PUCK SP CKN	13.5 O	8	9310030
8	105	1380010002	STFRS BEEF CHOP SUEY	12 OZ	12	9129041
9	105	1380010011	STOUFFER'S BEEF STR	9.7 OZ	12	9110430
10	105	1380010012	STOFR BEEF TERIYAKI	10 OZ	12	9111070
11	105	1380010017	STOFFERS CHILI W/BEA	8.75 O	12	9126951
12	105	1380010018	STFRS CRMD CHP BEEF	11 OZ	12	9124101
13	105	1380010019	STFRS IRZ CHILI REAL	9.2 OZ	12	912398
14	105	1380010021	STFR LXP ORIENTAL B	9.625	12	9120041
15	105	1380010023	STOUFFERS CHDR PASTA	11 OZ	12	9124741
16	105	1380010025	STFRS R C BEEF POT R	9.25 O	12	9134181
17	105	1380010032	STFRS GREEN PEPPER S	10.5 O	12	9124551
18	105	1380010036	STFRS R C BEEF DIJON	9.5 OZ	12	9134201
19	105	1380010037	STFRS R C BEEF CACCI	10.1 O	12	9134051
20	105	1380010040	STFRS TORTILLA GRAND	9.6 OZ	12	9124431
21	105	1380010045	STFRS H/S RIGATONI W	9 OZ	12	9124261
22	105	1380010050	LC BEEF/BEAN ENCHANA	9.25...	12	9310761
23	105	1380010055	STFRS IRZ JAMBALAYA	10 OZ	12	912400
24	105	1380010060	STOFR SALISBURY STEA	9.78 O	12	9129420
25	105	1380010065	STOFR RIBS OF BEEF	9 OZ	12	9129550
26	105	1380010066	STOUFFERS SNGL SERVE	10 OZ	12	9124311
27	105	1380010067	STFRS STUFFED PEPPER	15.5 O	12	9123901
28	105	1380010068	STFRS SWEDISH MTBALL	11 OZ	12	9129001
29	105	1380010070	STFRS H/S SALISBURY	8.63 O	12	9124191
30	105	1380010071	STFRS H/S VEAL PARMI	9.25 O	12	9124251
31	105	1380010072	STFRS H/S MEATLOAF	9.88 O	12	9124211
32	105	1380010073	STFRS H/S BEEF & NOO	7.13 O	12	9124231
33	105	1380010074	STFRS H/S BEEF POT R	9.5 OZ	12	9124221
34	105	1380010075	STFRS BEEF FAJITAS	6.9 OZ	12	9124501

WeekTable

	Week Number	Start	End	Special Events
1	1	9/14/1989	9/20/1989	
2	2	9/21/1989	9/27/1989	
3	3	9/28/1989	10/4/1989	
4	4	10/5/1989	10/11/1989	
5	5	10/12/1989	10/18/1989	
6	6	10/19/1989	10/25/1989	
7	7	10/26/1989	11/1/1989	Halloween
8	8	11/2/1989	11/8/1989	
9	9	11/9/1989	11/15/1989	
10	10	11/16/1989	11/22/1989	
11	11	11/23/1989	11/29/1989	Thanksgiving
12	12	11/30/1989	12/6/1989	
13	13	12/7/1989	12/13/1989	
14	14	12/14/1989	12/20/1989	
15	15	12/21/1989	12/27/1989	
16	16	12/28/1989	1/3/1990	Christmas
17	17	1/4/1990	1/10/1990	New-Year
18	18	1/11/1990	1/17/1990	
19	19	1/18/1990	1/24/1990	
20	20	1/25/1990	1/31/1990	
21	21	2/1/1990	2/7/1990	
22	22	2/8/1990	2/14/1990	
23	23	2/15/1990	2/21/1990	Presidents Day
24	24	2/22/1990	2/28/1990	
25	25	3/1/1990	3/7/1990	
26	26	3/8/1990	3/14/1990	
27	27	3/15/1990	3/21/1990	
28	28	3/22/1990	3/28/1990	Easter
29	29	3/29/1990	4/4/1990	
30	30	4/5/1990	4/11/1990	
31	31	4/12/1990	4/18/1990	
32	32	4/19/1990	4/25/1990	
33	33	4/26/1990	5/2/1990	
34	34	5/3/1990	5/9/1990	

Store

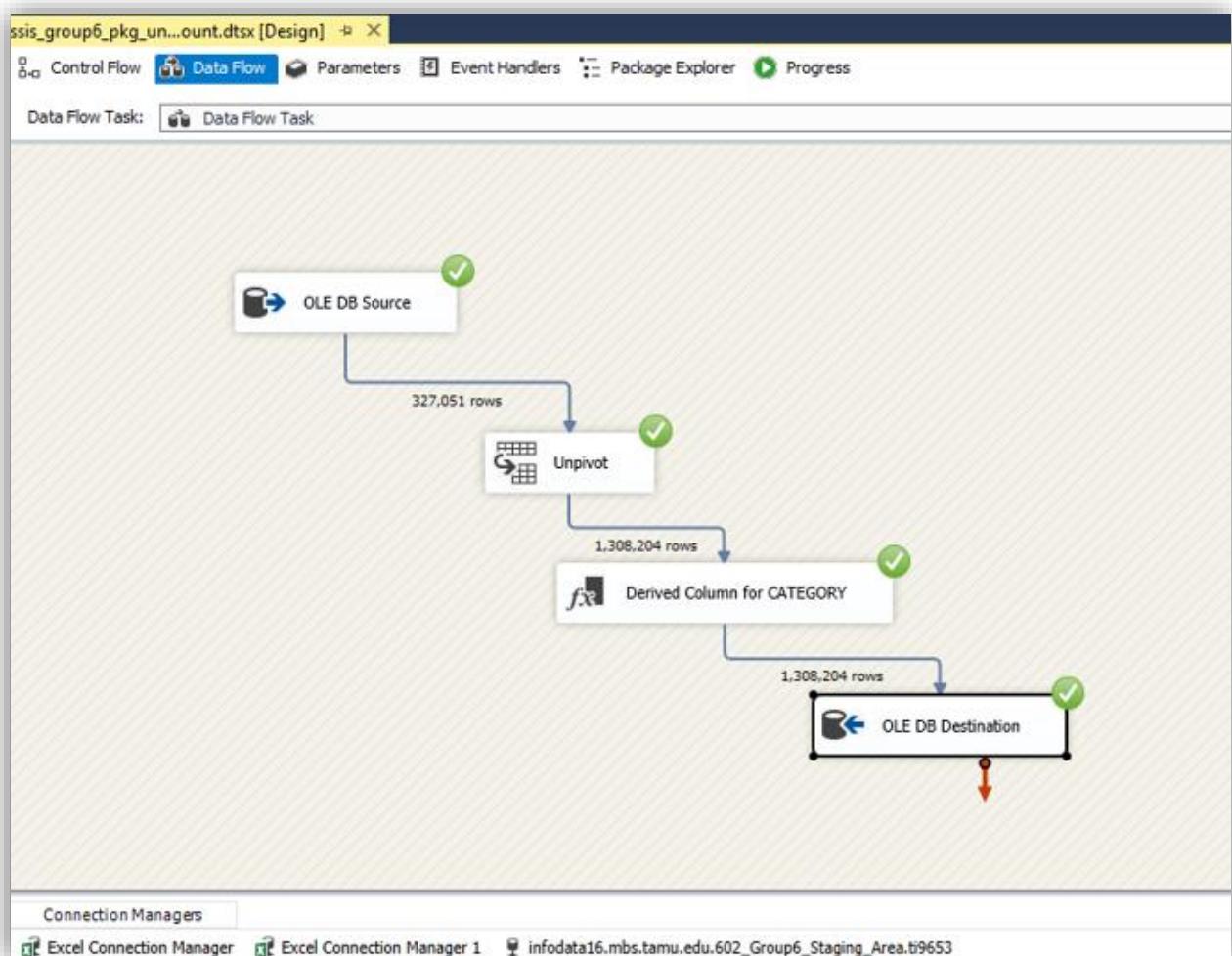
	Store	City	Price Tier	Zone	Zip Code	Address
1	2	River Forest	High	1	60305	7501 W. North Ave.
2	4	Park Ridge	Medium	2	60068	Closed
3	5	Palatine	Medium	2	60067	223 Northwest HWY.
4	8	Oak Lawn	Low	5	60435	8700 S. Cicero Ave.
5	9	Morton Grove	Medium	2	60053	6931 Dempster
6	12	Chicago	High	7	60660	6009 N. Broadway Ave.
7	14	Glenview	High	1	60025	1020 Waukegan Rd.
8	18	River Grove	Low	5	60171	8355 W. Belmont Ave.
9	19	Glen Ellyn	None	None	60137	Closed
10	21	Hanover Park	CubFighter	6	60103	1440 Irving Park Rd.
11	25	Chicago	None	None	60639	Closed
12	28	Mt. Prospect	Medium	2	60054	1145-55 Mt Prospect Pz.
13	32	Park Ridge	High	1	60068	1900 S. Cumberland Ave.
14	33	Chicago	High	7	60657	3012 N. Broadway Ave.
15	39	Waukegan	None	None	60085	Closed
16	40	Bridgeview	CubFighter	6	60455	8825 S. Harlem
17	44	Western Spring	Medium	2	60558	14 Garden Market
18	45	Wheeling	Medium	2	60090	550 W. Dundee Rd.
19	46	Carol Stream	Low	5	60187	Closed
20	47	Addison	Medium	2	60101	545 W. Lake St.
21	48	Schaumburg	Medium	2	60193	20 E. Golf Rd.
22	49	Downers Grove	Medium	2	60515	120 E. Ogden Ave.
23	50	Hickory Hills	Medium	2	60457	8631 W. 95th St.
24	51	Palos Heights	Medium	3	60463	6401 W. 127th St.
25	52	Northbrook	High	1	60062	4125 Dundee Rd.
26	53	Chicago	High	7	60662	3145 W. Pratt Ave.
27	54	Naperville	Medium	2	60540	1295 E. Ogden Ave.
28	55	Tinley Park	None	None	60577	Closed
29	56	Countryside	Medium	2	60525	6704 Joliet Rd.
30	59	Crystal Lake	CubFighter	6	60014	6000 Northwest Hwy.
31	60	Downers Grove	None	None	60515	Closed
32	62	Northfield	High	1	60093	1822 Willow Rd.
33	64	Villa Park	Medium	2	60181	302 W. North Ave.
34	65	Hanover Park	Medium	2	60103	Closed

4.5 Data Transformation and Cleansing rules

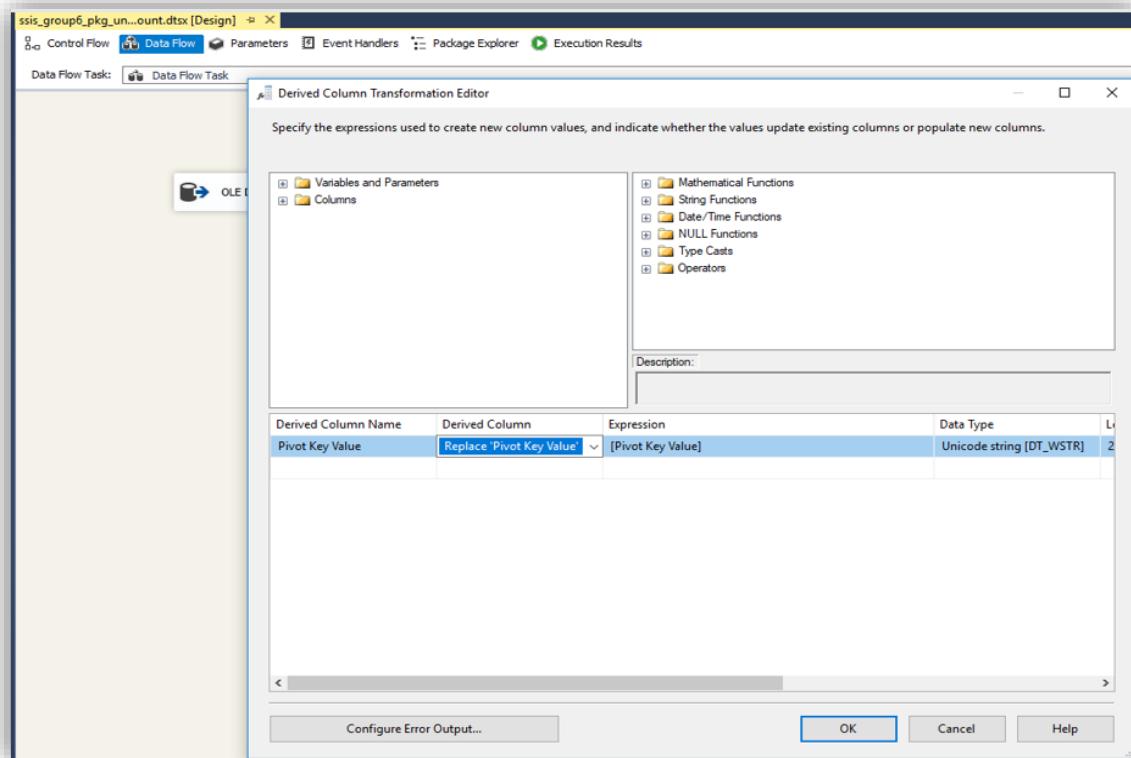
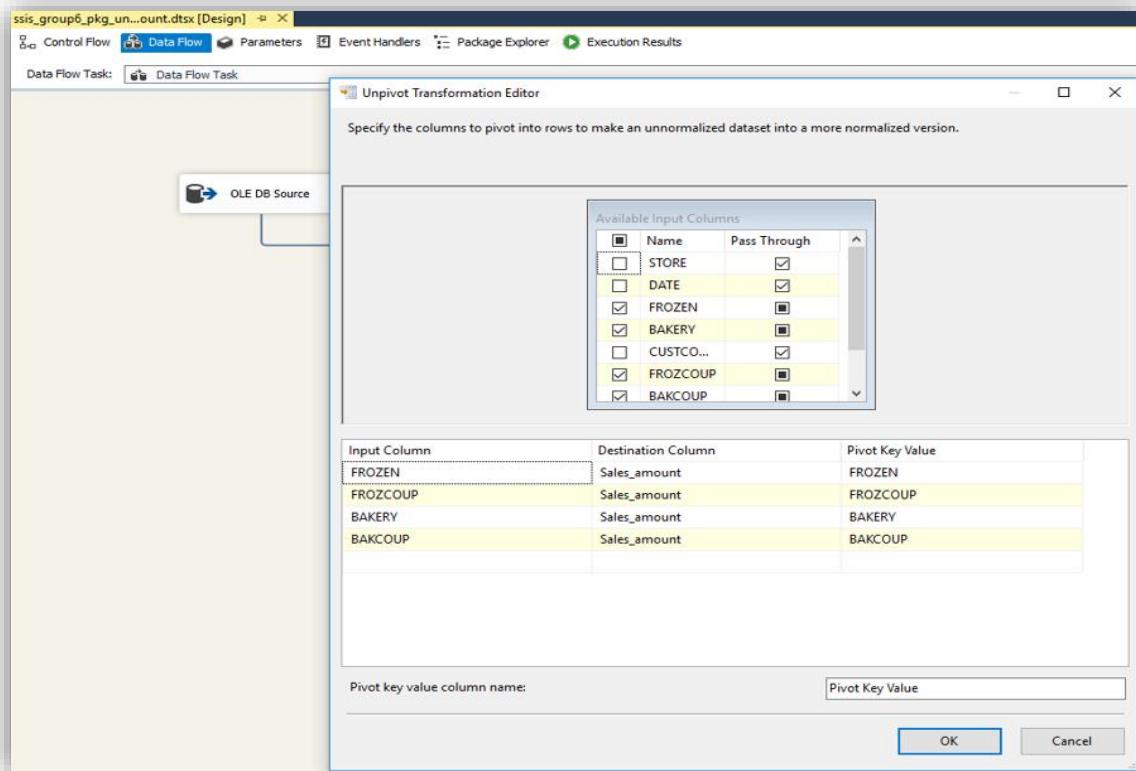
Transformation 1: For CCount, the data we need the data in the format as shown below:

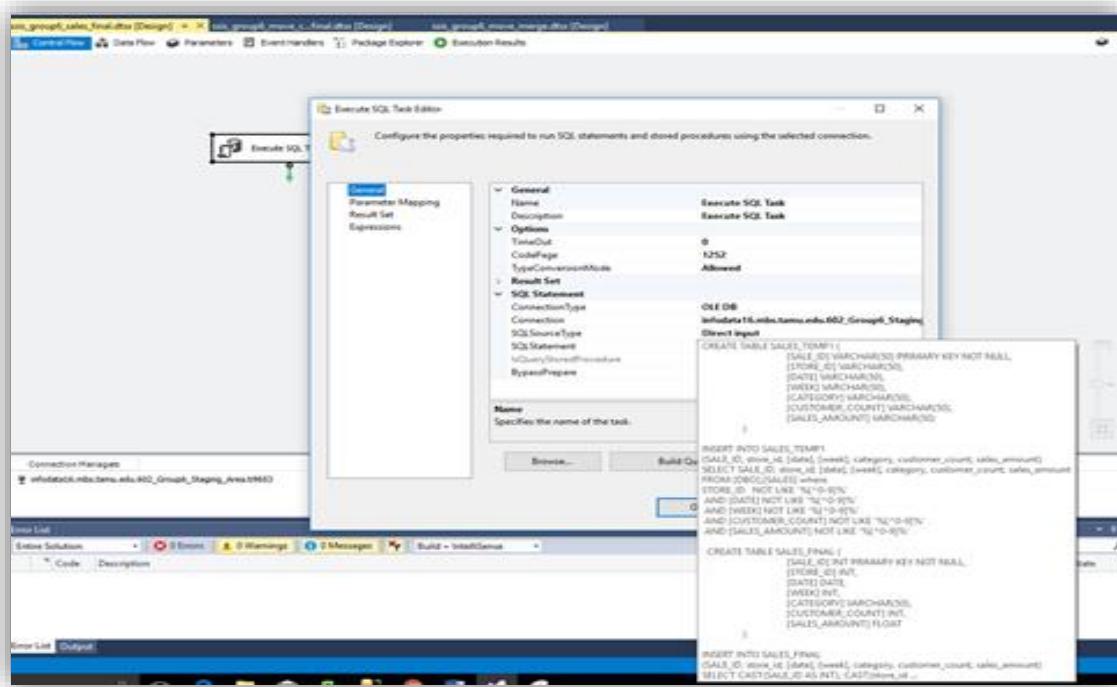
Sales ID	StoreID	Date	Category	Sales	Customer Count
1	47	880101	GROCERY	14900	1546
2	47	880101	DAIRY	3321	1546

For this purpose, we need to apply transformation to this *CCount*.



Next, we used derived column to store the category (i.e., column name in CCOUNT)





```

CREATE TABLE SALES_TEMP1 (
    [SALE_ID] VARCHAR(50) PRIMARY KEY NOT NULL,
    [STORE_ID] VARCHAR(50),
    [DATE] VARCHAR(50),
    [WEEK] VARCHAR(50),
    [CATEGORY] VARCHAR(50),
    [CUSTOMER_COUNT] VARCHAR(50),
    [SALES_AMOUNT] VARCHAR(50)
)

INSERT INTO SALES_TEMP1
(
    SALE_ID, store_id, [date], [week], category, customer_count, sales_amount)
SELECT SALE_ID, store_id, [date], [week], category, customer_count, sales_amount
FROM [DBO].[SALES] where
STORE_ID NOT LIKE '%[0-9]%'
AND [DATE] NOT LIKE '%[0-9]%'
AND [WEEK] NOT LIKE '%[0-9]%'
AND [CUSTOMER_COUNT] NOT LIKE '%[0-9]%'
AND [SALES_AMOUNT] NOT LIKE '%[0-9]%'
)

CREATE TABLE SALES_FINAL (
    [SALE_ID] INT PRIMARY KEY NOT NULL,
    [STORE_ID] INT,
    [DATE] DATE,
    [WEEK] INT,
    [CATEGORY] VARCHAR(50),
    [CUSTOMER_COUNT] INT,
    [SALES_AMOUNT] FLOAT
)

```

```

INSERT INTO SALES_FINAL
( SALE_ID, store_id, [date], [week], category, customer_count, sales_amount)
SELECT CAST(SALE_ID AS INT), CAST(store_id AS INT), convert(varchar(10), cast([DATE] as date), 101),
CAST([week] AS INT), category, CAST(customer_count AS FLOAT), CAST(sales_amount AS FLOAT)
FROM [DBO].[SALES_TEMP1] WHERE cast(STORE_ID as INT) <140

```

Transformation 2: Store

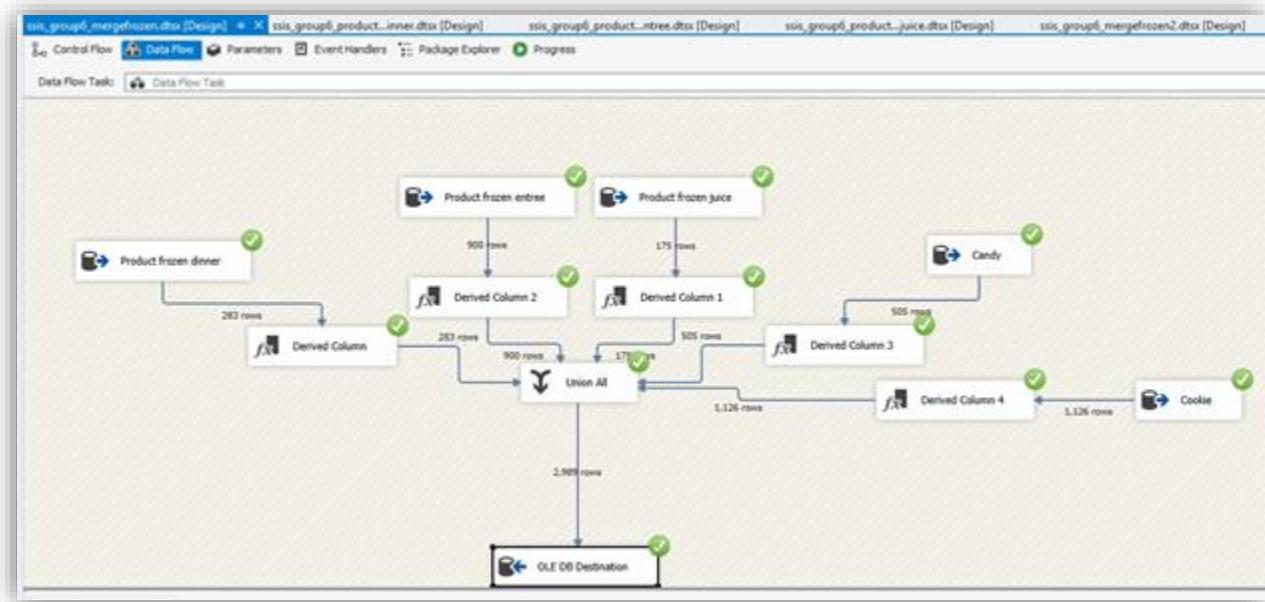
1. The store table after extraction to staging area had all the data types as varchar(255)/nvarchar(255). In order to create the dimension table from this, we will convert the value to appropriate data types.
2. The Store number will be converted to integer value to suit our needs
3. The other columns were kept as varchar and their sizes will be changed from 255 to needed number
4. Since we did not want to lose any data from this master table, we will add a value ‘None’ to all the column values which were NULL or empty strings. For example, we need the price tier column for one question, so we will change the value to ‘None’ wherever this value was NULL

Transformation 3: Week

1. For the week data taken from the Dominick’s data, after extraction into the staging area, we will need to a new week 0 was added into the week table as there is existing data in the sales and movement data

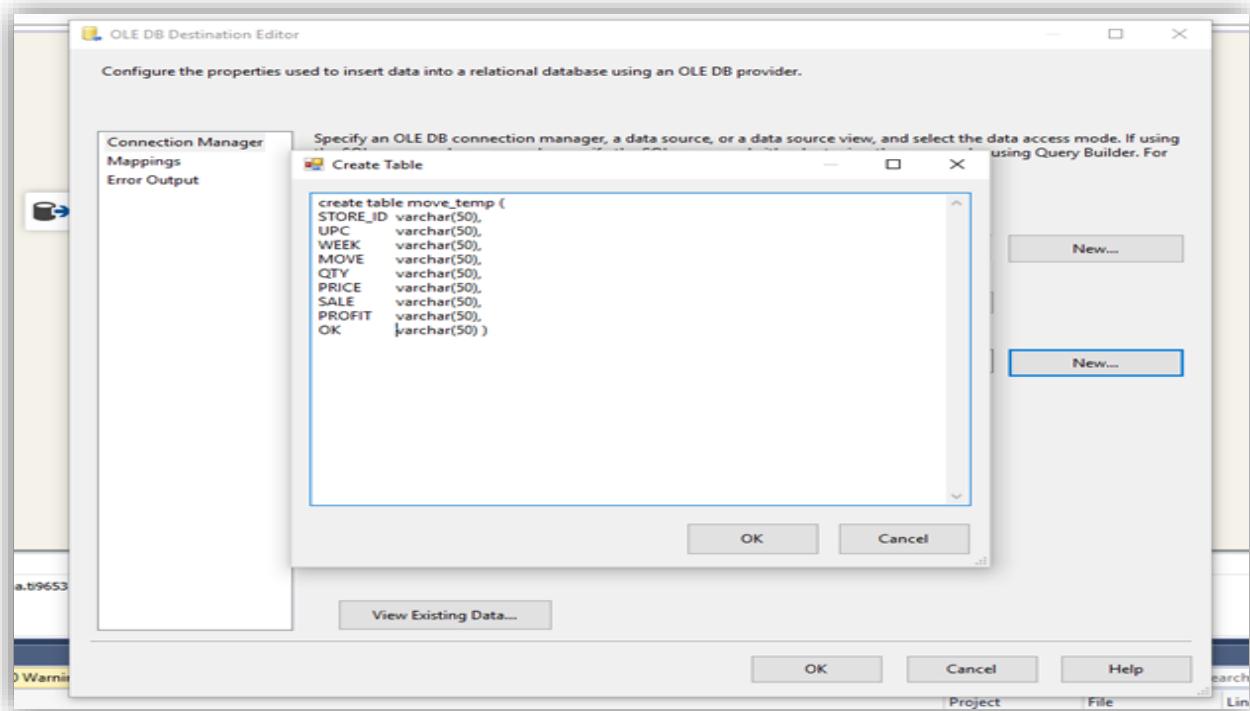
Transformation 4: Product

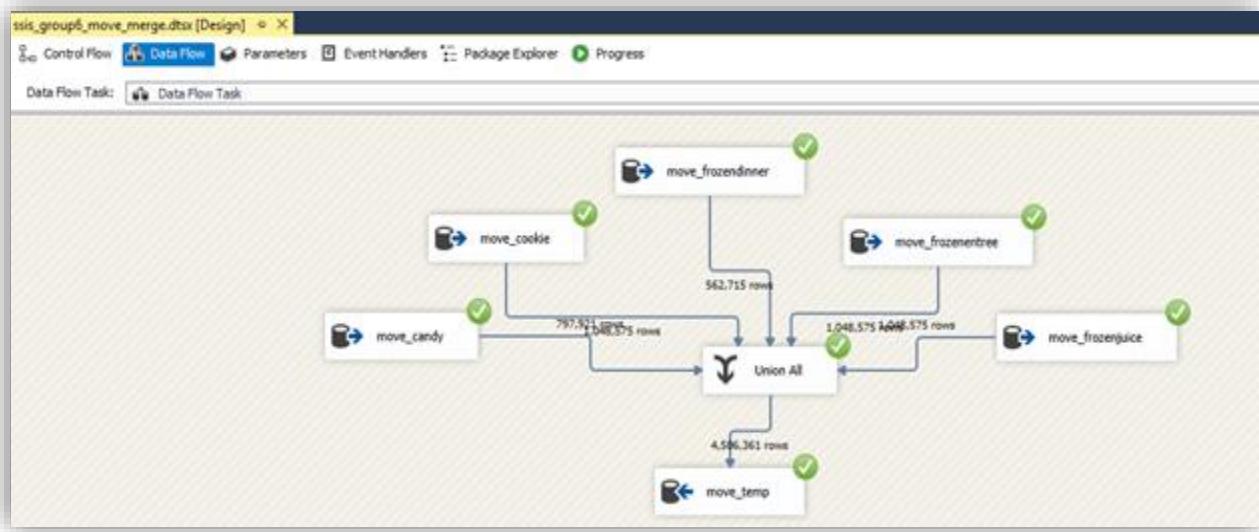
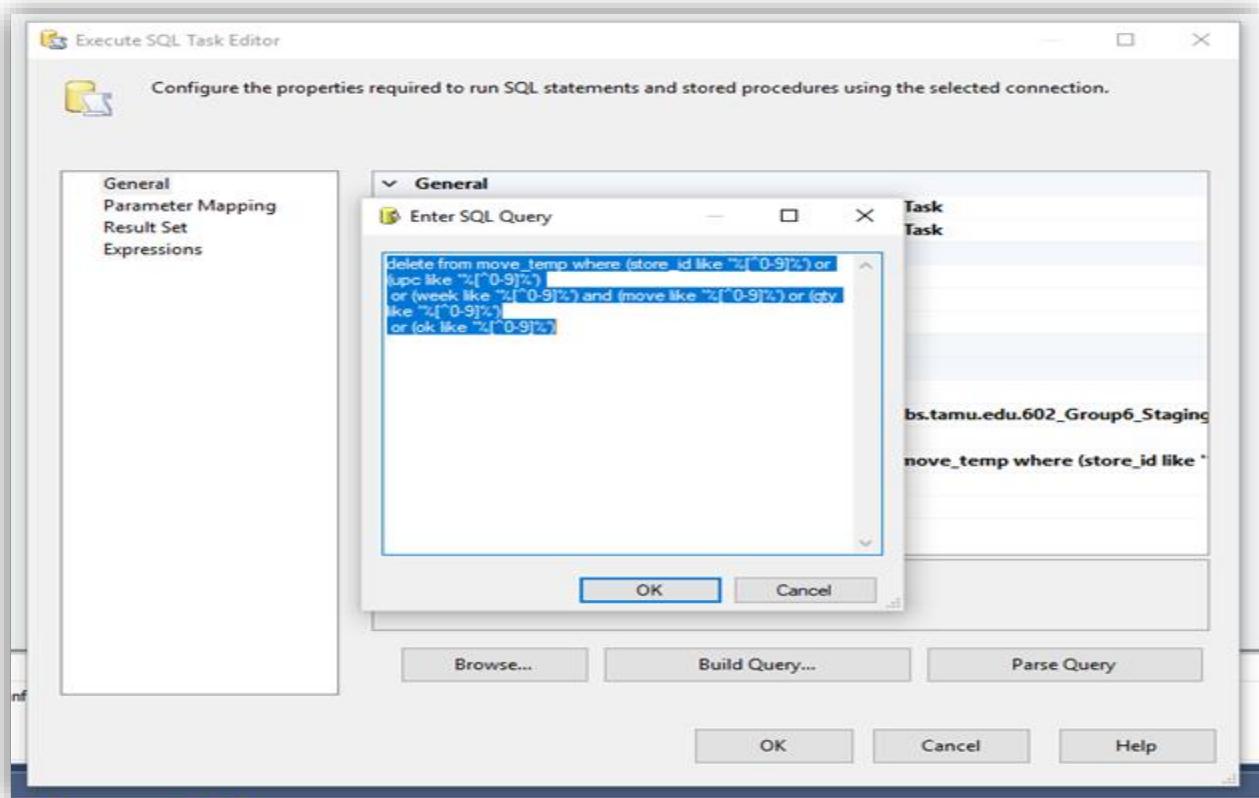
1. After loading the data for the products we need into the staging area, we will use SSIS to integrate all the tables and use derived column to add a new column
2. This new column is the ‘Category’ column we need to distinguish the products into a unique category
3. After this, we will store it into the Product table in the staging area

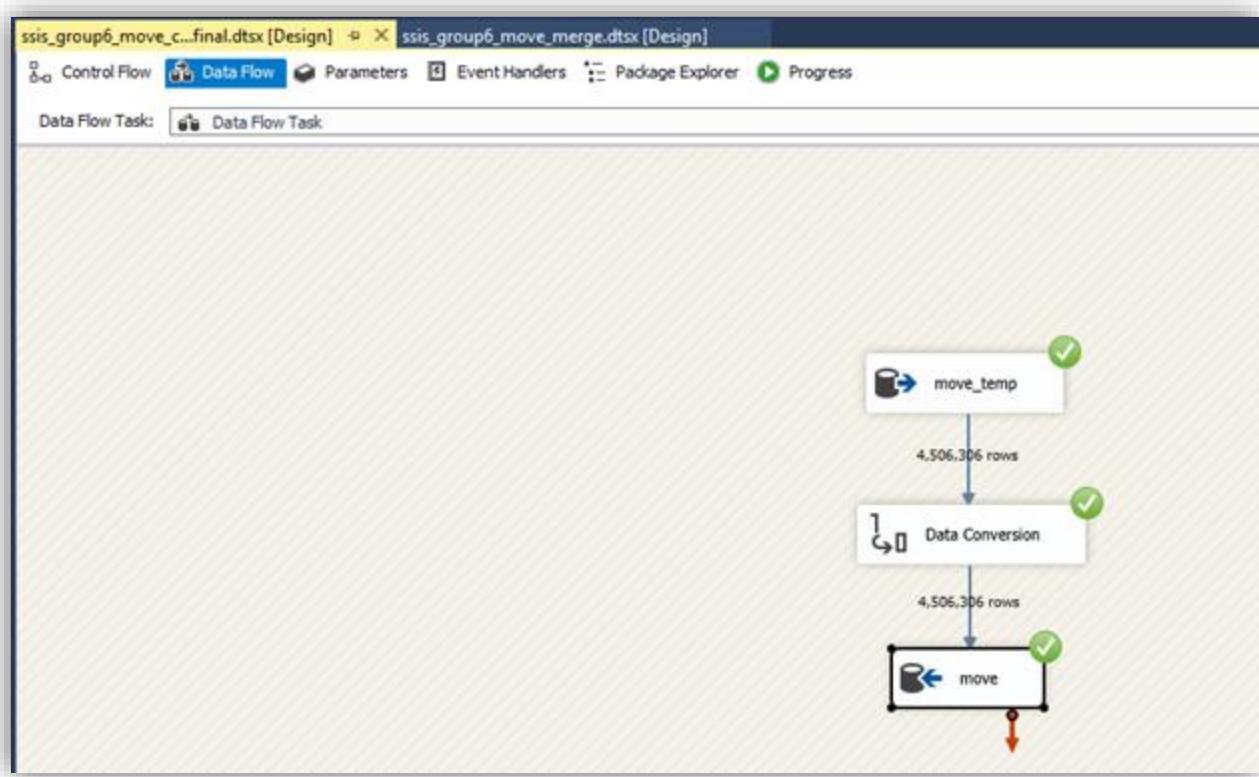


Transformation 5: Movement

1. After loading the data for the Movement we need into the staging area, we will use SSIS to integrate all the tables
2. After this, we will store it into a temporary Move table in the staging area







4.6 Aggregate tables

Our aggregation plans included aggregation in the fact tables as the lowest granularity is week. Data is given date wise for Movement, UPC and CCount. So, to bring it to the required granularity level, we will aggregate the data store wise and week wise in our fact tables.

4.7 Organization of data staging area

1. We will load the raw data into tables in the staging area as it is and create tables such as
 - dbo.CCOUNT
 - dbo.move_frozendinner
 - dbo.move_frozenentree
 - dbo.move_frozenjuice
 - dbo.move_cookie
 - dbo.move_candy
 - dbo.product_frozendinner
 - dbo.product_frozenentree
 - dbo.product_frozenjuice
 - dbo.product_cookie
 - dbo.product_candy
2. After this we will clean all the data, apply transformations, integrate the product tables into one product table, and integrate movement tables into one table to create the sales table
3. In between the processes, we will be using temporary tables to apply transformations and at the end we will drop these temporary tables to clean the staging area

Integrated tables:

Move

```
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [STORE_ID]
, [UPC]
, [WEEK]
, [MOVE]
, [QTY]
, [PRICE]
, [SALE]
, [PROFIT]
, [OK]
FROM [602_Group6_Staging_Area].[dbo].[move]
```

100 %

	STORE_ID	UPC	WEEK	MOVE	QTY	PRICE	SALE	PROFIT	OK
1	124	10700000339	202	66	1	0.42	B	61.19	1
2	124	10700000339	203	78	1	0.42	B	61.42	1
3	124	10700000339	204	86	1	0.42	B	60.71	1
4	124	10700000339	205	91	1	0.42	B	53.8	1
5	124	10700000339	206	62	1	0.42	B	48.57	1
6	124	10700000339	207	43	1	0.42	B	48.57	1
7	124	10700000339	208	61	1	0.45		52	1
8	124	10700000339	209	43	1	0.45		58.66	1
9	124	10700000339	210	66	1	0.45		60.88	1
10	124	10700000339	211	58	1	0.46		59.13	1
11	124	10700000339	212	54	1	0.49		51.02	1
12	124	10700000339	213	49	1	0.49		50.4	1
13	124	10700000339	214	27	1	0.49		50.4	1
14	124	10700000339	215	48	1	0.49		49.79	1
15	124	10700000339	216	37	1	0.49		49.79	1
16	124	10700000339	217	40	1	0.49		49.59	1
17	124	10700000339	218	66	1	0.47		46.38	1
18	124	10700000339	219	0	1	0		0	0
19	124	10700000339	220	76	1	0.39	B	33.84	1
20	124	10700000339	221	47	1	0.49		47.34	1
21	124	10700000339	222	64	1	0.49		47.14	1
22	124	10700000339	223	61	1	0.49		47.14	1
23	124	10700000339	224	59	1	0.49		46.53	1
24	124	10700000339	225	76	1	0.49		45.3	1
25	124	10700000339	226	86	1	0.49		45.3	1
26	124	10700000339	227	73	1	0.49		43.67	1
27	124	10700000339	228	73	1	0.49		43.67	1
28	124	10700000339	229	75	1	0.49		43.46	1
29	124	10700000339	230	73	1	0.49		42.65	1

Product

```
SELECT TOP (1000) [PRODUCT]
, [ITEM_CODE]
, [UPC]
, [SIZE]
, [CASE]
, [CATEGORY]
, [DESCRIPTION]
, [COMMODITY_CODE]
FROM [602_Group6_Staging_Area].[dbo].[PRODUCT]
```

100 %

	PRODUCT	ITEM_CODE	UPC	SIZE	CASE	CATEGORY	DESCRIPTION	COMMODITY_CODE
1	1	9309691	1380013201	13.20Z	12	Frozen Dinner	LC HP CHICKEN FLOREN	104
2	2	9309711	1380013202	14 OZ	12	Frozen Dinner	LC HP ROASTED TURKEY	104
3	3	9309771	1380013203	14.25O	12	Frozen Dinner	LC HP SRLN BF TIPS	104
4	4	9309791	1380013204	14 OZ	12	Frozen Dinner	LC HP GRLD CHKN W/PE	104
5	5	9309801	1380013205	15.30Z	12	Frozen Dinner	LC HP JUMBO RIGATONI	104
6	6	9309861	1380013206	14 OZ	12	Frozen Dinner	LC HP ORIENTAL GLAZE	104
7	7	9309651	1380013207	14 OZ	12	Frozen Dinner	LC HP BEEF LO MEIN	104
8	8	9309671	1380013208	12.5 O	12	Frozen Dinner	LC HP ROASTED CHICK	104
9	9	9310121	1380013209	15 OZ	12	Frozen Dinner	LC HEARTY PORTIONS L	104
10	10	9310141	1380013210	15.5 O	12	Frozen Dinner	LC HRTY PORTIONS CHS	104
11	11	9310041	1380013304	17 OZ	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
12	12	9309931	1380013305	16 OZ	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
13	13	9310051	1380013306	15.10Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
14	14	9309951	1380013307	16.75Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
15	15	9309991	1380013308	16 OZ	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
16	16	9310001	1380013309	17.50Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
17	17	9310131	1380013310	16 OZ	12	Frozen Dinner	STOUFFERS HP CNTRY F	104
18	18	9310011	1380013311	13.5 O	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
19	19	9309881	1380013312	15.4 O	12	Frozen Dinner	STOUFFERS HEARTY POR	104
20	20	9043031	2113150434	17 OZ	8	Frozen Dinner	MARIE CALLENDER SPAG	104
21	21	9043081	2113150440	16 OZ	8	Frozen Dinner	MARIE CALLENDER RAVI	104
22	22	9043041	2113150475	13 OZ	8	Frozen Dinner	FETTUCINI W/BROCCOLI	104
23	23	9043001	2113150560	14 OZ	8	Frozen Dinner	MARIE CALLENDER MEAT	104
24	24	9043051	2113150575	13 OZ	8	Frozen Dinner	ESCALLOPED NOODLES &	104
25	25	9043061	2113150595	15 OZ	8	Frozen Dinner	MARIE CALLENDER LASA	104

Sales_Final

```

SELECT TOP (1000) [SALE_ID]
      ,[STORE_ID]
      ,[DATE]
      ,[WEEK]
      ,[CATEGORY]
      ,[CUSTOMER_COUNT]
      ,[SALES_AMOUNT]
  FROM [602_Group6_Staging_Area].[dbo].[SALES_FINAL]

```

00 %

	SALE_ID	STORE_ID	DATE	WEEK	CATEGORY	CUSTOMER_COUNT	SALES_AMOUNT
1	1	68	1992-07-12	148	BAKCOUP	2670	0
2	3	68	1992-07-12	148	FROZCOUP	2670	0
3	5	68	1992-07-13	148	BAKCOUP	2336	0
4	7	68	1992-07-13	148	FROZCOUP	2336	0
5	9	68	1992-07-14	148	BAKCOUP	2378	0
6	11	68	1992-07-14	148	FROZCOUP	2378	0
7	13	68	1992-07-15	148	BAKCOUP	2539	0
8	15	68	1992-07-15	148	FROZCOUP	2539	0
9	17	68	1992-07-16	149	BAKCOUP	2487	0
10	19	68	1992-07-16	149	FROZCOUP	2487	0
11	21	68	1992-07-17	149	BAKCOUP	2554	0
12	23	68	1992-07-17	149	FROZCOUP	2554	0
13	25	68	1992-07-18	149	BAKCOUP	2722	0
14	27	68	1992-07-18	149	FROZCOUP	2722	0
15	29	68	1992-07-19	149	BAKCOUP	2713	0
16	31	68	1992-07-19	149	FROZCOUP	2713	0
17	33	68	1992-07-20	149	BAKCOUP	2379	0
18	35	68	1992-07-20	149	FROZCOUP	2379	0
19	37	68	1992-07-21	149	BAKCOUP	2466	0
20	39	68	1992-07-21	149	FROZCOUP	2466	0
21	41	68	1992-07-22	149	BAKCOUP	2274	0
22	43	68	1992-07-22	149	FROZCOUP	2274	0
23	45	68	1992-07-23	150	BAKCOUP	2802	0
24	47	68	1992-07-23	150	FROZCOUP	2802	0

DIM_STORE

Object Explorer

```

SELECT [STORE_KEY]
      ,[STORE_ID]
      ,[PRICE_TIER]
      ,[ADDRESS]
      ,[CITY]
      ,[ZONE]
      ,[ZIP_CODE]
  FROM [602_Group6_Warehouse].[dbo].[DIM_STORE]

```

100 %

	STORE_KEY	STORE_ID	PRICE_TIER	ADDRESS	CITY	ZONE	ZIP_CODE
1	1	2	High	7501 W. North Ave.	River Forest	1	60305
2	2	4	Medium	Closed	Park Ridge	2	60068
3	3	5	Medium	223 Northwest HWY.	Palatine	2	60067
4	4	8	Low	8700 S. Cicero Ave.	Oak Lawn	5	60435
5	5	9	Medium	6931 Dempster	Morton Grove	2	60053
6	6	12	High	6009 N. Broadway Ave.	Chicago	7	60660
7	7	14	High	1020 Waukegan Rd.	Glenview	1	60025
8	8	18	Low	8355 W. Belmont Ave.	River Grove	5	60171
9	9	19	None	Closed	Glen Ellyn	None	60137
10	10	21	CubFighter	1440 Irving Park Rd.	Hanover Park	6	60103
11	11	25	None	Closed	Chicago	None	60639
12	12	28	Medium	1145-55 Mt Prospect Pz.	Mt. Prospect	2	60054
13	13	32	High	1900 N. Cumberland Ave.	Park Ridge	1	60068
14	14	33	High	3012 N. Broadway Ave.	Chicago	7	60657
15	15	39	None	Closed	Waukegan	None	60085
16	16	40	CubFighter	8925 S. Harlem	Bridgeview	6	60455
17	17	44	Medium	14 Garden Market	Western Spring	2	60558
18	18	45	Medium	550 W. Dundee Rd.	Wheeling	2	60090
19	19	46	Low	Closed	Carol Stream	5	60187
20	20	47	Medium	545 W. Lake St.	Addison	2	60101
21	21	48	Medium	20 E. Ogden Ave.	Schaumburg	2	60193
22	22	49	Medium	120 E. Ogden Ave.	Downers Grove	2	60515
23	23	50	Medium	8631 W. 95th St.	Hickory Hills	2	60457
24	24	51	Medium	6401 W. 127th St.	Palos Heights	3	60463
25	25	52	High	4125 Dundee Rd.	Northbrook	1	60062
26	26	53	High	3145 W. Pratt Ave.	Chicago	7	60662
27	27	54	Medium	1295 E. Ogden Ave.	Naperville	2	60540

DIM_WEEK

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects, including tables like COUPON_REDEMPTION_FACT, DIM_PRODUCT, DIM_STORE, and DIM_WEEK. The right pane displays a query window with the following SQL code:

```
SELECT [week_key]
      ,[week_number]
      ,[start]
      ,[end]
      ,[special_events]
  FROM [602_Group6_Warehouse].[dbo].[DIM_WEEK]
```

The results grid shows data for weeks from week 1 to week 27, with specific events marked for certain weeks:

week_key	week_number	start	end	special_events
1	1	1989-09-14	1989-09-20	
2	2	1989-09-21	1989-09-27	
3	3	1989-09-28	1989-10-04	
4	4	1989-10-05	1989-10-11	
5	5	1989-10-12	1989-10-18	
6	6	1989-10-19	1989-10-25	
7	7	1989-10-26	1989-11-01	Halloween
8	8	1989-11-02	1989-11-08	
9	9	1989-11-09	1989-11-15	
10	10	1989-11-16	1989-11-22	
11	11	1989-11-23	1989-11-29	Thanksgiving
12	12	1989-11-30	1989-12-06	
13	13	1989-12-07	1989-12-13	
14	14	1989-12-14	1989-12-20	
15	15	1989-12-21	1989-12-27	Christmas
16	16	1989-12-28	1990-01-03	New-Year
17	17	1990-01-04	1990-01-10	
18	18	1990-01-11	1990-01-17	
19	19	1990-01-18	1990-01-24	
20	20	1990-01-25	1990-01-31	
21	21	1990-02-01	1990-02-07	
22	22	1990-02-08	1990-02-14	
23	23	1990-02-15	1990-02-21	Presidents Day
24	24	1990-02-22	1990-02-28	
25	25	1990-03-01	1990-03-07	
26	26	1990-03-08	1990-03-14	
27	27	1990-03-15	1990-03-21	

DIM_PRODUCT

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects, including tables like COUPON_REDEMPTION_FACT, DIM_PRODUCT, and DIM_STORE. The right pane displays a query window with the following SQL code:

```
SELECT [PRODUCT_KEY]
      ,[PRODUCT]
      ,[ITEM_CODE]
      ,[UPC]
      ,[SIZE]
      ,[CASE]
      ,[CATEGORY]
      ,[DESCRIPTION]
      ,[COMMODITY_CODE]
  FROM [602_Group6_Warehouse].[dbo].[DIM_PRODUCT]
```

The results grid shows data for various products, categorized by category and commodity code:

PRODUCT_KEY	PRODUCT	ITEM_CODE	UPC	SIZE	CASE	CATEGORY	DESCRIPTION	COMMODITY_CODE
1	1	9309691	1380013201	13.20Z	12	Frozen Dinner	LC HP CHICKEN FLOREN	104
2	2	9309711	1380013202	14.0Z	12	Frozen Dinner	LC HP ROASTED TURKEY	104
3	3	9309771	1380013203	14.25O	12	Frozen Dinner	LC HP SRLN BF TIPS	104
4	4	9309791	1380013204	14.0Z	12	Frozen Dinner	LC HP GRLD CHKN W/PE	104
5	5	9309801	1380013205	15.30Z	12	Frozen Dinner	LC HP JUMBO RIGATONI	104
6	6	9309861	1380013206	14.0Z	12	Frozen Dinner	LC HP ORIENTAL GLAZE	104
7	7	9309651	1380013207	14.0Z	12	Frozen Dinner	LC HP BEEF LO MEIN	104
8	8	9309671	1380013208	12.5O	12	Frozen Dinner	LC HP ROASTED CHICKE	104
9	9	9310121	1380013209	15.0Z	12	Frozen Dinner	LC HEARTY PORTIONS L	104
10	10	9310141	1380013210	15.5O	12	Frozen Dinner	LC HRTY PORTIONS CHS	104
11	11	9310041	1380013304	17.0Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
12	12	9309931	1380013305	16.0Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
13	13	9310051	1380013306	15.10Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
14	14	9309951	1380013307	16.75Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
15	15	9309991	1380013308	16.0Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
16	16	9310001	1380013309	17.50Z	12	Frozen Dinner	STOUFFER'S HRTY PRTN	104
17	17	9310131	1380013310	16.0Z	12	Frozen Dinner	STOUFFERS HP CNTRY F	104
18	18	9310011	1380013311	13.5 O	12	Frozen Dinner	STOUFFERS HRTY PRTN	104
19	19	9309881	1380013312	15.4 O	12	Frozen Dinner	STOUFFERS HEARTY POR	104
20	20	9043031	2113150434	17.0Z	8	Frozen Dinner	MARIE CALLENDAR SPAG	104
21	21	9043081	2113150440	16.0Z	8	Frozen Dinner	MARIE CALLENDAR RAVI	104
22	22	9043041	2113150475	13.0Z	8	Frozen Dinner	FETTUCINI W/BROCCOLI	104
23	23	9043001	2113150560	14.0Z	8	Frozen Dinner	MARIE CALLENDAR MEAT	104
24	24	9043051	2113150575	13.0Z	8	Frozen Dinner	ESCALOPED NOODLES &	104
25	25	9043061	2113150595	15.0Z	8	Frozen Dinner	MARIE CALLENDAR LASA	104
26	26	9043011	2113150605	16.0Z	8	Frozen Dinner	MARIE CALLENDAR COUN	104
27	27	9043091	2113150630	14.0Z	8	Frozen Dinner	MARIE CALLENDAR ROAS	104

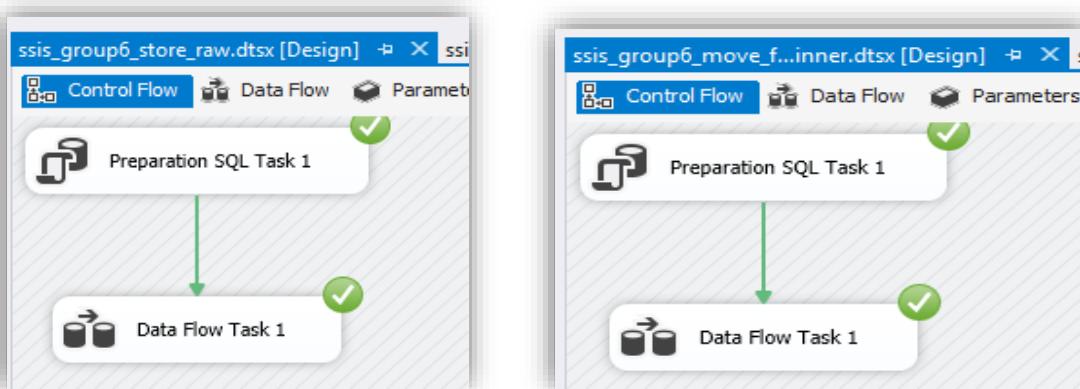
COUPON_REDEMPTION_FACT

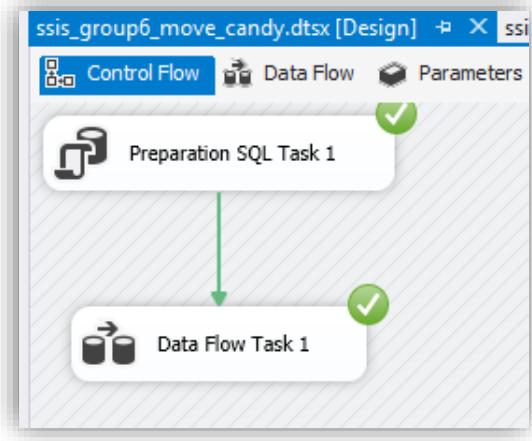
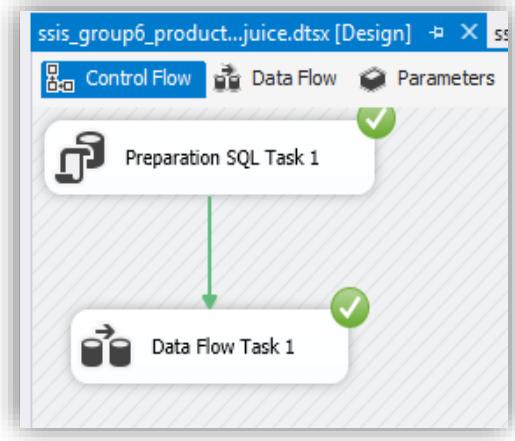
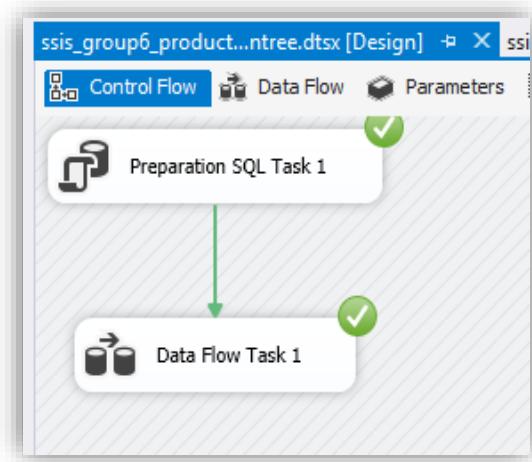
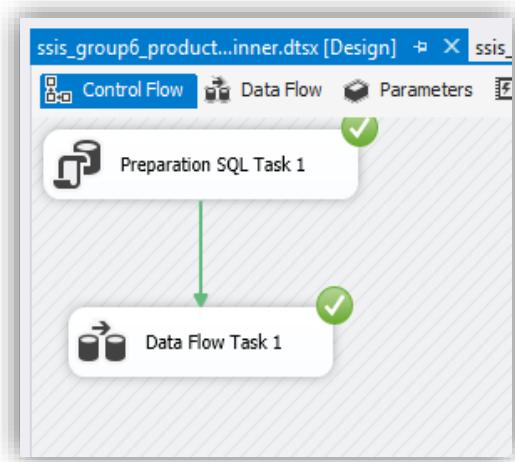
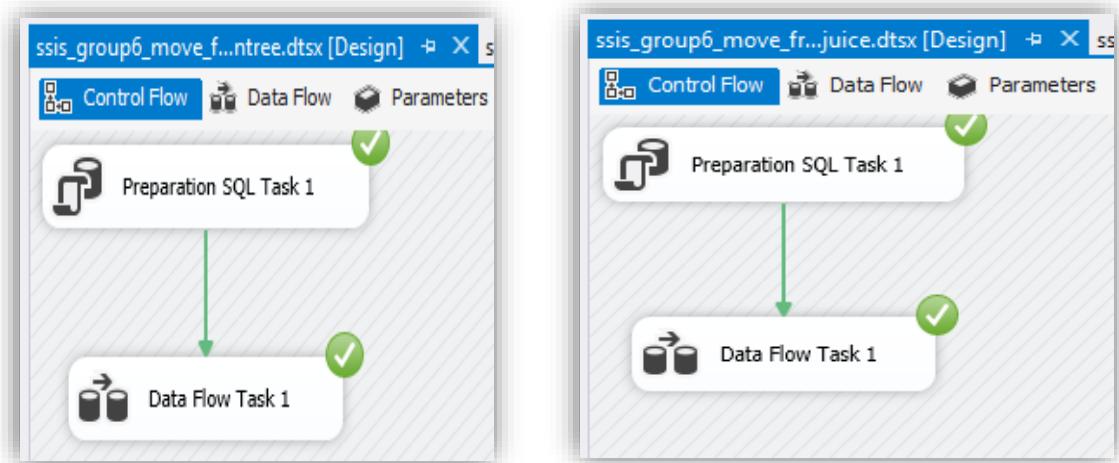
The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer pane displays a tree view of database objects, including several groups of databases (602_Group2_StagingDB, 602_Group6_Staging_Area, 602_Group6_Warehouse) and their respective tables, views, and other metadata. On the right, a query window titled 'SQLQuery35.sql' is open, displaying a SELECT statement and its results. The results grid shows data for the 'dbo.COUPON_REDEMPTION_FACT' table, with columns: COUPON_REDEMPTION_KEY, STORE_KEY, WEEK_KEY, PRODUCT_KEY, and SALES_AMOUNT. The data consists of 30 rows of integer values.

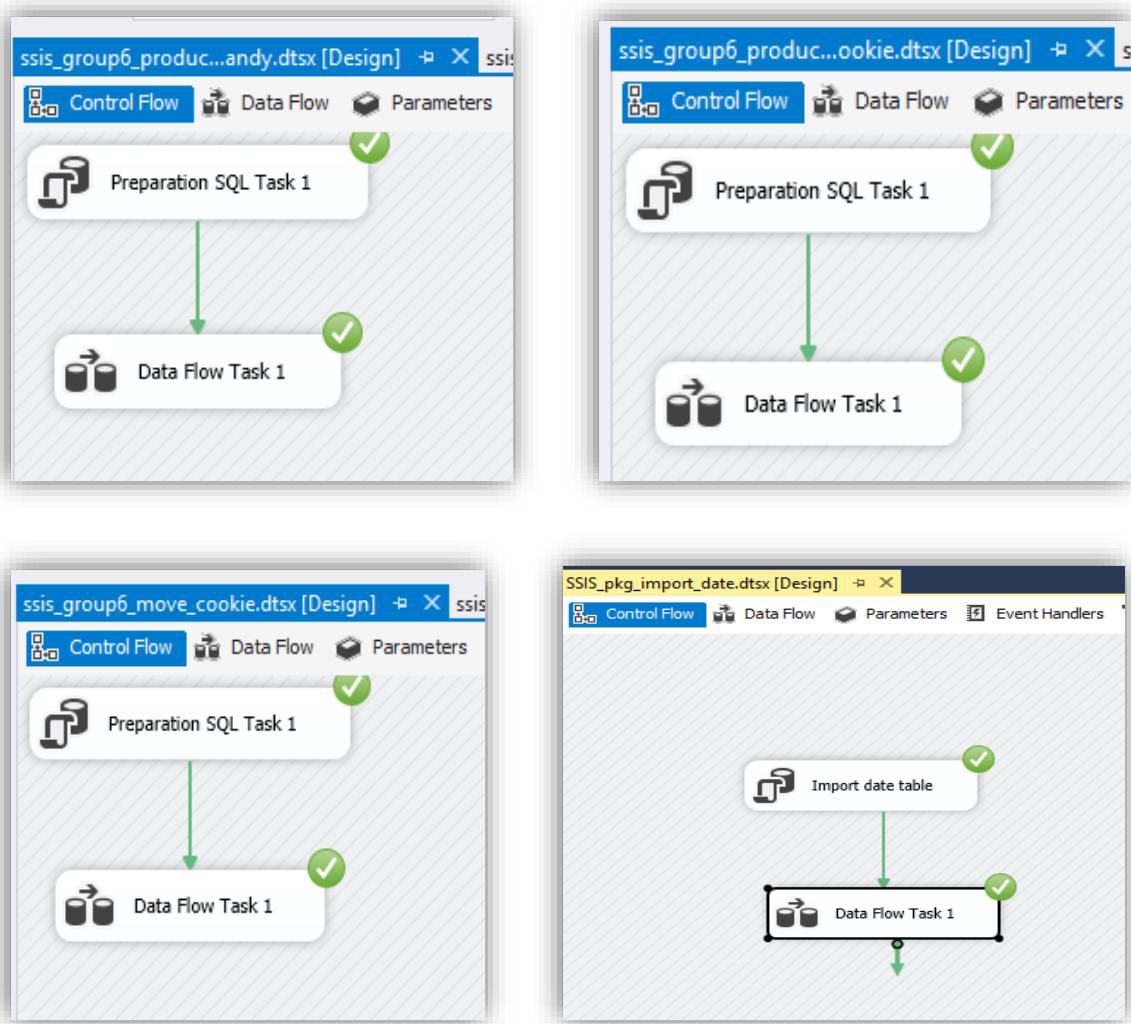
	COUPON_REDEMPTION_KEY	STORE_KEY	WEEK_KEY	PRODUCT_KEY	SALES_AMOUNT
1	1	13	281	3003	0
2	2	14	217	3003	0
3	3	26	317	3002	0
4	4	27	253	3002	0
5	5	29	125	3002	0
6	6	25	381	3002	0
7	7	94	339	3003	0
8	8	95	275	3003	0
9	9	4	108	3003	0
10	10	3	300	3003	0
11	11	5	44	3003	0
12	12	12	208	3002	0
13	13	73	230	3003	0
14	14	71	358	3003	0
15	15	76	38	3003	0
16	16	74	166	3003	0
17	17	72	294	3003	0
18	18	75	102	3003	0
19	19	93	138	3002	0
20	20	91	266	3002	0
21	21	89	394	3002	0
22	22	92	202	3002	0
23	23	90	330	3002	0
24	24	58	274	3005	2911
25	25	2	35	3002	0
26	26	1	163	3002	0
27	27	51	121	3003	0
28	28	50	249	3003	0
29	29	49	313	3003	0
30	30	70	62	3002	0

4.8 Procedures for all data extractions and loadings

The sources for this project are the csv files given which are UPC, Movement, CCount, Demographics. Additionally, we have data for Store and Date dimension. The data extractions and load are shown in the SSIS package execution below.



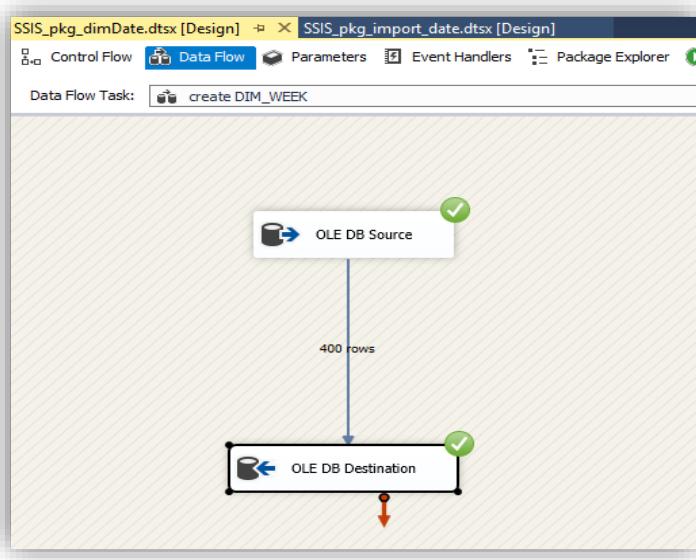




4.9 ETL for dimension tables

4.9.1 DIM_WEEK

The week dimension is simply created from the WeekTable in the staging area.



```

CREATE TABLE [DIM_WEEK]
(
    [WeekNumber] int,
    [Start] date,
    [End] date,
    [Special Events] varchar(50)
)

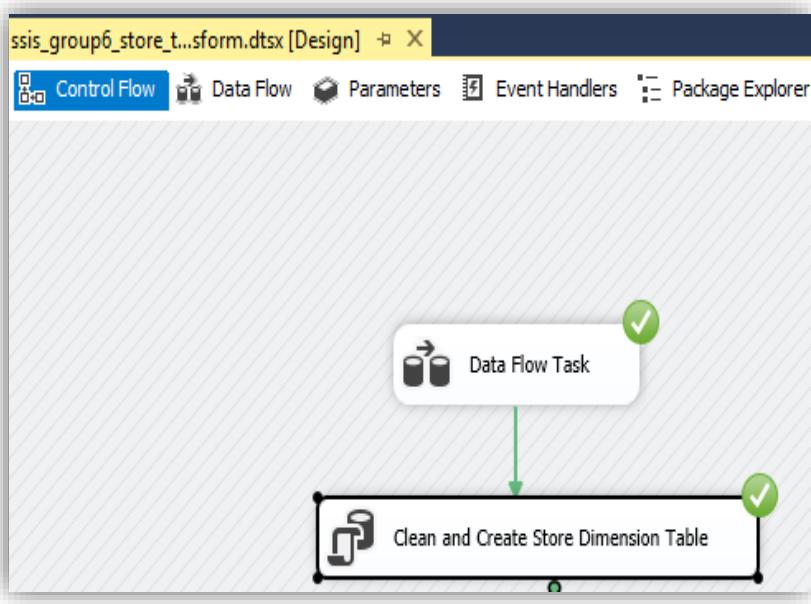
INSERT INTO DIM_STORE(STORE_ID ,PRICE_TIER ,ADDRESS ,CITY ,ZONE ,ZIP_CODE )
SELECT [Store],[Price Tier], [Address],[City], [Zone],
[Zip Code]
FROM STORE

INSERT INTO DIM_WEEK
SELECT
WEEK_KEY INT NOT NULL PRIMARY KEY IDENTITY (1,1),
[WeekNumber],
    [Start],
    [End],
    [Special Events]
FROM WEEKTABLE

```

4.9.2 DIM_STORE

The store dimension is created from the table loaded ‘Store’ into the staging area. Transformations were applied to it before creating the dimension as mentioned in the transformation and cleaning phase above.



```

UPDATE [STORE]
SET [Price Tier]='None'
WHERE [Price Tier] IS NULL OR [Price Tier] = '';

UPDATE [STORE]
SET [Zone]='None'
WHERE [Zone] IS NULL OR [Zone] = '';

CREATE TABLE DIM_STORE
(
STORE_KEY INT NOT NULL PRIMARY KEY IDENTITY(1,1),
STORE_ID INT NOT NULL,
PRICE_TIER VARCHAR(10) NOT NULL,
ADDRESS VARCHAR(100) ,
CITY VARCHAR(50) NOT NULL,
ZONE VARCHAR(10) ,
ZIP_CODE VARCHAR(10)

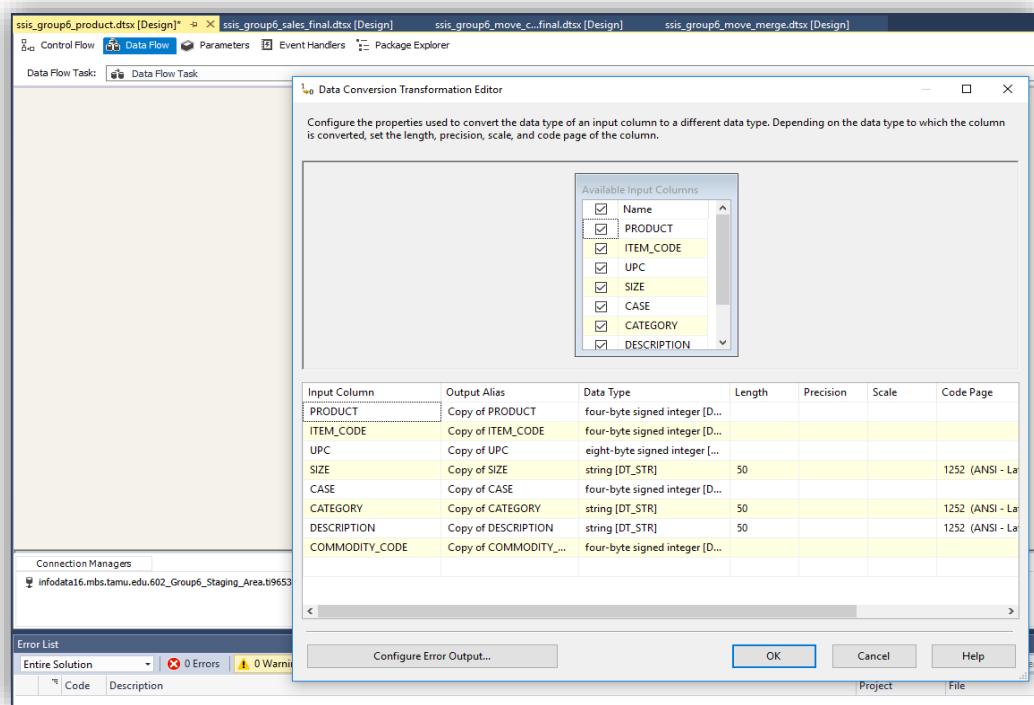
);

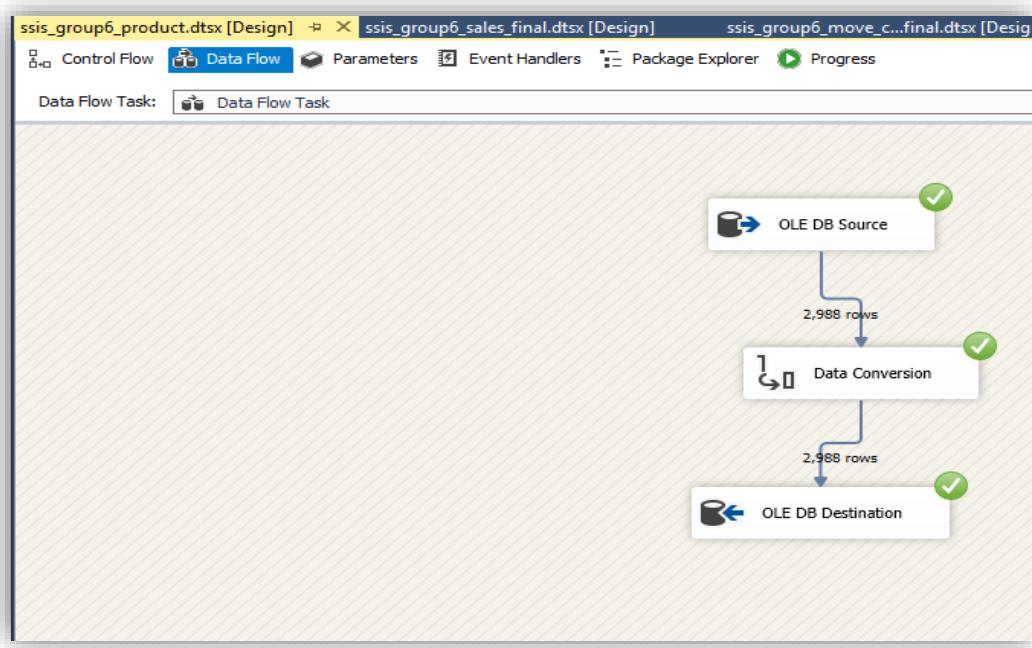
INSERT INTO DIM_STORE(STORE_ID ,PRICE_TIER ,ADDRESS ,CITY ,ZONE ,ZIP_CODE )
SELECT [Store],[Price Tier], [Address],[City], [Zone],
[Zip Code]
FROM STORE

```

4.9.3 DIM_PRODUCT

The product dimension is created from the integrated Product table in the staging area. The data types and length of the columns were changed in the SSIS as shown below.





```

CREATE TABLE [DIM_PRODUCT]
(
    [PRODUCT_KEY] int IDENTITY(1,1) PRIMARY KEY,
    [PRODUCT] int,
    [ITEM_CODE] bigint,
    [UPC] bigint,
    [SIZE] varchar(50),
    [CASE] int,
    [CATEGORY] varchar(50),
    [DESCRIPTION] varchar(50),
    [COMMODITY_CODE] int
)

```

4.10 ETL for fact tables

4.10.1 CUSTOMER_COUNT_FACT TABLE

Step 1: Cleaning the required columns by only taking numerical values in table SALES_CCF_TEMP1

```

CREATE TABLE SALES_CCF_TEMP1 (
    [STORE_ID] VARCHAR(50),
    [WEEK] VARCHAR(50),
    [CUSTOMER_COUNT] VARCHAR(50),
)

```

```

INSERT INTO SALES_CCF_TEMP1
(store_id, [week], customer_count)
SELECT store, [week], CUSTCOUN

```

```
FROM [DBO].[CCOUNT] WHERE [WEEK] NOT LIKE '%[^0-9]%'  
AND [CUSTCOUN] NOT LIKE '%[^0-9]%'  
AND STORE NOT LIKE '%[^0-9]%'
```

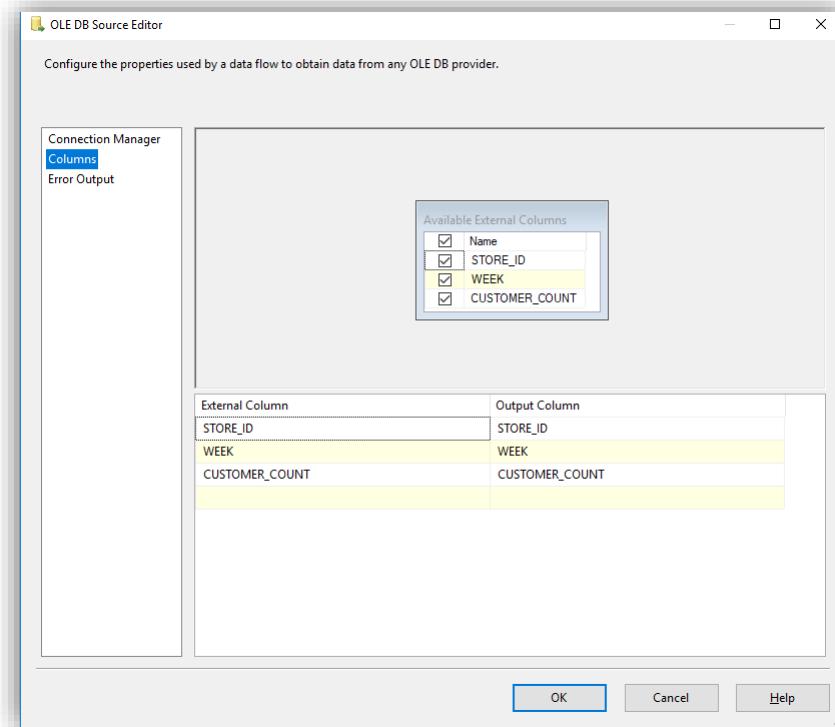
Step 2: Further cleaned table by converting values in integer and removing wrong values in the table SALES_CCF_TEMP2

```
CREATE TABLE SALES_CCF_TEMP2 (  
    [STORE_ID] INT,  
    [WEEK] INT,  
    [CUSTOMER_COUNT] INT,  
)  
  
INSERT INTO SALES_CCF_TEMP2  
( store_id, [week], customer_count)  
SELECT CAST(store_id AS INT), CAST([week] AS INT), CAST(customer_count AS FLOAT)  
FROM [DBO].[SALES_CCF_TEMP1] WHERE cast(STORE_ID as INT) <140  
AND STORE_ID IS NOT NULL  
AND WEEK IS NOT NULL  
AND CUSTOMER_COUNT IS NOT NULL  
AND STORE_ID NOT IN ('135','1','0');
```

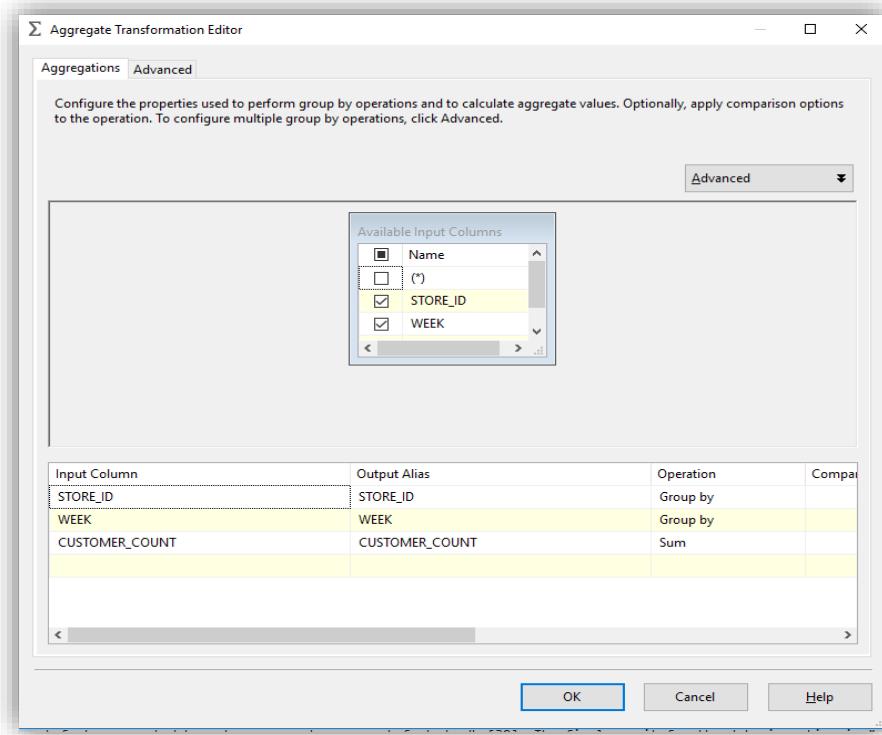
Step 3: Data before running SSIS commands

```
***** Script for SelectTopNRows command from SSMS *****
SELECT TOP (1000) [CUSTOMER_COUNT_KEY]
, [week_key]
, [STORE_KEY]
, [CUSTOMER_COUNT]
FROM [602_Group6_Warehouse].[dbo].[CUSTOMER_COUNT_FACT]
```

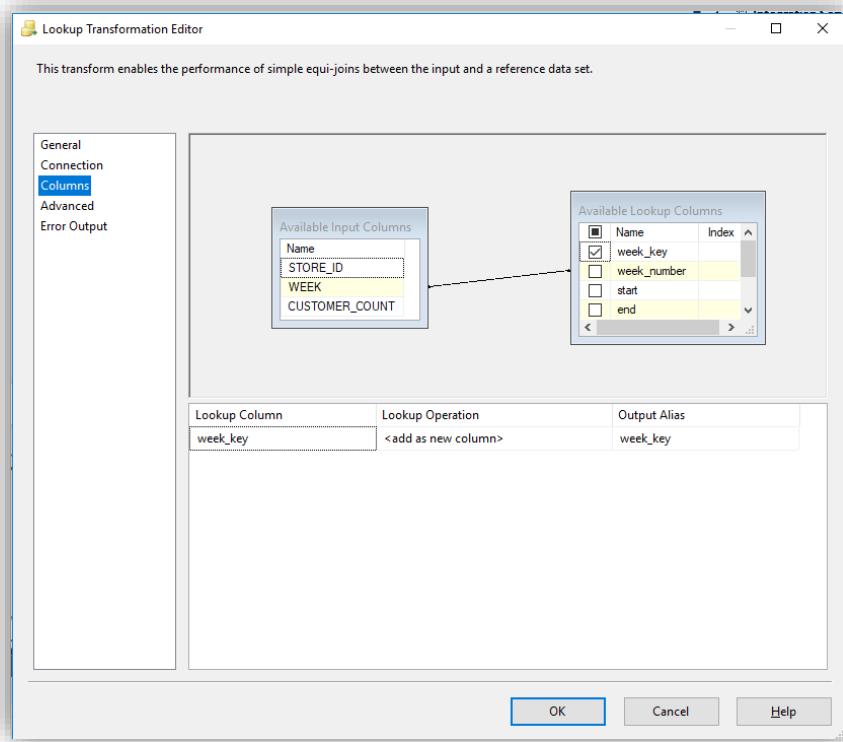
Step 4: Fetching store_id, week and customer count from SALES_CCF_TEMP2 TABLE



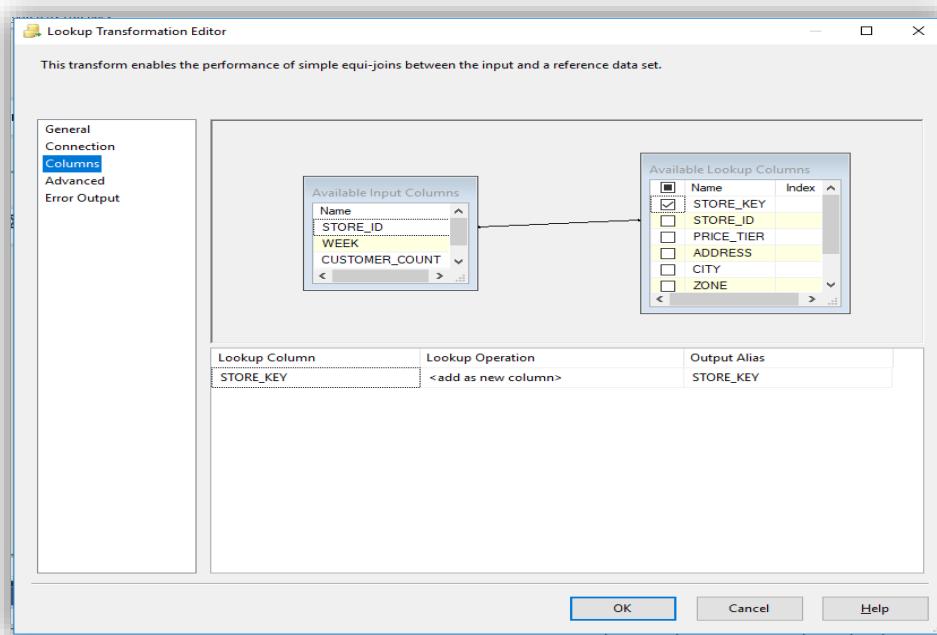
Step 5: Aggregating data on store_id and week and by summing the CUSTOMER_COUNT



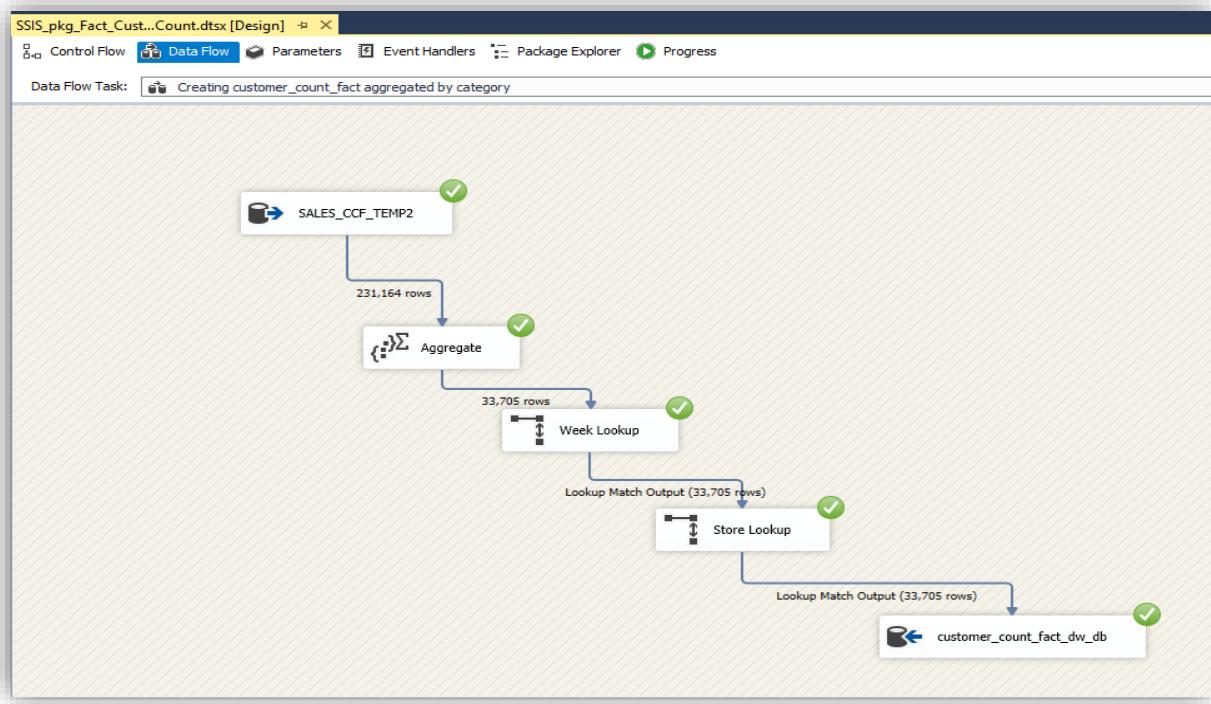
Step 6: Next, performing look up by DIM_WEEK (on week & WEEK_NUMBER) and selecting week_key



Step 7: Performing look up by DIM_STORE (on store_id and store_id) and selection STORE_KEY



Step 8: Final execution of “creating_customer_count_fact”



Step 9: Result after running above SSIS

SQLQuery26.sql - in...house (ti9653 (79)) → X SQLQuery21.sql - in...g_Area (ti9653 (67))*

```

***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [CUSTOMER_COUNT_KEY]
, [week_key]
, [STORE_KEY]
, [CUSTOMER_COUNT]
FROM [602_Group6_Warehouse].[dbo].[CUSTOMER_COUNT_FACT]

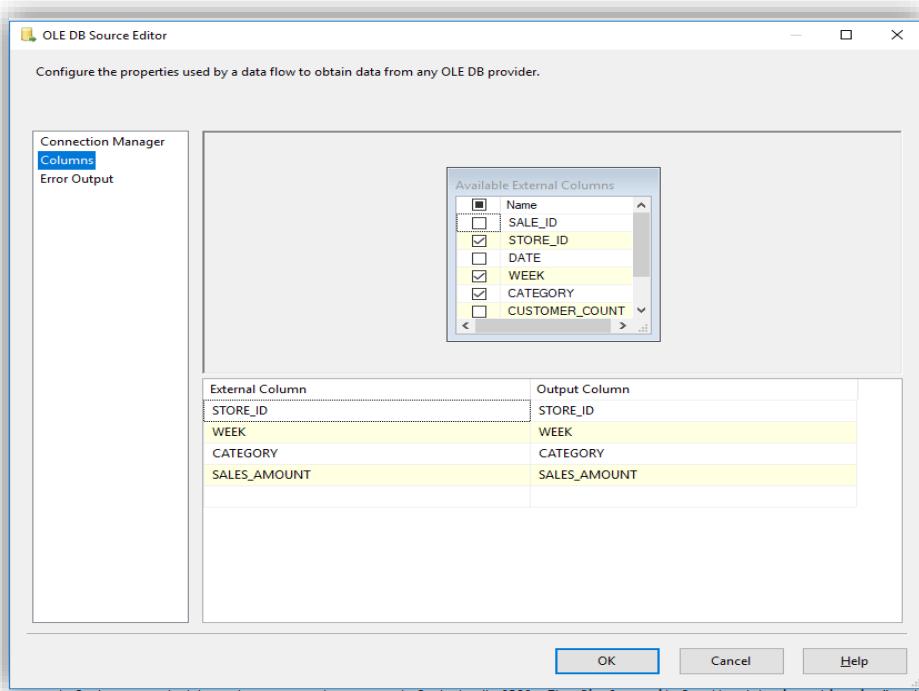
```

100 %

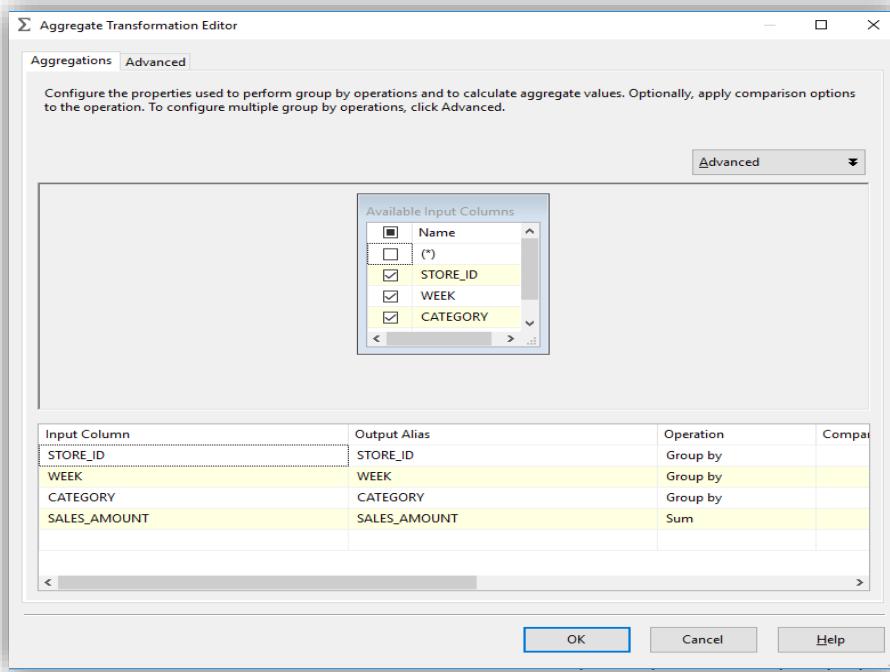
	CUSTOMER_COUNT_KEY	week_key	STORE_KEY	CUSTOMER_COUNT
1	1	401	1	13728
2	2	1	1	13870
3	3	2	1	15539
4	4	3	1	13987
5	5	4	1	14810
6	6	5	1	14415
7	7	6	1	13812
8	8	7	1	14574
9	9	8	1	14649
10	10	9	1	13551
11	11	10	1	17016
12	12	11	1	12731
13	13	12	1	13706
14	14	13	1	13890
15	15	14	1	15250
16	16	15	1	13977
17	17	16	1	13778

4.10.2 COUPONS_REDEMPTION_FACT

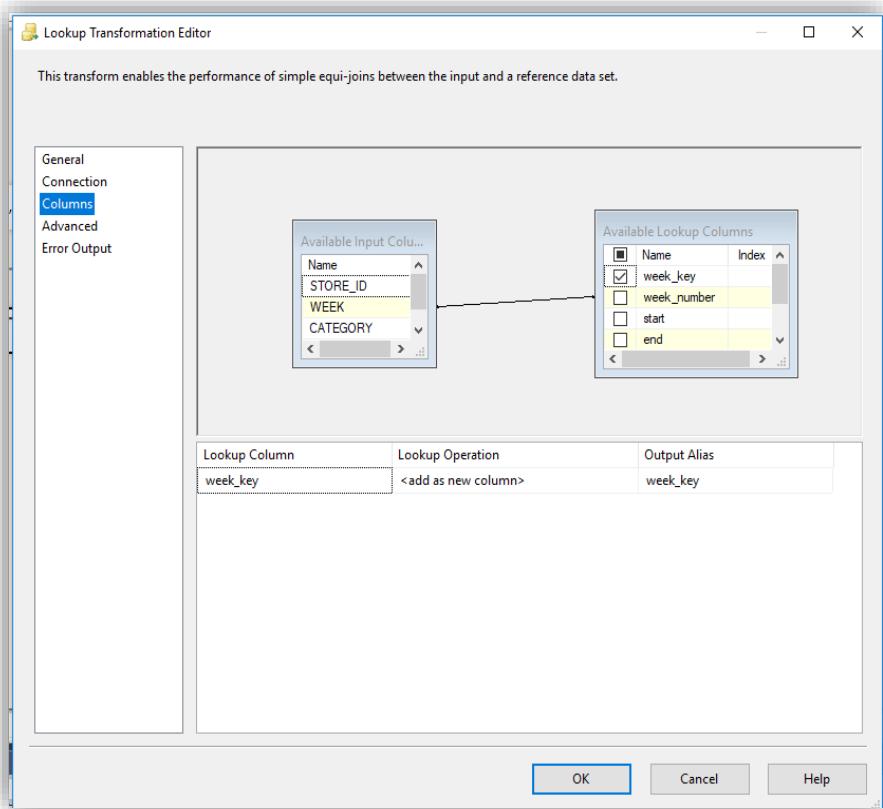
Step 1: Take source as SALES_FINAL (defined earlier how SALES_FINAL was formed) and selecting STORE_ID, WEEK, CATEGORY, SALES_AMOUNT



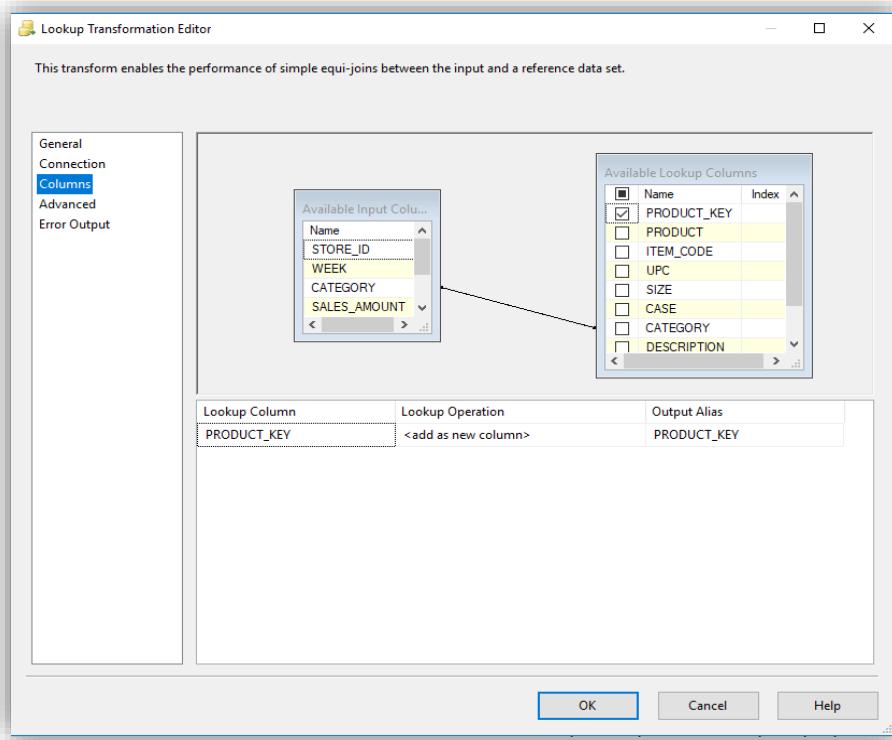
Step 2: Grouping by columns by store and week and category and by summing SALES_AMOUNT



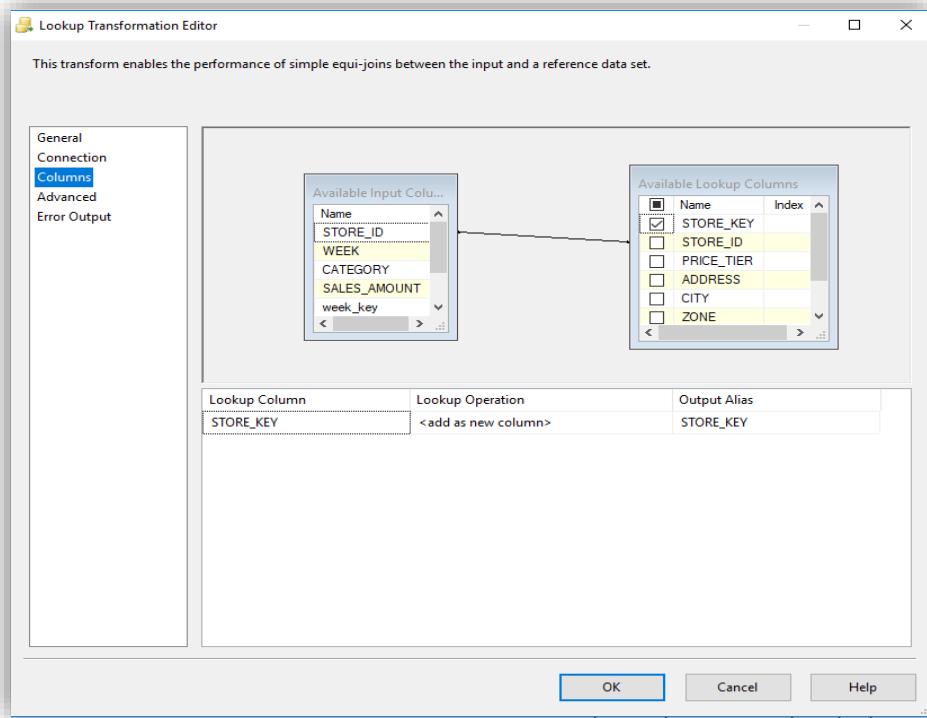
Step 3: Looking up by DIM_WEEK



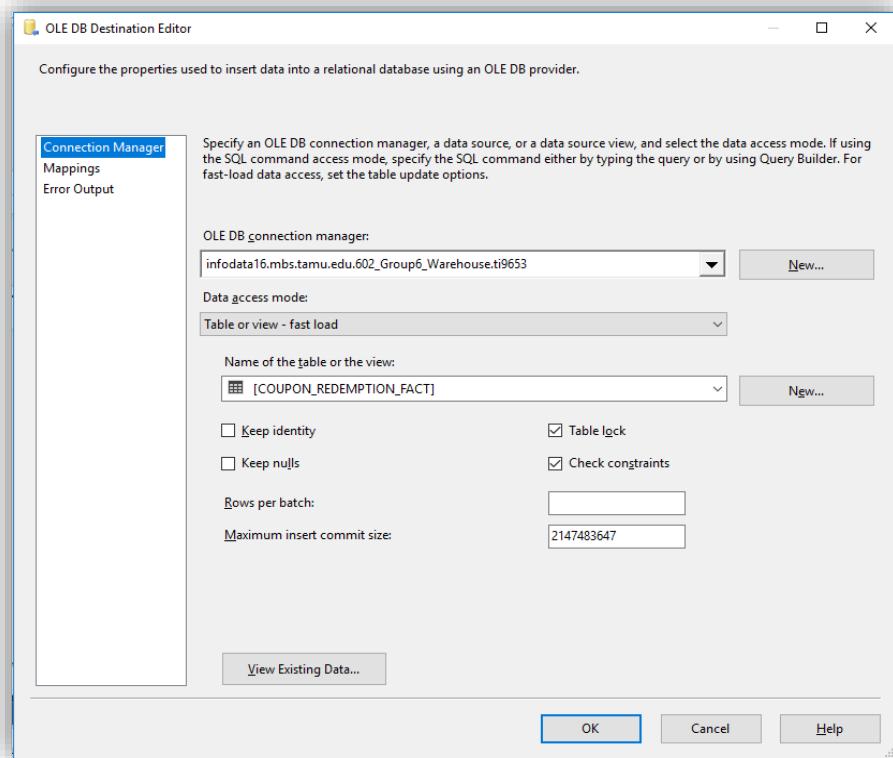
Step 4: Looking up by DIM_PRODUCT

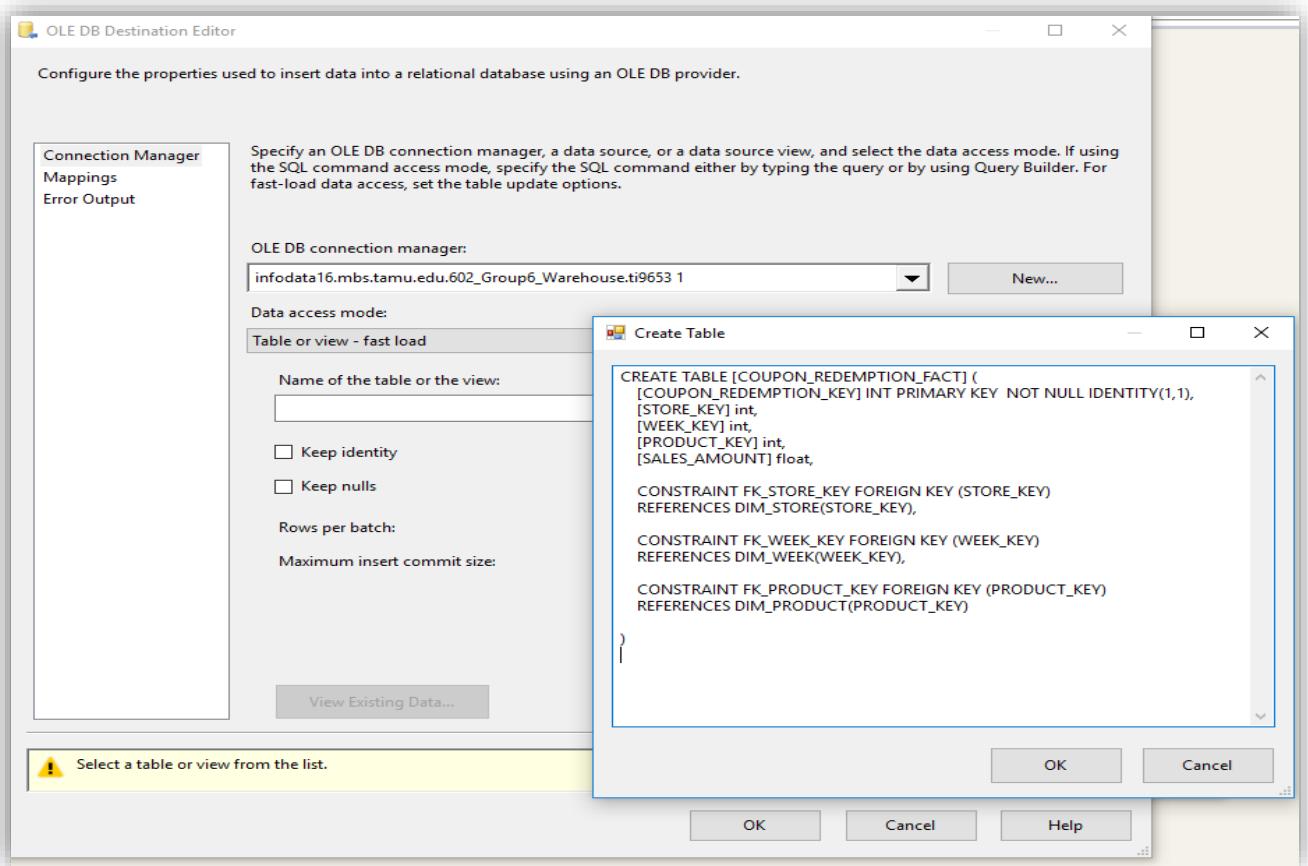


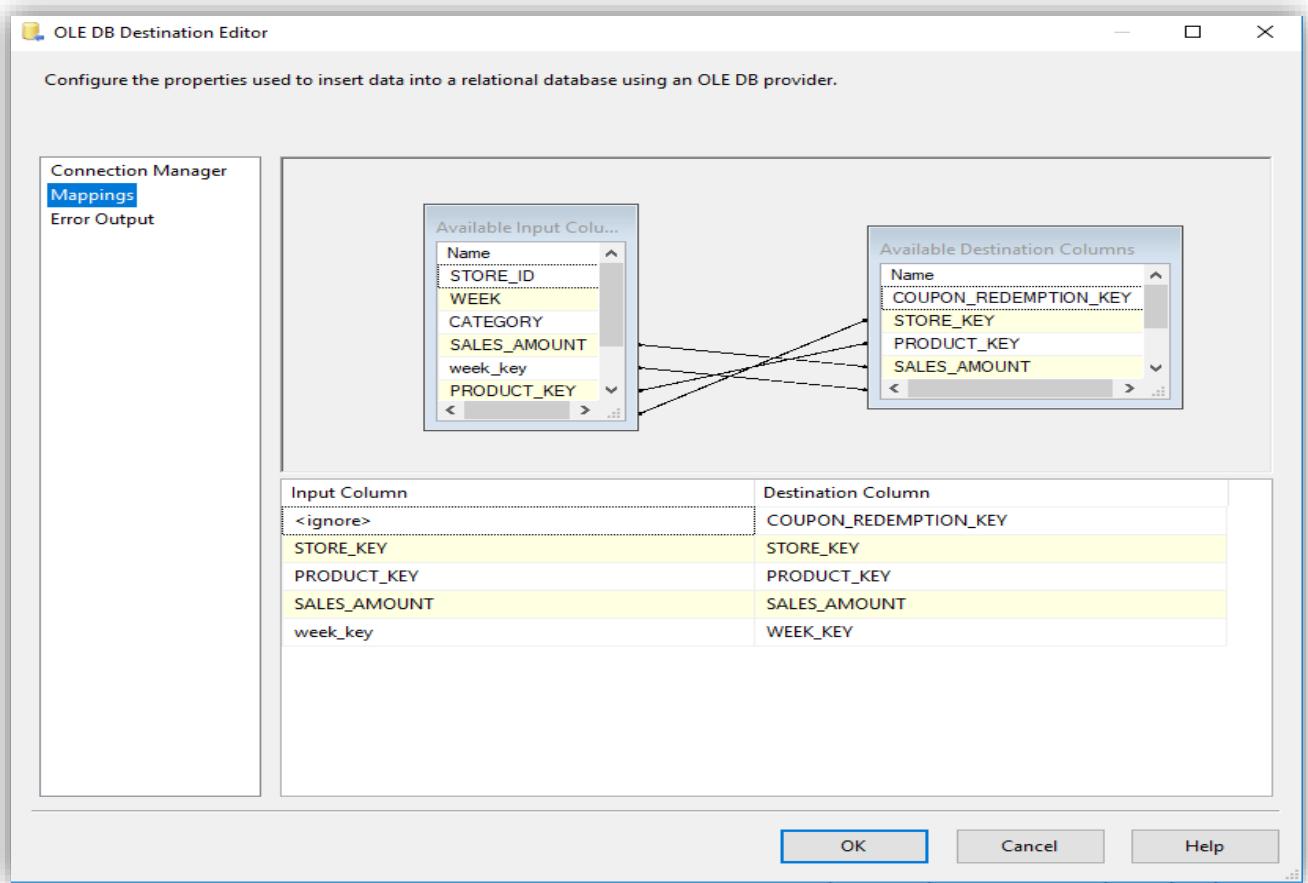
Step 5: Looking up by DIM_STORE



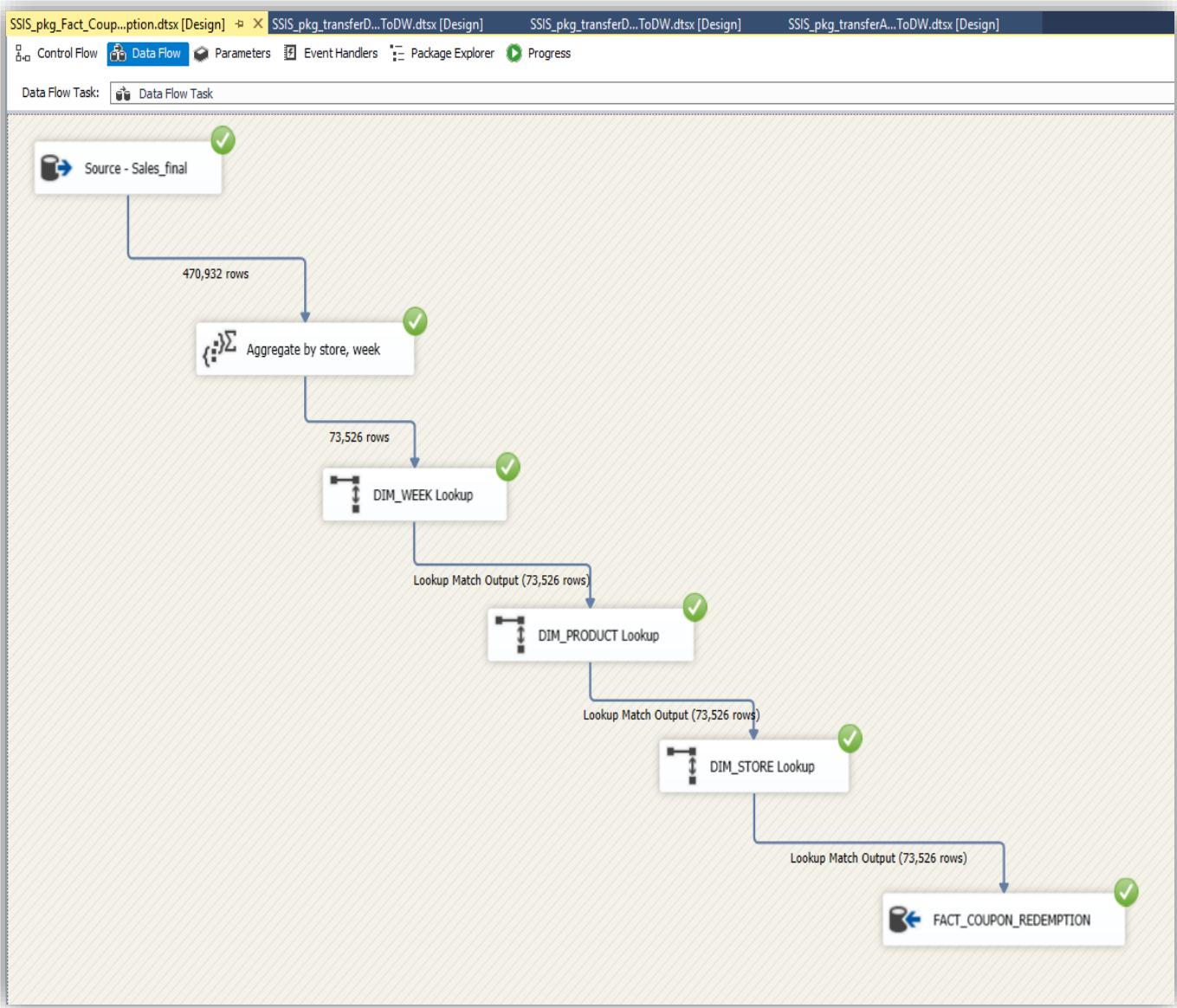
Step 6: Storing data in fact table in Data warehouse with name COUPON_REDEMPTION_FACT







Step 7: Running the SSIS



Step 8: Table after running SSIS

The screenshot shows a SQL Server Management Studio (SSMS) window. At the top, there is a script pane with the following T-SQL code:

```
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [COUPON_REDEMPTION_KEY]
, [STORE_KEY]
, [WEEK_KEY]
, [PRODUCT_KEY]
, [SALES_AMOUNT]
FROM [602_Group6_Warehouse].[dbo].[COUPON_REDEMPTION_FACT]
```

Below the script pane is a results grid. The results grid has a header row with columns: COUPON_REDEMPTION_KEY, STORE_KEY, WEEK_KEY, PRODUCT_KEY, and SALES_AMOUNT. The data rows are numbered 1 through 16. The first column contains the row number, and the second column contains the COUPON_REDEMPTION_KEY value.

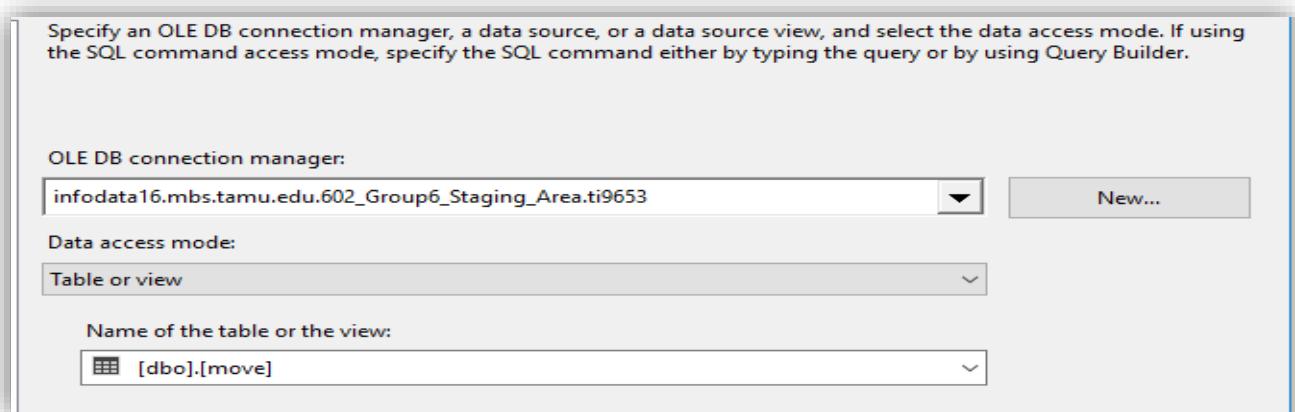
	COUPON_REDEMPTION_KEY	STORE_KEY	WEEK_KEY	PRODUCT_KEY	SALES_AMOUNT
1	1	13	281	3003	0
2	2	14	217	3003	0
3	3	26	317	3002	0
4	4	27	253	3002	0
5	5	29	125	3002	0
6	6	25	381	3002	0
7	7	94	339	3003	0
8	8	95	275	3003	0
9	9	4	108	3003	0
10	10	3	300	3003	0
11	11	5	44	3003	0
12	12	12	208	3002	0
13	13	73	230	3003	0
14	14	71	358	3003	0
15	15	76	38	3003	0
16	16	74	166	3003	0

NOTE: Since data is taken from CCOUNT which has only category not product (based on UPC) we created some product keys with these categories to maintain referential integrity

```
INSERT INTO DIM_PRODUCT (PRODUCT, CATEGORY)
VALUES (3000, 'BAKCOUP'),
VALUES (3000, 'FROZCOUP'),
VALUES (3000, 'BAKERY'),
VALUES (3000, 'FROZEN');
```

4.10.3 Creating SALES_MOVE_FACT

Step 1: Creating OLE source to fetch data from [DBO].[MOVE] (MOVE discussed in transformation)

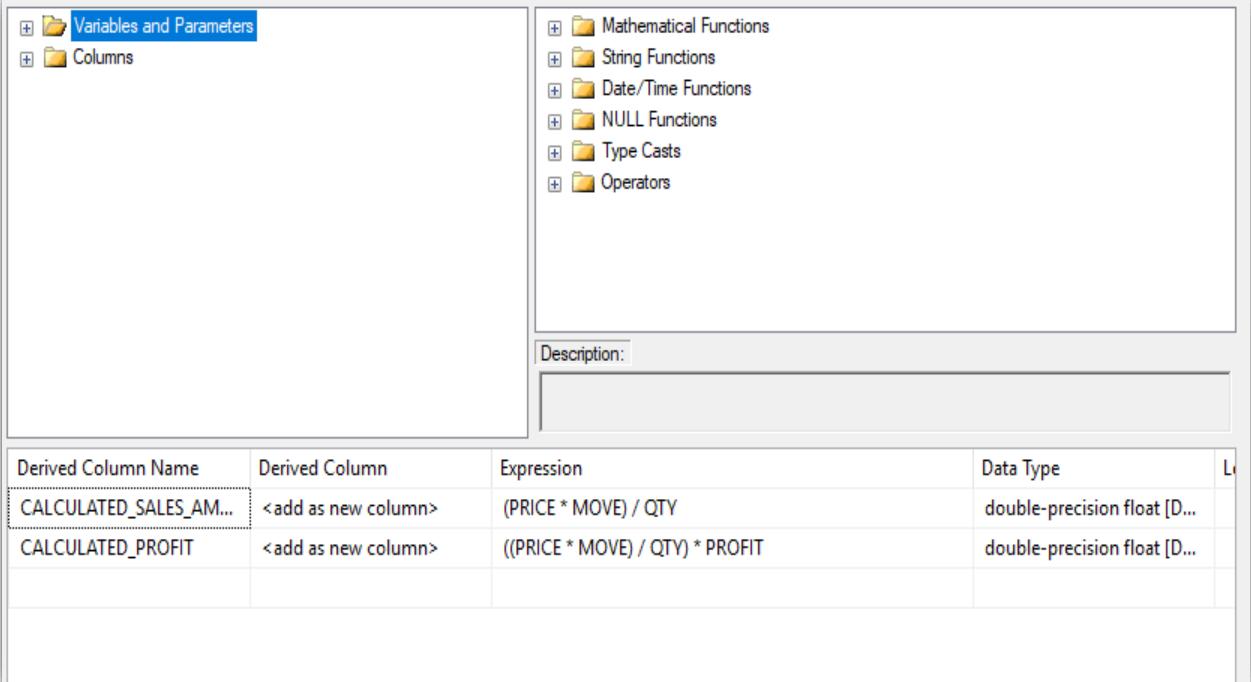


The interface shows the "Available External Columns" dialog box at the top right, listing columns: STORE_ID, UPC, WEEK, MOVE, QTY, PRICE, SALE, PROFIT, and OK. The "STORE_ID" column is selected.

The main area displays a mapping grid:

External Column	Output Column
STORE_ID	STORE_ID
UPC	UPC
WEEK	WEEK
MOVE	MOVE
QTY	QTY
PRICE	PRICE
SALE	SALE
PROFIT	PROFIT

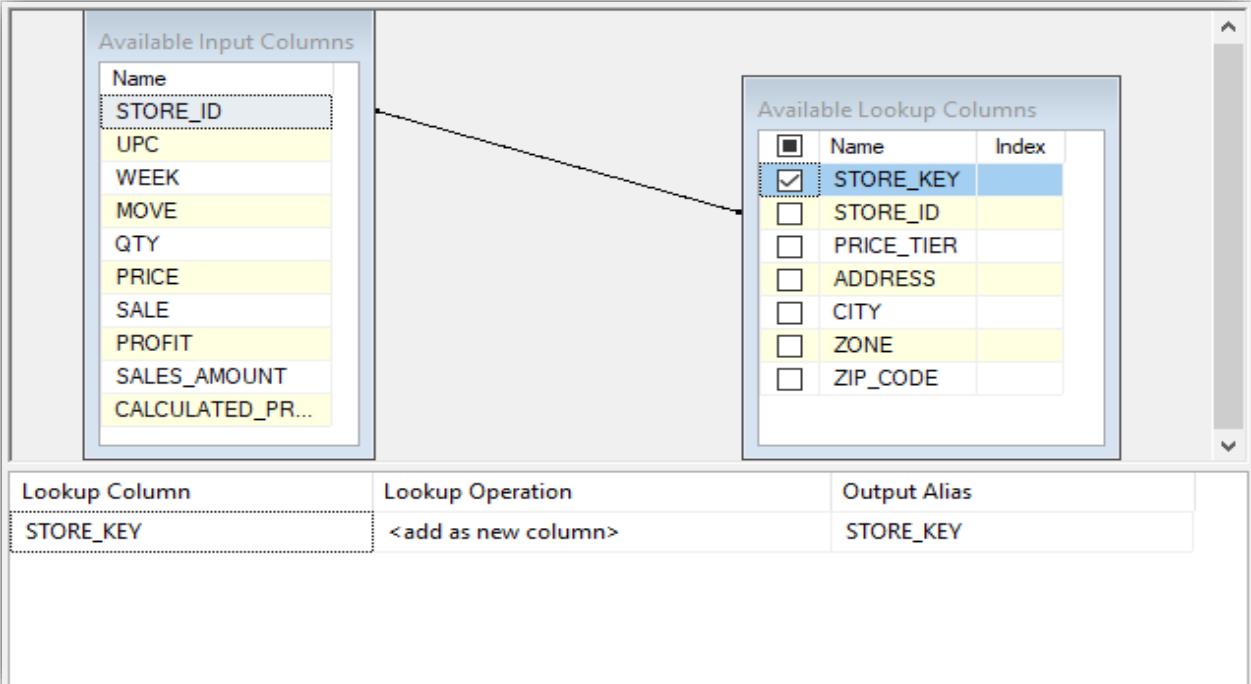
Step 2: To calculate sales and profit, we used derived column



The screenshot shows the SSIS Derived Column transformation editor. On the left, there's a tree view with 'Variables and Parameters' and 'Columns'. On the right, a list of built-in functions is shown under categories: Mathematical Functions, String Functions, Date/Time Functions, NULL Functions, Type Casts, and Operators. Below the functions is a 'Description:' text box. At the bottom is a table for defining derived columns:

Derived Column Name	Derived Column	Expression	Data Type	Length
CALCULATED_SALES_AM...	<add as new column>	(PRICE * MOVE) / QTY	double-precision float [D...]	15
CALCULATED_PROFIT	<add as new column>	((PRICE * MOVE) / QTY) * PROFIT	double-precision float [D...]	15

Step 3: Looking up by DIM_STORE



The screenshot shows the SSIS Lookup transformation editor. It has two main panes: 'Available Input Columns' on the left and 'Available Lookup Columns' on the right. A line connects the 'STORE_ID' column from the input pane to the 'STORE_KEY' column in the lookup pane.

Available Input Columns

Name
STORE_ID
UPC
WEEK
MOVE
QTY
PRICE
SALE
PROFIT
SALES_AMOUNT
CALCULATED_PR...

Available Lookup Columns

Name	Index
STORE_KEY	
STORE_ID	
PRICE_TIER	
ADDRESS	
CITY	
ZONE	
ZIP_CODE	

Lookup Column

Lookup Column	Lookup Operation	Output Alias
STORE_KEY	<add as new column>	STORE_KEY

Step 4: Looking up by DIM_PRODUCT

The screenshot shows a data integration interface with two main panels: "Available Input Columns" and "Available Lookup Columns".

Available Input Columns:

Name
STORE_ID
UPC
WEEK
MOVE
QTY
PRICE
SALE
PROFIT
SALES_AMOUNT
CALCULATED_PR...
STORE_KEY

Available Lookup Columns:

	Name	Index
<input checked="" type="checkbox"/>	PRODUCT_KEY	
<input type="checkbox"/>	PRODUCT	
<input type="checkbox"/>	ITEM_CODE	
<input type="checkbox"/>	UPC	
<input type="checkbox"/>	SIZE	
<input type="checkbox"/>	CASE	
<input type="checkbox"/>	CATEGORY	
<input type="checkbox"/>	DESCRIPTION	
<input type="checkbox"/>	COMMODITY_...	

Lookup Column Mapping:

Lookup Column	Lookup Operation	Output Alias
PRODUCT_KEY	<add as new column>	PRODUCT_KEY

Step 5: Looking up by DIM_WEEK

The screenshot shows a data integration interface with two main panels: "Available Input Columns" and "Available Lookup Columns".

Available Input Columns:

Name
STORE_ID
UPC
WEEK
MOVE
QTY
PRICE
SALE
PROFIT
SALES_AMOUNT
CALCULATED_PROFIT
STORE_KEY
PRODUCT_KEY

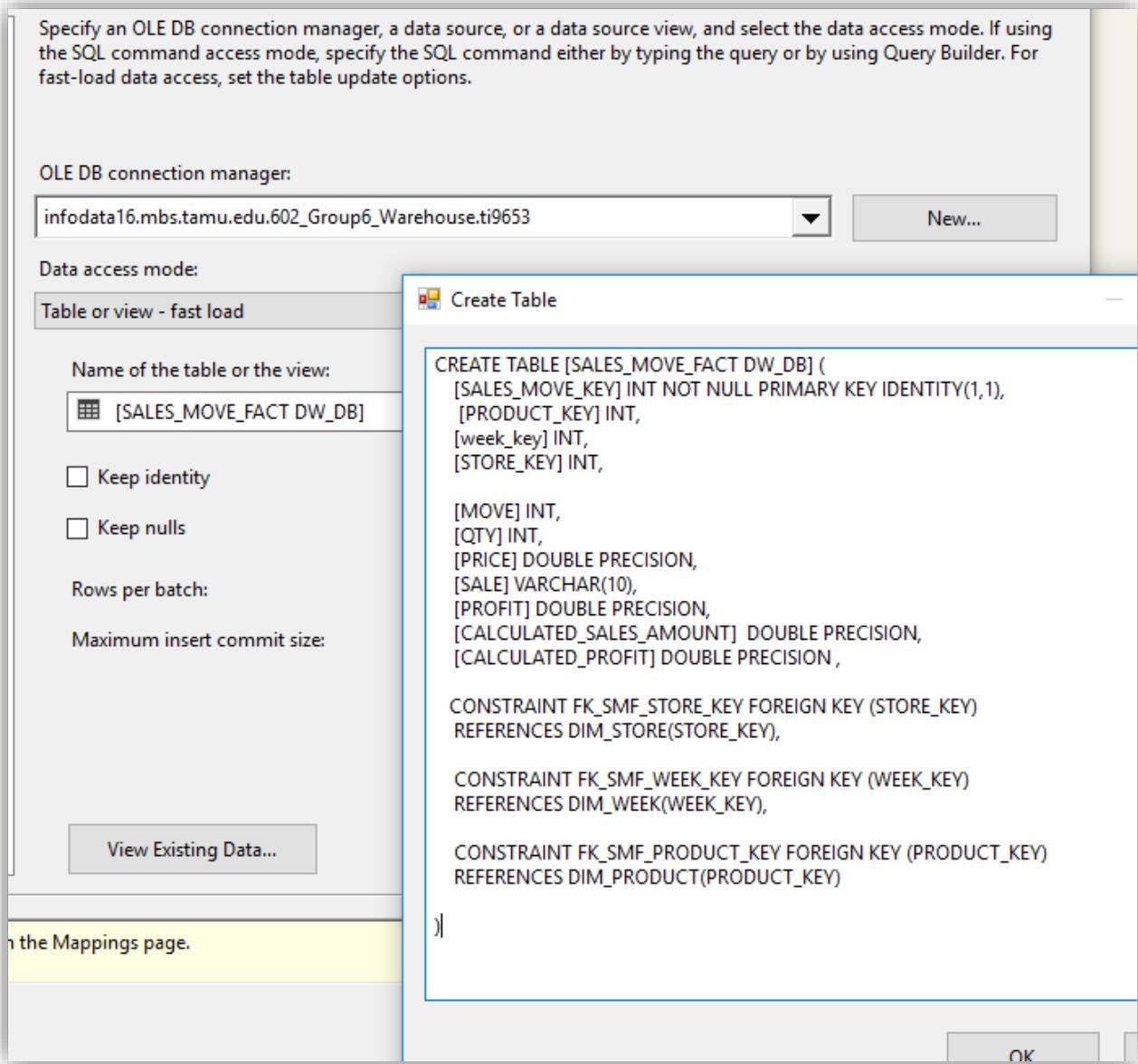
Available Lookup Columns:

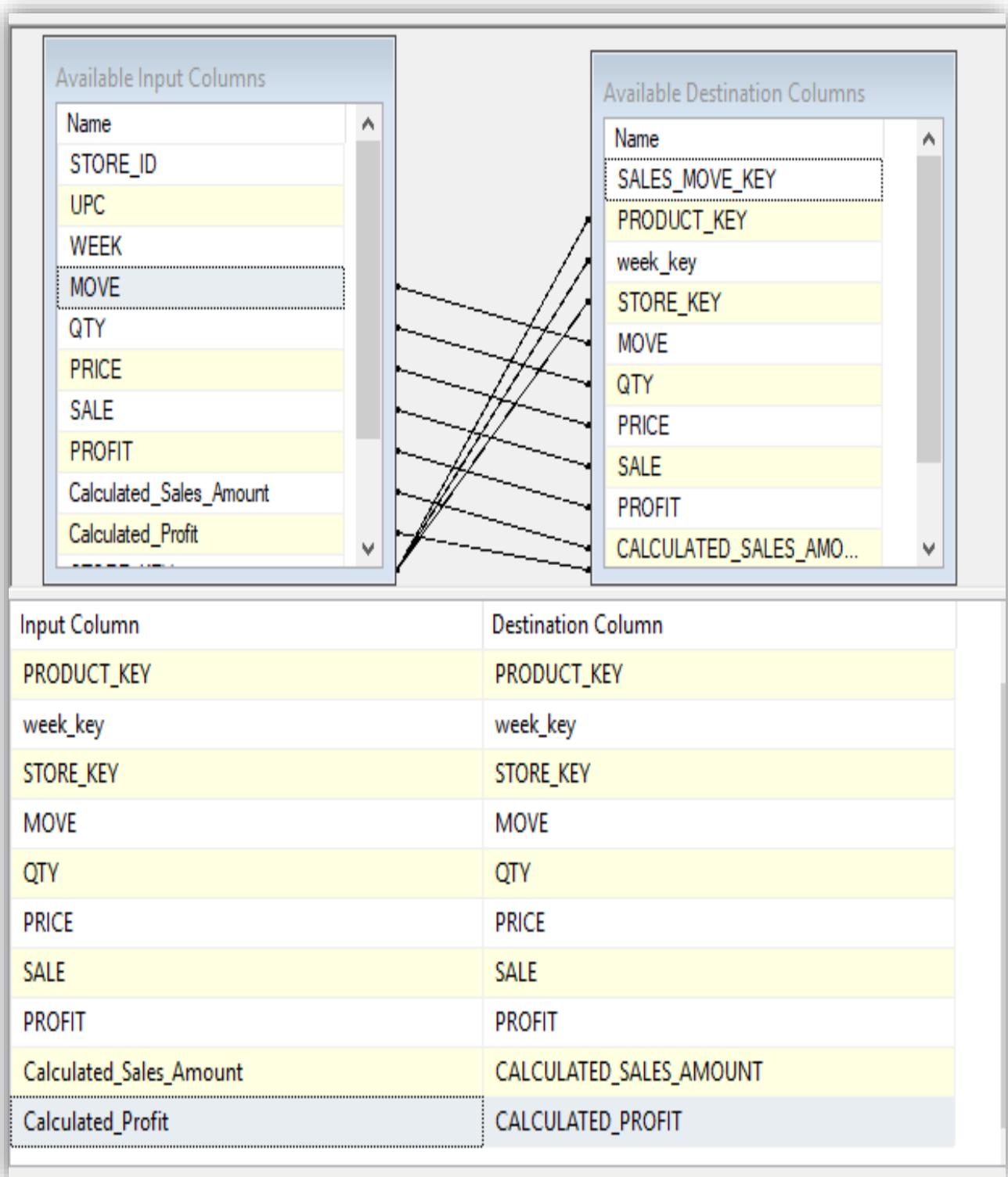
	Name	Index
<input checked="" type="checkbox"/>	week_key	
<input type="checkbox"/>	week_number	
<input type="checkbox"/>	start	
<input type="checkbox"/>	end	
<input type="checkbox"/>	special_events	

Lookup Column Mapping:

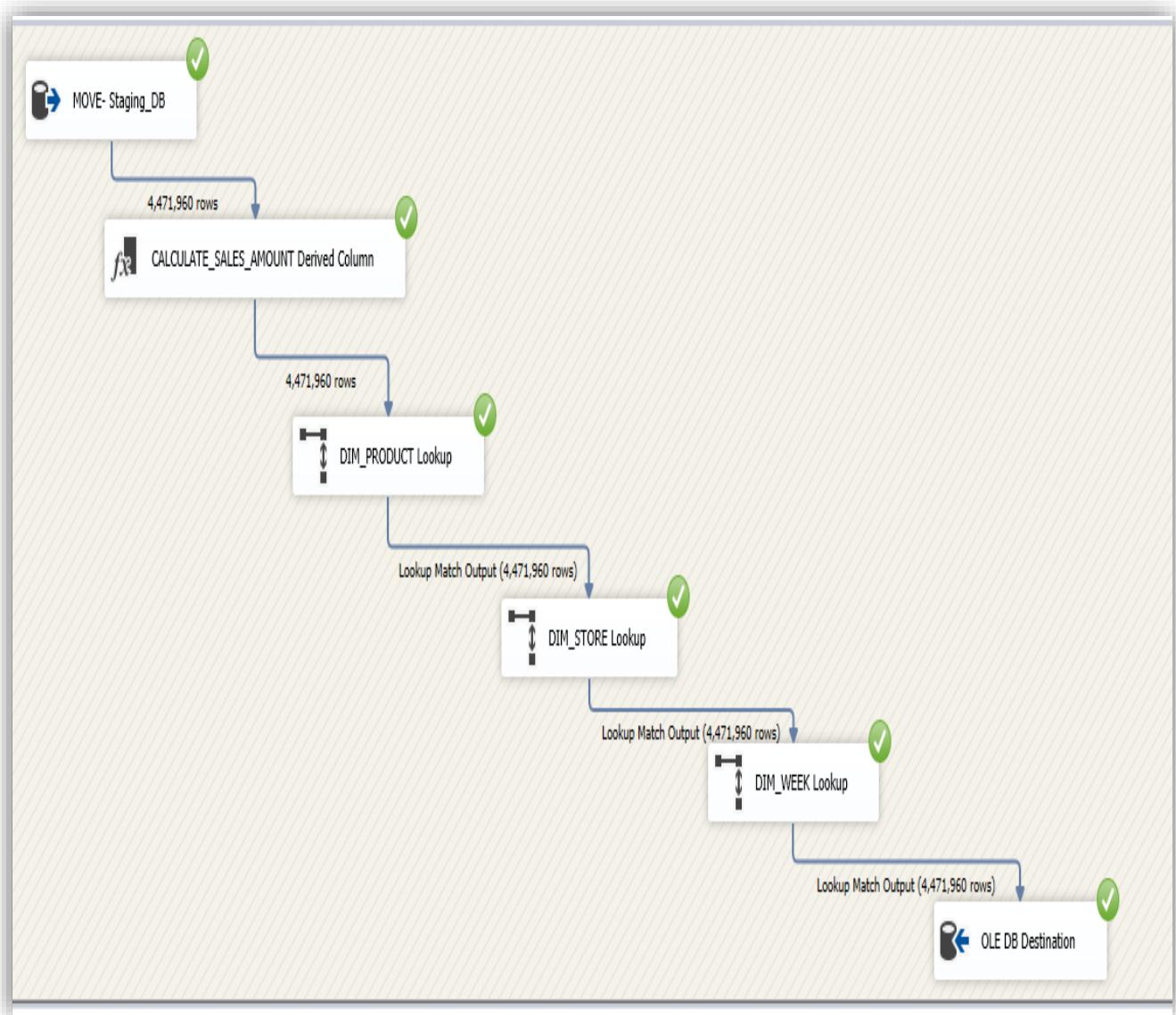
Lookup Column	Lookup Operation	Output Alias
week_key	<add as new column>	week_key

Step 6: Creating destination DB in Data warehouse





Step 7: Running SSIS Package



Step 8: SALES_MOVE_FACT table after running SSIS commands

```

***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [SALES_MOVE_KEY]
    ,[PRODUCT_KEY]
    ,[week_key]
    ,[STORE_KEY]
    ,[MOVE]
    ,[QTY]
    ,[PRICE]
    ,[SALE]
    ,[PROFIT]
    ,[CALCULATED_SALES_AMOUNT]
    ,[CALCULATED_PROFIT]
FROM [602_Group6_Warehouse].[dbo].[SALES_MOVE_FACT]

```

	SALES_MOVE_KEY	PRODUCT_KEY	week_key	STORE_KEY	MOVE	QTY	PRICE	SALE	PROFIT	CALCULATED_SALES_AMOUNT	CALCULATED_PROFIT
1	1	292	202	85	66	1	0.42	B	61.19	27.72	1696.1868
2	2	292	203	85	78	1	0.42	B	61.42	32.76	2012.1192
3	3	292	204	85	86	1	0.42	B	60.71	36.12	2192.8452
4	4	292	205	85	91	1	0.42	B	53.8	38.22	2056.236
5	5	292	206	85	62	1	0.42	B	48.57	26.04	1264.7628
6	6	292	207	85	43	1	0.42	B	48.57	18.06	877.1742
7	7	292	208	85	61	1	0.45		52	27.45	1427.4
8	8	292	209	85	43	1	0.45		58.66	19.35	1135.071
9	9	292	210	85	66	1	0.45		60.88	29.7	1808.136
10	10	292	211	85	58	1	0.46		59.13	26.68	1577.5884
11	11	292	212	85	54	1	0.49		51.02	26.46	1349.9892
12	12	292	213	85	49	1	0.49		50.4	24.01	1210.104
13	13	292	214	85	27	1	0.49		50.4	13.23	666.792
14	14	292	215	85	48	1	0.49		49.79	23.52	1171.0608
15	15	292	216	85	37	1	0.49		49.79	18.13	902.6927
16	16	292	217	85	40	1	0.49		49.59	19.6	971.964

4.11 List of all temporary tables in the Staging area

- 602_Group6_Staging_Area.CCOUNT
- 602_Group6_Staging_Area.move
- 602_Group6_Staging_Area.move_candy
- 602_Group6_Staging_Area.move_cookie
- 602_Group6_Staging_Area.move_frozendinner
- 602_Group6_Staging_Area.move_frozenentree
- 602_Group6_Staging_Area.move_frozenjuice
- 602_Group6_Staging_Area.move_temp
- 602_Group6_Staging_Area.PRODUCT
- 602_Group6_Staging_Area.product_candy
- 602_Group6_Staging_Area.product_cookie
- 602_Group6_Staging_Area.product_frozendinner
- 602_Group6_Staging_Area.product_frozenentree
- 602_Group6_Staging_Area.product_frozenjuice
- 602_Group6_Staging_Area.SALES
- 602_Group6_Staging_Area.SALES_FINAL
- 602_Group6_Staging_Area.store
- 602_Group6_Staging_Area.WEEKTABLE

4.12 Data Granularity at the independent data mart level

Data Granularity refers to the level of detail that is included in the data marts based on the requirement. The data given in CCount is at date level but the data in Movement table, which eventually provides sales data is at week level. Thus, we have taken lowest granularity level at week for the time dimension. The week dimension also consists of Start and End date for the week, which can provide us with the year in case of roll up.

The granularity for the independent data marts is at store, week and product.

5. Physical Design

5.1 Aggregation Plan

We are storing the data at the lowest granularity level of time dimension i.e. week. Of the several types of aggregation, the most common is called a roll-up aggregation. In this case, we will take the customer count and daily sales data given in the Customer Count table and roll it up for weekly sales table. Additionally, we can use roll up for this same data to get the monthly and yearly sales tables. For the sales and profit data, we can roll up the sales amounts from weekly basis to monthly and yearly sales tables. These types of summaries are easily computed from the base data warehouse by using the SQL SUM operator.

Essentially, we can either aggregate at runtime or pre-aggregate the data offline, thus making the totals available without real-time computation. The option of calculation aggregate at runtime will provide with real time data but it may affect the performance. Hence, we have come up with the alternative to write to write SQL to pre-aggregate data according to the dimensions that end users will frequently want to see.

5.2 Indexing Plan

The following columns will be used as index for the corresponding tables. Binary tree is being used to enhance performance of the indexes. Primary key is considered index for all tables.

Indexes		
Table	Index	Indexing schema
DIM_STORE	STORE_KEY, STORE_ID	Binary Tree
DIM_PRODUCT	PRODUCT_KEY, PRODUCT_ID	
DIM_DATE	TIME_KEY, WEEK, YEAR	
COUPON_REDEMPTION_FA CT	COUPON_REDEMPTION_KEY	
CUST_COUNT_FACT	CUST_COUNT_KEY	
SALES_FACT	SALES_KEY	

5.3 Data Standardization plan

The following naming standards are to be followed for the warehouse objects:

Naming Standards	
Fact tables	Ending with "_FACT"

Dimension tables	Starts with "DIM_ "
Staging tables	Starts with "STAG_ "
Aggregated tables	Starts with "AGGR"

The following standards for the name length will be followed along with the general guidelines for the naming conventions in Microsoft SQL Server databases:

Name Length	
Table	30
Column Headers	20
Keys	20

5.4 Storage plan

The table below mentions are rough estimate for table sizes with the available data with us. This size is going to vary with cleanup and addition/deletion of tuples. The estimation is done by taking size of data types and multiplying by the data available with the number of rows and columns.

Warehouse Table Size (in Megabytes)	
DIM_STORE	~ 0.30 (3 numeric, 3 string, 140 rows)
DIM_PRODUCT	~ 6 (3 numeric, 6 string, 18k rows)
DIM_TIME	~ 0.04 (3 numeric, 1 string, 400 rows)
COUPONS_REDEMPTION_FACT	~ 10 (6 numeric, 350k rows, divide by 7*)
CUST_COUNT_FACT	~ 7 (4 numeric, 350k rows, divide by 7*)
SALES_FACT	~ 2000 (7 numeric, 90M rows)

Estimated table growth every year: 10 %

Additional Table Size	
Temp area	~ 500 MB (auto grow by 128 MB)
Staging area	~ 2000 MB
Indexes	~ 500 MB

OLAP system files	~ 1500 MB
Application	~ 2000 MB

Estimated growth every year: 15 %

5.5 Partitioning Plan

The tables to be partitioned:

- DIM_PRODUCT: Since it has many columns and huge data, this dimension table has to be partitioned vertically.
- SALES_FACT: Since this table contains millions of rows, this table should be horizontally partitioned.

Also, the corresponding queries need to be optimized for recognizing the partitions.

6. BI Reporting

6.1 Reporting plan

6.1.1 Target Reports & Template

- a. Weekly trend of customer in traffic for store 2 between 1995 and 1996

Report using SSRS

This report will have two columns which tell us the customer traffic in the store #2 as Customer Count and Week Number.

Customer Count	Week Number

- b. In category of frozen products, the method of discount (Coupon or price reduction) gives more profit

Type: Report using Report Builder

Report for the category of frozen products will have two columns Sale, B or S, which depicts is the discount type and the average calculated profit.

Sale	Average Calculated Profit

- c. To find UPCs had most sale in high, medium and low tier for cookies

Type: Report using SSAS and SSRS

To find the UPCs with most sale amount in the three tiers, we need columns UPC, Category, Calculated Sales amount and Calculated Profit. The page level in SSRS will be tier values '*high*', '*medium*', and '*low*'.

UPC	Category	Calculated Sales amount	Calculated Profit

- d. Trend of candies' sale during Halloween year by year

Type: Report using SSAS

To capture sales of candies during Halloween for each year, columns required are Category='Candy', year, Special Event='Halloween', Week Number, Calculated Sales amount and Calculated Profit. This data will be the basis for creating a pivot table in Excel.

Category	Year	Special Events	Week Number	Calculated Sales amount	Calculated Profit

- e. Relation between bakery sales and bakery coupons redeemed for all stores

Type: Report using Report Builder

To find the sales of bakery and bakery coupons redeemed, columns required in the report are Store ID, Category (Bakery or Bakery Coupon) and the sum of sales amount.

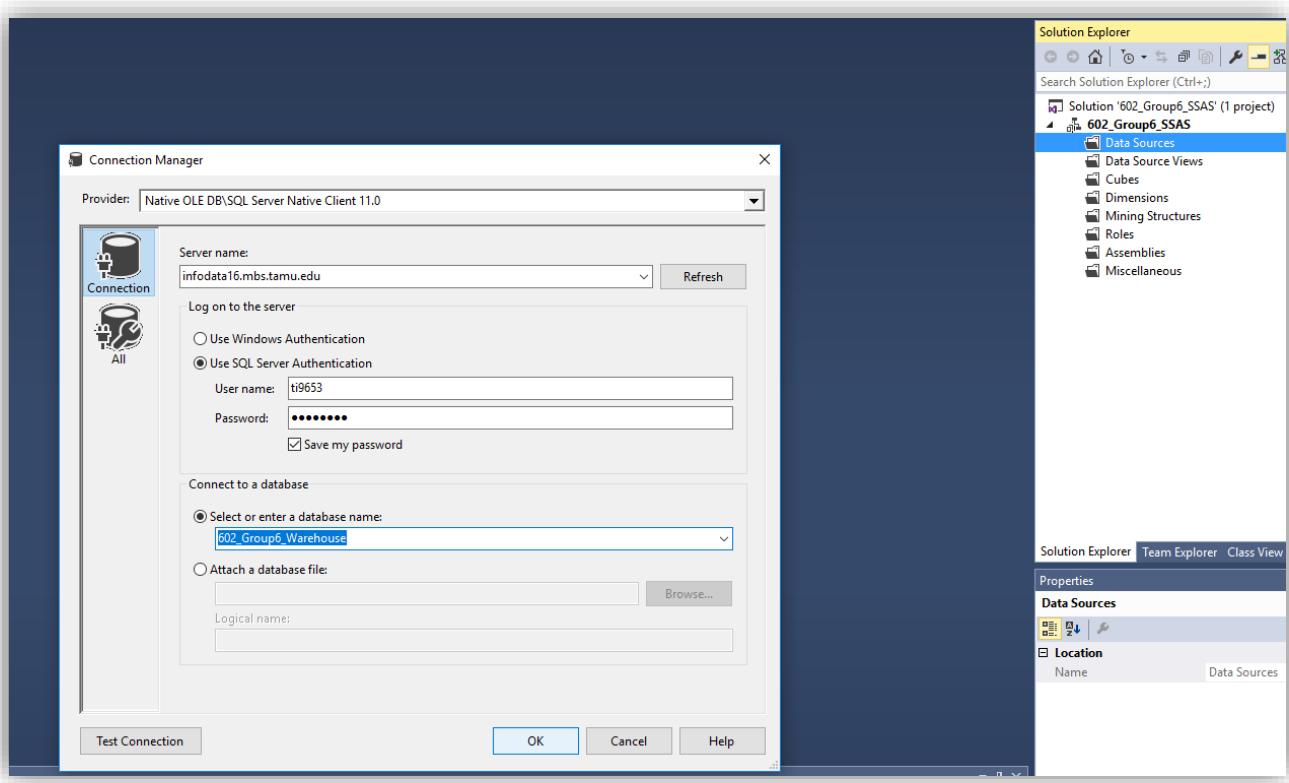
Store ID	Category	Sum Sales Amount

6.1.2 Mapping for report attributes from data warehouse

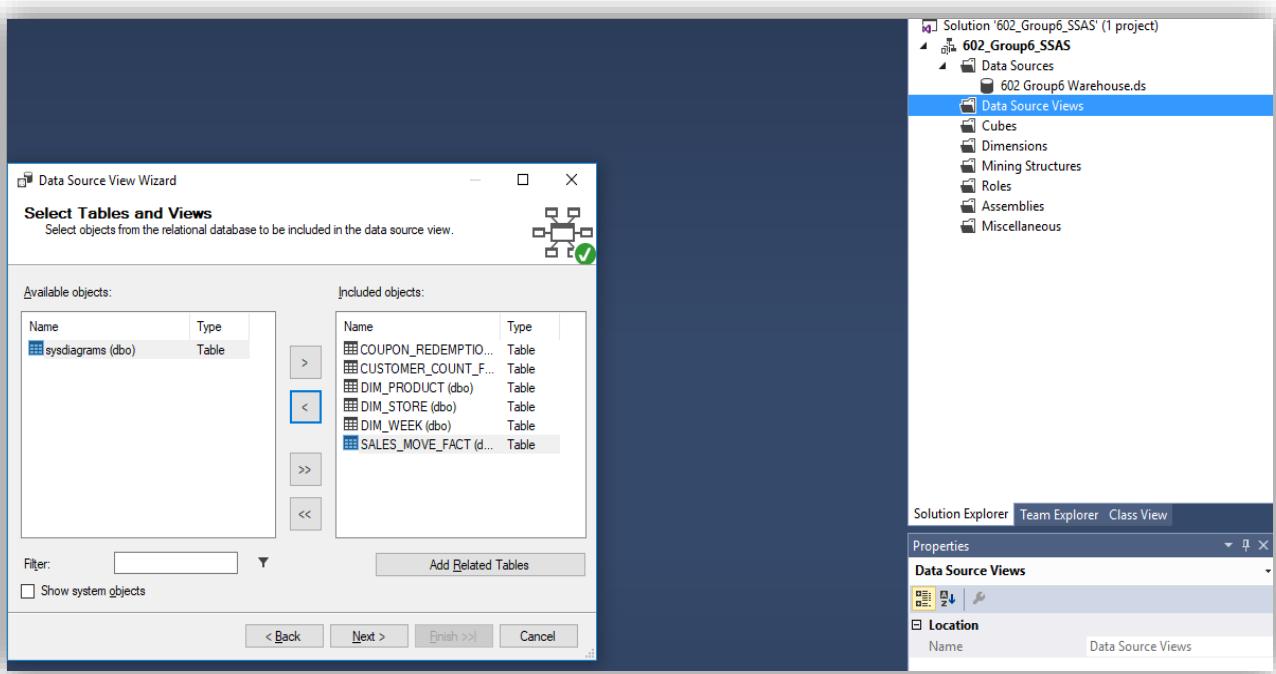
Data Warehouse Table Name	Data Warehouse Column Name	Report Column Name
CUST_COUNT_FACT	CUSTOMER_COUNT	Customer Count
DIM_WEEK	WEEK_KEY	Week Number
SALES_MOVE_FACT	SALE	Sale
SALES_MOVE_FACT	CALCULATED_PROFIT	Average Calculated Profit
DIM_PRODUCT	UPC	UPC
DIM_PRODUCT	CATEGORY	Category
DIM_WEEK	START/END	Year
DIM_WEEK	SPECIAL_EVENTS	Special Events
SALES_MOVE_FACT	CALCULATED_SALES_AMOUNT	Calculated Sales Amount
SALES_MOVE_FACT	CALCULATED_PROFIT	Calculated Profit
DIM_STORE	STORE_ID	Store ID
SALES_MOVE_FACT	CALCULATED_SALES_AMOUNT	Sum Sales Amount
DIM_STORE	PRICE_TIER	Price Tier

6.2 Cube generation using SSAS

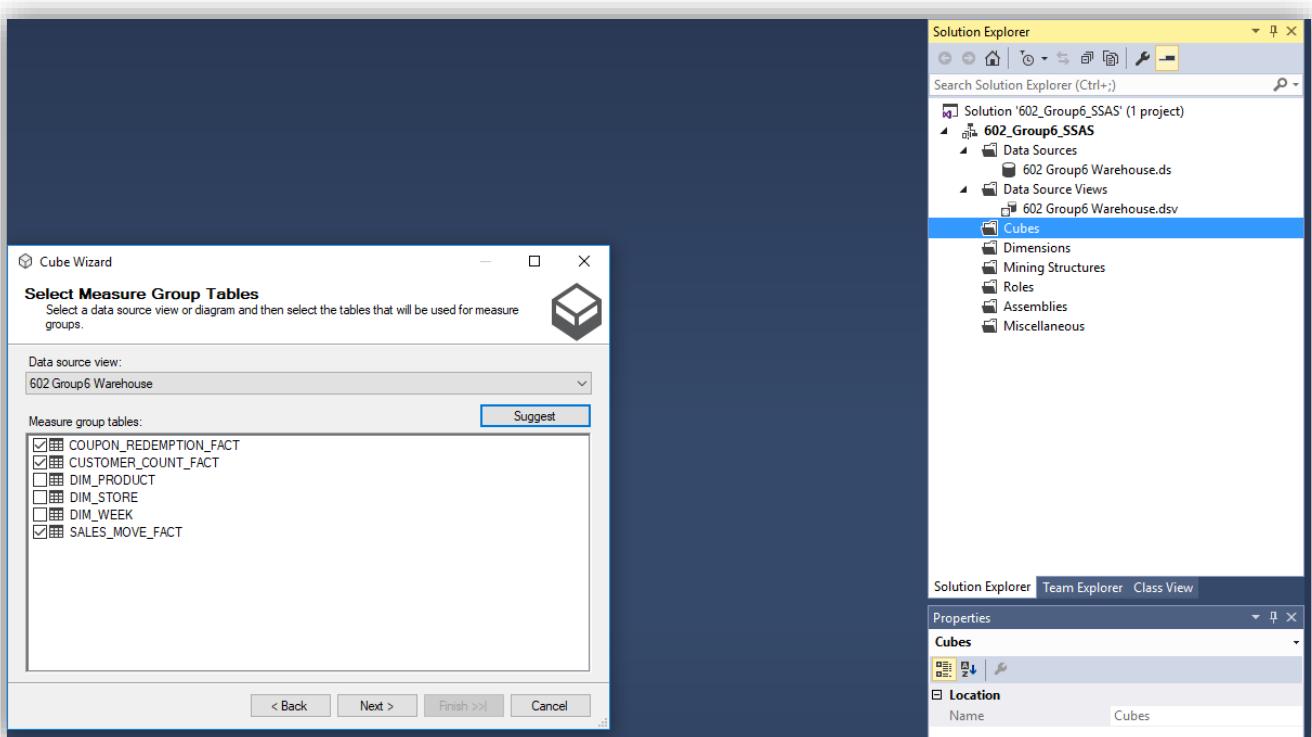
Step 1: Selecting data source



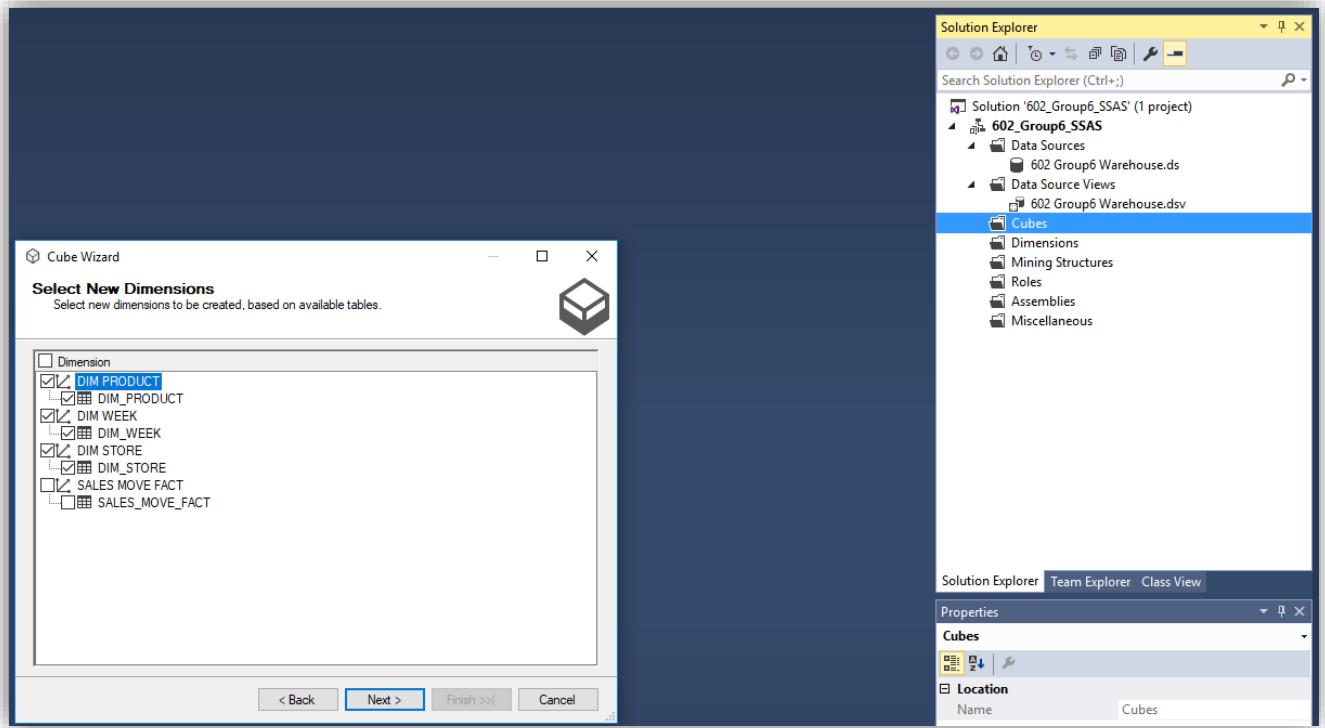
Step 2: Creating Data source view



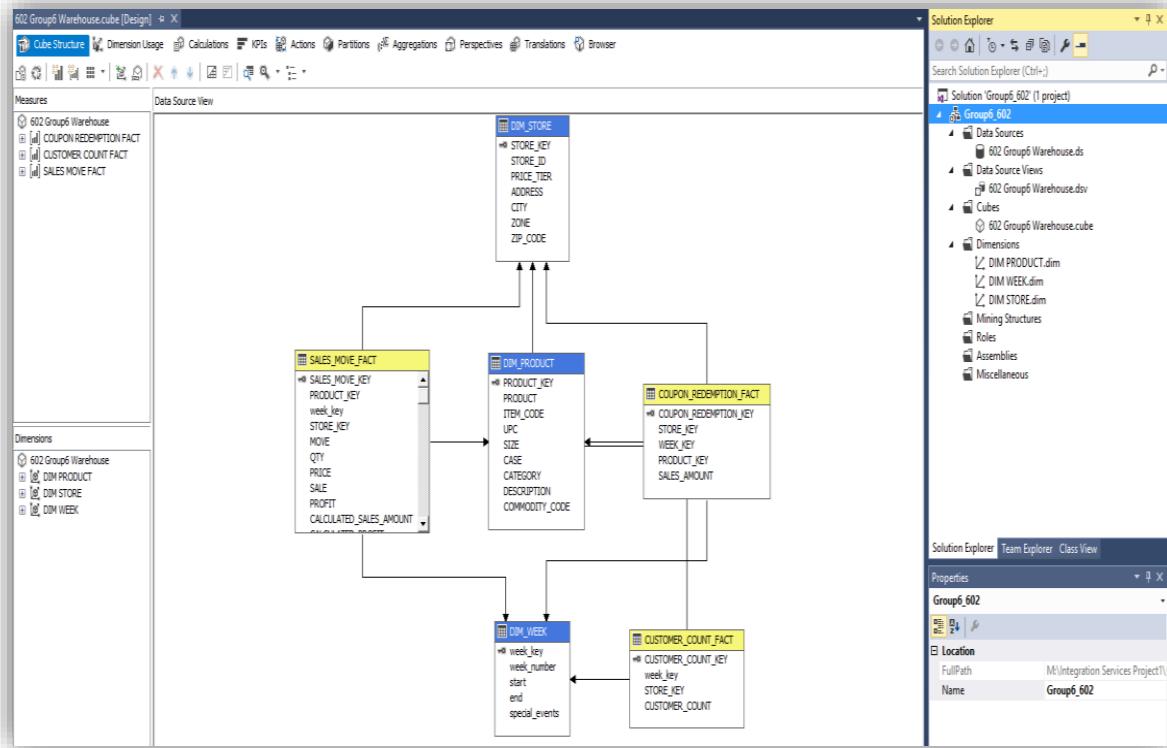
Step 3: Selecting Measures



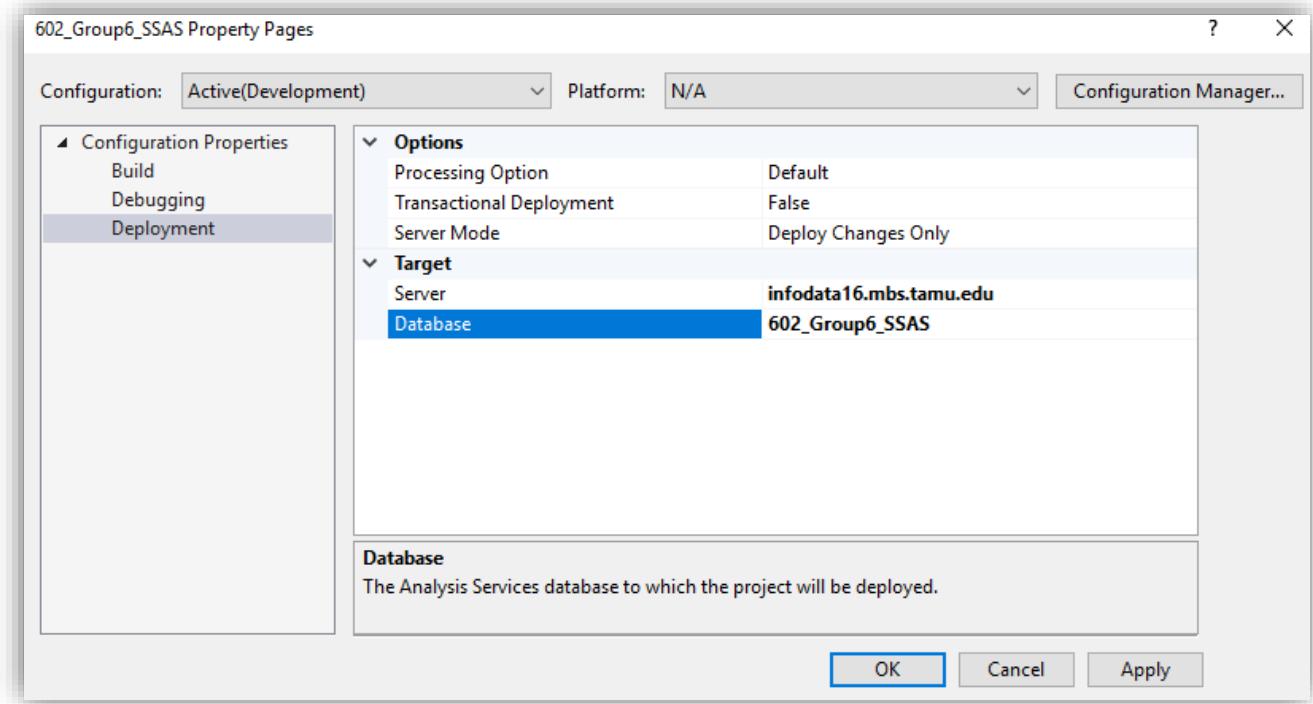
Step 4: Selecting Dimensions



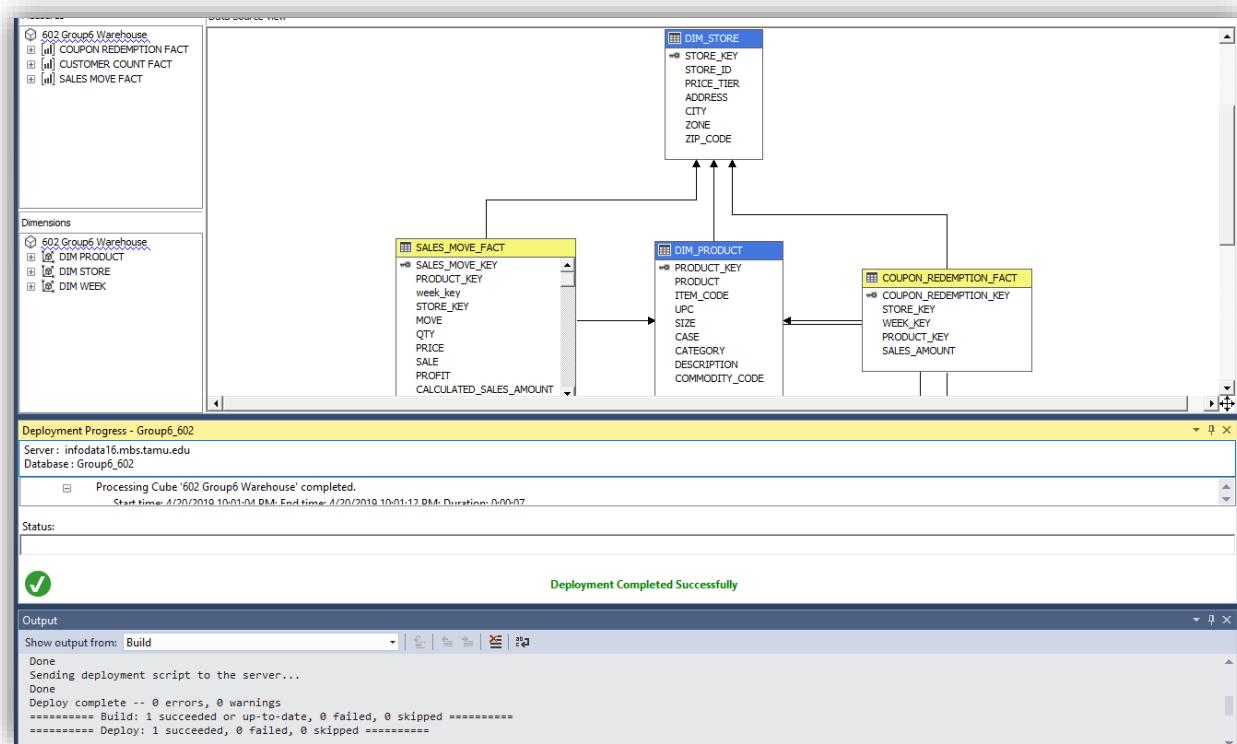
Step 5: Generating Cube

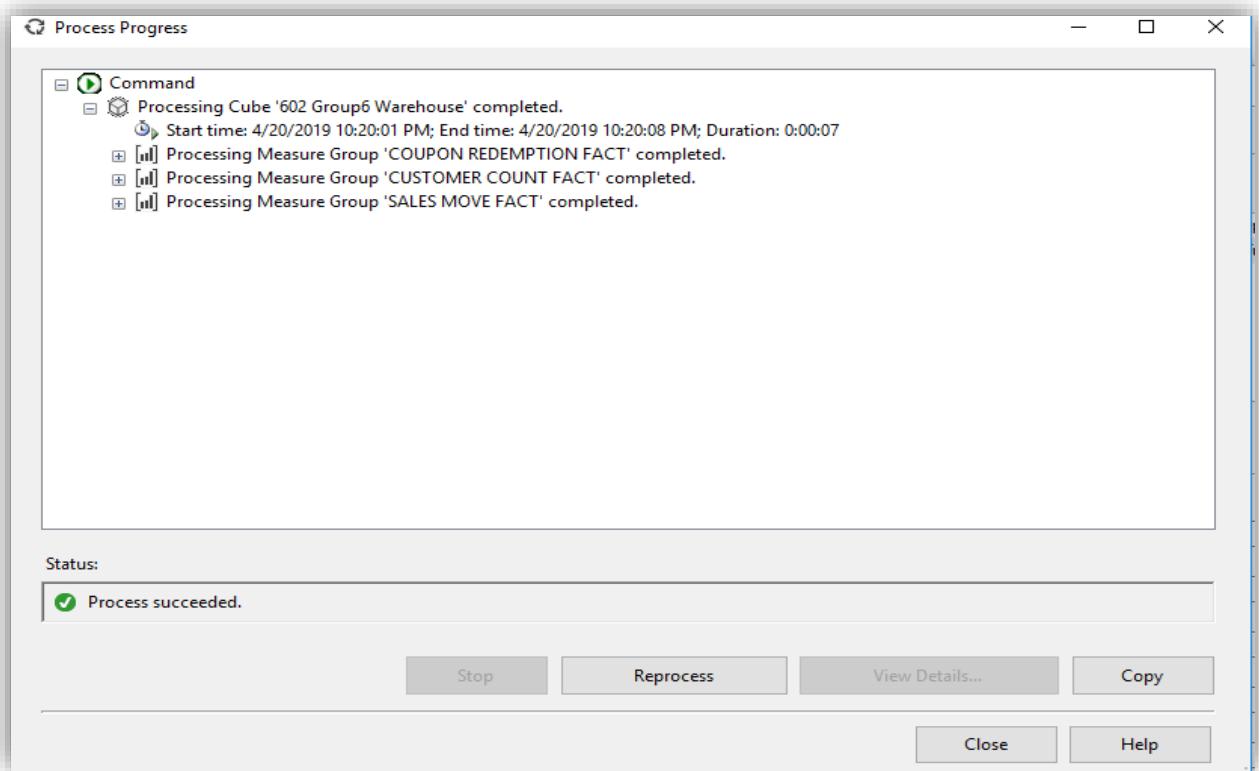
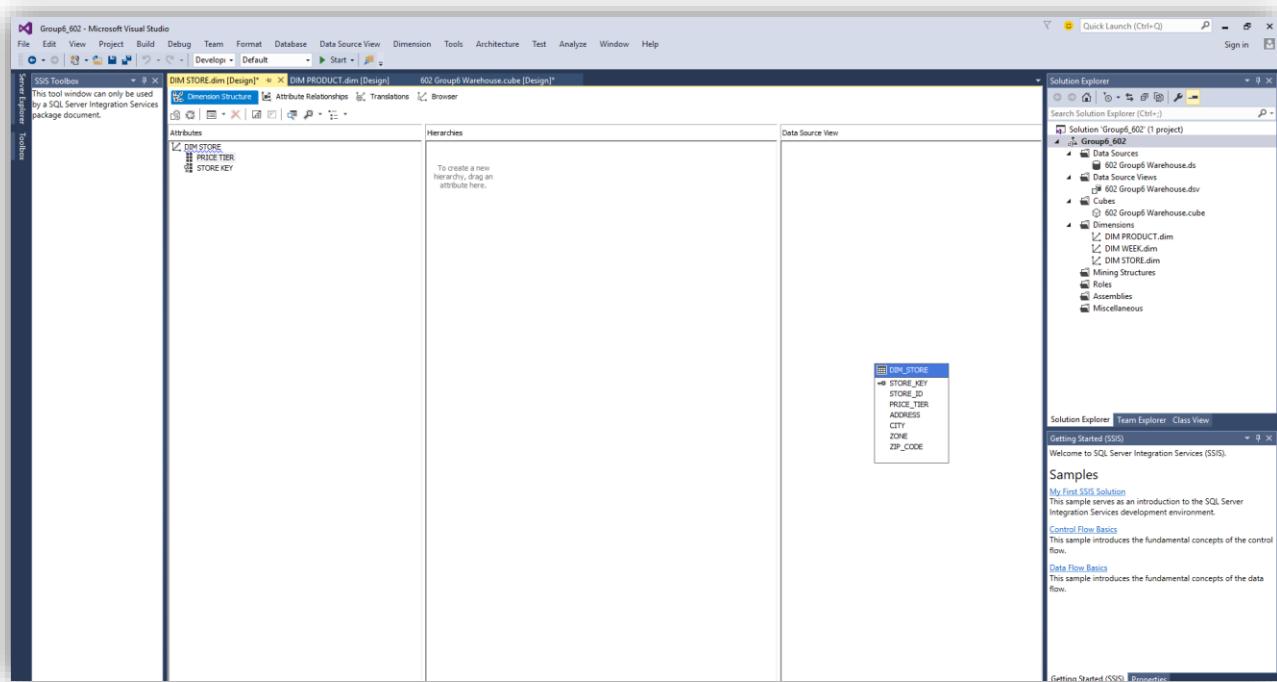


Step 6: Setting up Database connection



Step 7: Deploying to database





Group6_602 - Microsoft Visual Studio

File Edit View Project Build Debug Team Database Cube Tools Architecture Test Analyze Window Help

Server Explorer Toolbox

SSIS Toolbox This window can only be used by a SQL Server Integration Services package document.

DIM PRODUCT.dim [Design] 602 Group6 Warehouse.cube [Design] □

Cube Structure Dimension Usage Calculations KPIs Actions Partitions Aggregations Perspectives Translators Browser

Edit as Text Import... MDX Language: Default

Dimension Hierarchy Operator Filter Expression Parameters

602 Group6 Warehouse

Metadata Measure Group: <Select dimension>

UPC CATEGORY PRICE TIER CALCULATED SALES AMOUNT CALCULATED PROFIT

10...	Candy	High	9466.12	4202.0496800284
10...	Candy	Low	1642.55	748.1939780009
10...	Candy	Medium	7843.5400000001	3385.9546289995
10...	Candy	CubFighter	10112.81	4290.4795799978
10...	Candy	High	12890.33	5850.29926799973
10...	Candy	Low	11100.09999985	4890.0427068023
10...	Candy	Medium	27895.2099999999	12072.335199998
10...	Candy	None	2562.41	1111.1662380027
10...	Candy	High	311.333333333333	89.632866666667
10...	Candy	Low	193.333333333333	55.660666666667
10...	Candy	Medium	197	56.691023323333
10...	Candy	CubFighter	474.866666666667	136.631266666667
10...	Candy	High	975.666666666667	282.046033333333
10...	Candy	Low	587.666666666667	169.1387
10...	Candy	Medium	1541.666666666667	443.770033333333
10...	Candy	None	97	27.5263
10...	Candy	High	257.436666666667	128.541186333333
10...	Candy	Low	126.286666666667	73.00662
10...	Candy	Medium	249.31	125.364746
10...	Candy	CubFighter	358.296666666667	182.045038666667
10...	Candy	High	799.803333333333	416.411783
10...	Candy	Low	558.62	208.778715
10...	Candy	Medium	1420.023333333333	707.468541333334
10...	Candy	None	20.55	5.55505

Calculated Members

Deployment Progress - Group6_602

Server: infodata16.mbstemu.edu Database: Group6_602

Command

Status:

Deployment Completed Successfully

Solution Explorer

Search Solution Explorer (Ctrl+F)

Solution 'Group6_602' (1 project)

- Data Sources
 - 602 Group6 Warehouse.ds
 - Data Source Views
 - 602 Group6 Warehouse.ds.vw
- Cubes
 - 602 Group6 Warehouse.cube
- Dimensions
 - DIM PRODUCT.dim
 - DIM WEEK.dim
 - DIM STORE.dim
 - Mining Structures
- Roles
- Assemblies
- Miscellaneous

Samples

My First SSIS Solution

This sample serves as an introduction to the SQL Server Integration Services development environment.

Control Flow Basics

This sample introduces the fundamental concepts of the control flow.

Data Flow Basics

This sample introduces the fundamental concepts of the data flow.

Getting Started (SSIS) Properties

Step 8: Data in SQL server

602 Group6 Warehouse [Browse] - Microsoft SQL Server Management Studio

File Edit View Project Debug Cube Tools Window Help

Connect Connect...

Object Explorer

602 Group6 Warehouse

SQLQuery2.sql - inf... Area (09653 (128))

Properties

ObjectName: 602 Group6 Warehouse

ViewersConnector

ViewerType: Cube

Dimension Hierarchy Operator Filter Expression Parameters

602 Group6 Warehouse

Metadata Measure Group: <Select dimension>

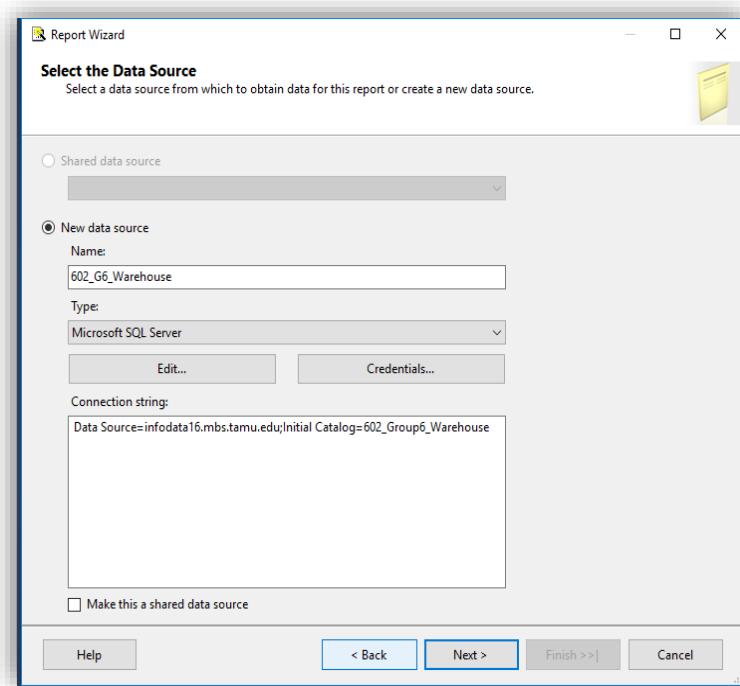
UPC PRICE TIER CALCULATED SALES AMOUNT CALCULATED PROFIT

107000039	High	9466.12	4202.0496800284
107000039	Low	1642.55	748.1939780009
107000039	Medium	7843.5400000001	3385.9546289995
107000039	CubFighter	10112.81	4290.4795799978
107000039	High	12890.33	5850.29926799973
107000039	Low	11100.09999985	4890.0427068023
107000039	Medium	27895.2099999999	12072.335199998
107000039	None	2562.41	1111.1662380027
10700008	High	311.333333333333	89.632866666667
10700008	Low	193.333333333333	55.660666666667
10700008	Medium	197	56.691023333333
10700008	CubFighter	474.866666666667	136.631266666667
10700008	High	975.666666666667	282.046033333333
10700008	Low	587.666666666667	169.1387
10700008	Medium	1541.666666666667	443.770033333333
10700008	None	97	27.5263
10700285	High	257.436666666667	128.541186333333
10700285	Low	126.286666666667	73.00662
10700285	Medium	249.31	125.364746
10700285	CubFighter	358.296666666667	182.045038666667
10700285	High	799.803333333333	416.411783
10700285	Low	558.62	208.778715
10700285	Medium	1420.023333333333	707.468541333334
10700285	None	20.55	5.55505
10700288	High	309.95	90.05547
10700288	Low	200.666666666667	57.771933333333
10700288	Medium	430.5	124.734
10700288	CubFighter	747.333333333333	216.742166666667
10700288	High	1365.556666666667	397.1461833333
10700288	Low	760.983333333333	220.064496666667
10700288	Medium	1030.023333333333	309.011866666667
10700288	None	98.46666666666667	28.406132666667
107003185	High	1377.379	462.510492
107003185	Low	473.191666666667	159.002666666667
107003185	Medium	1197.6	397.883023

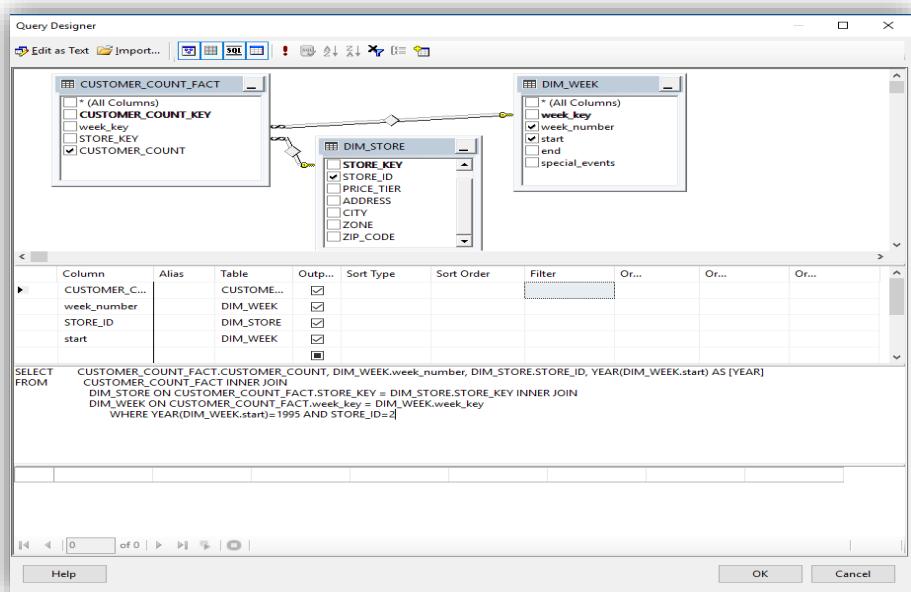
Calculated Members

6.3 Report using SSRS (Business Question #1)

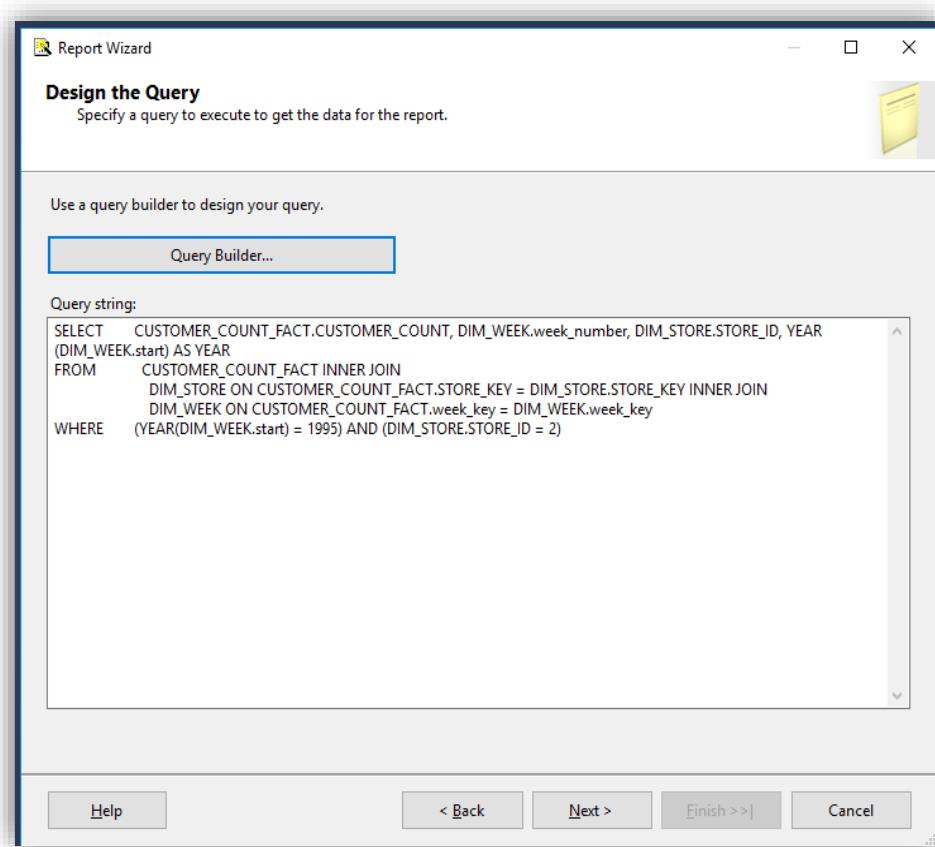
Step 1: Establishing connection to data warehouse



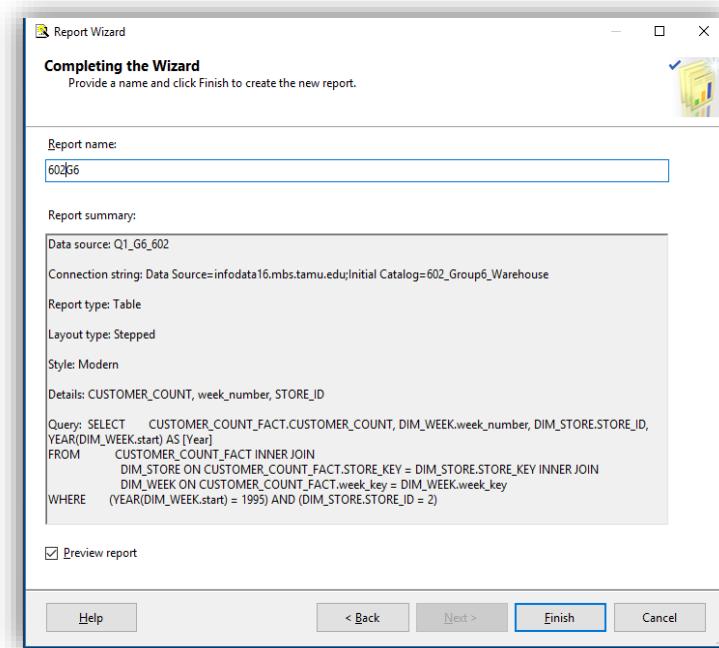
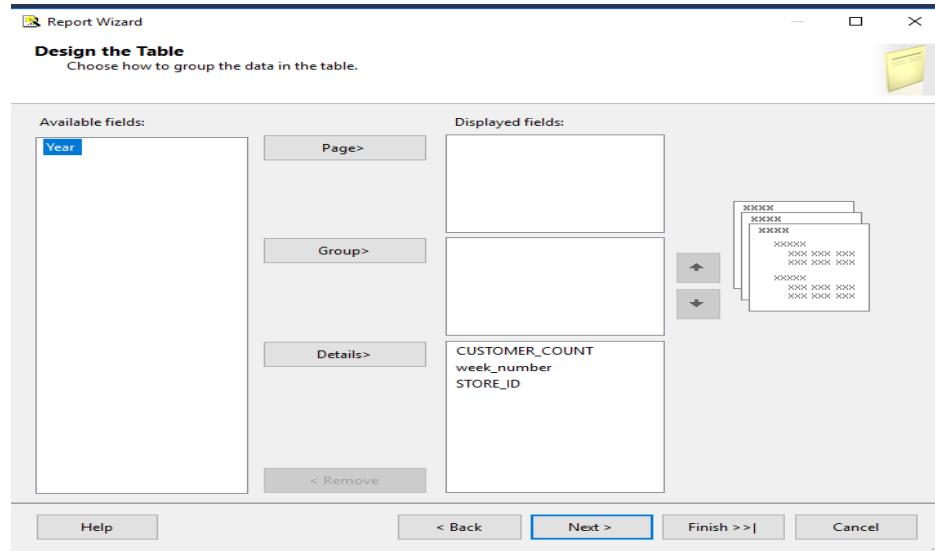
Step 2: Selecting required tables and fields



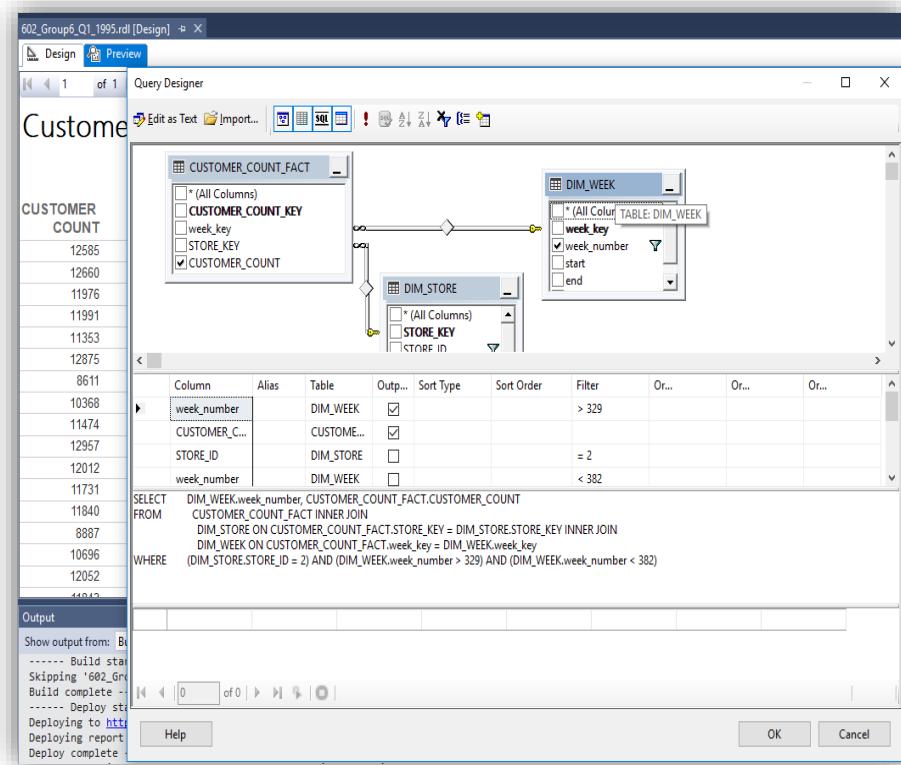
Step 3: Building query for year 1995



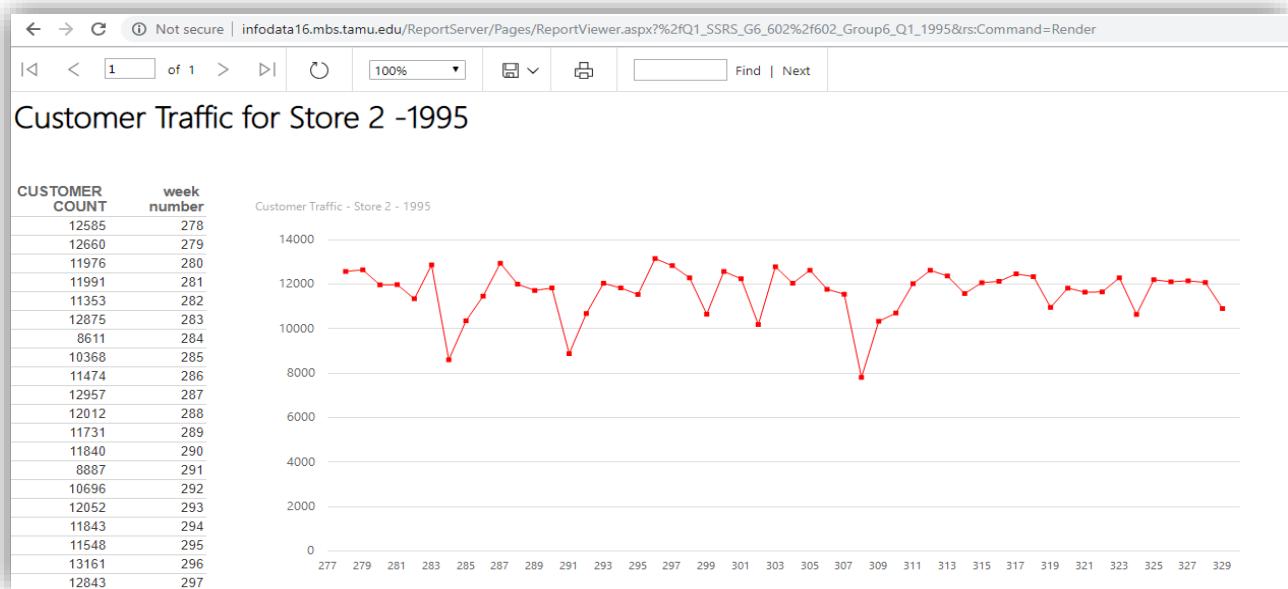
Step 4: Selecting rows and values



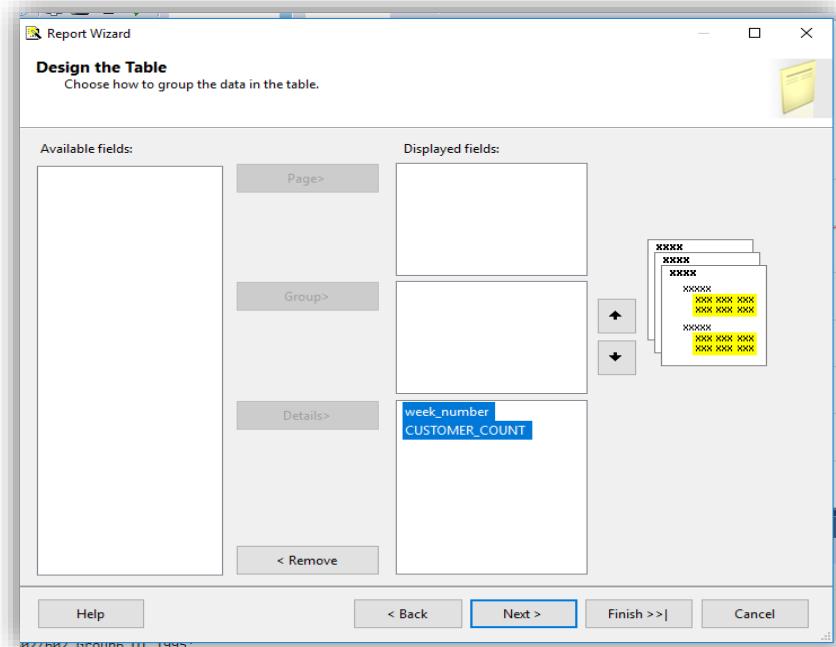
Step 5: Building query for 1995



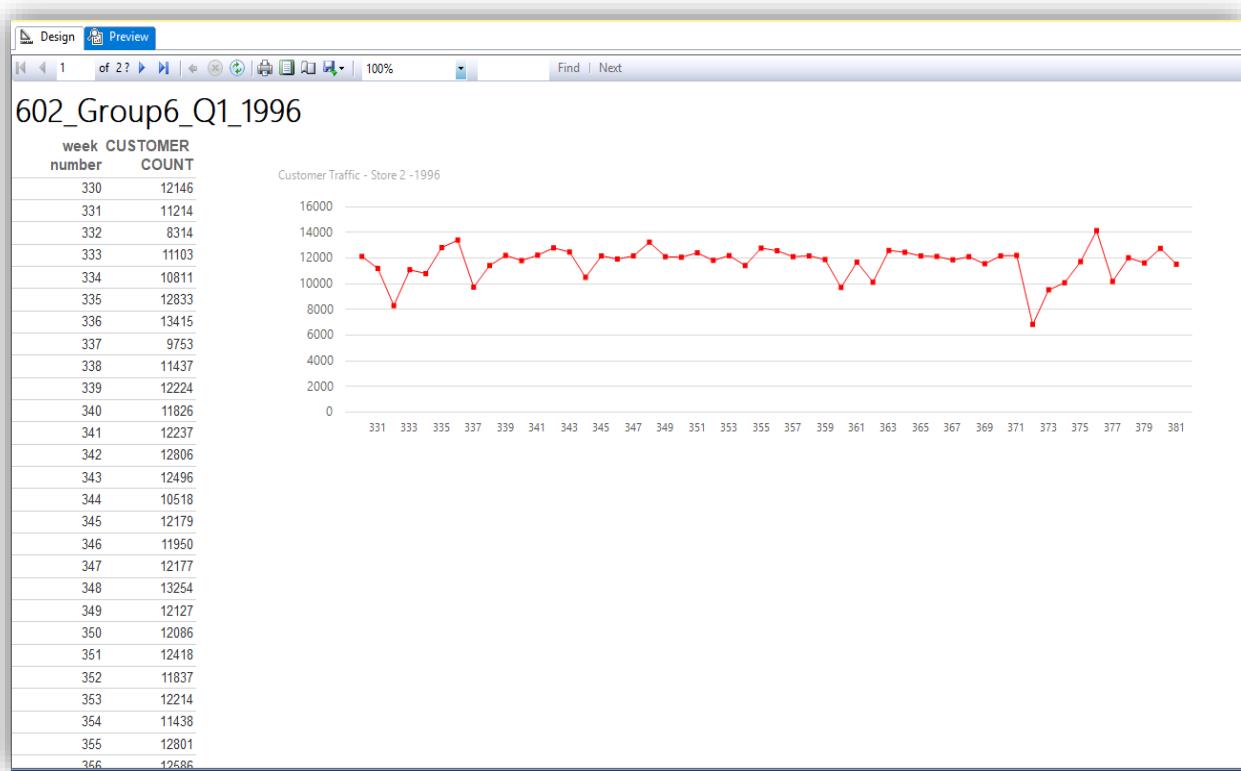
Step 6: Final report for 1995



Step 6: Selecting report columns and values

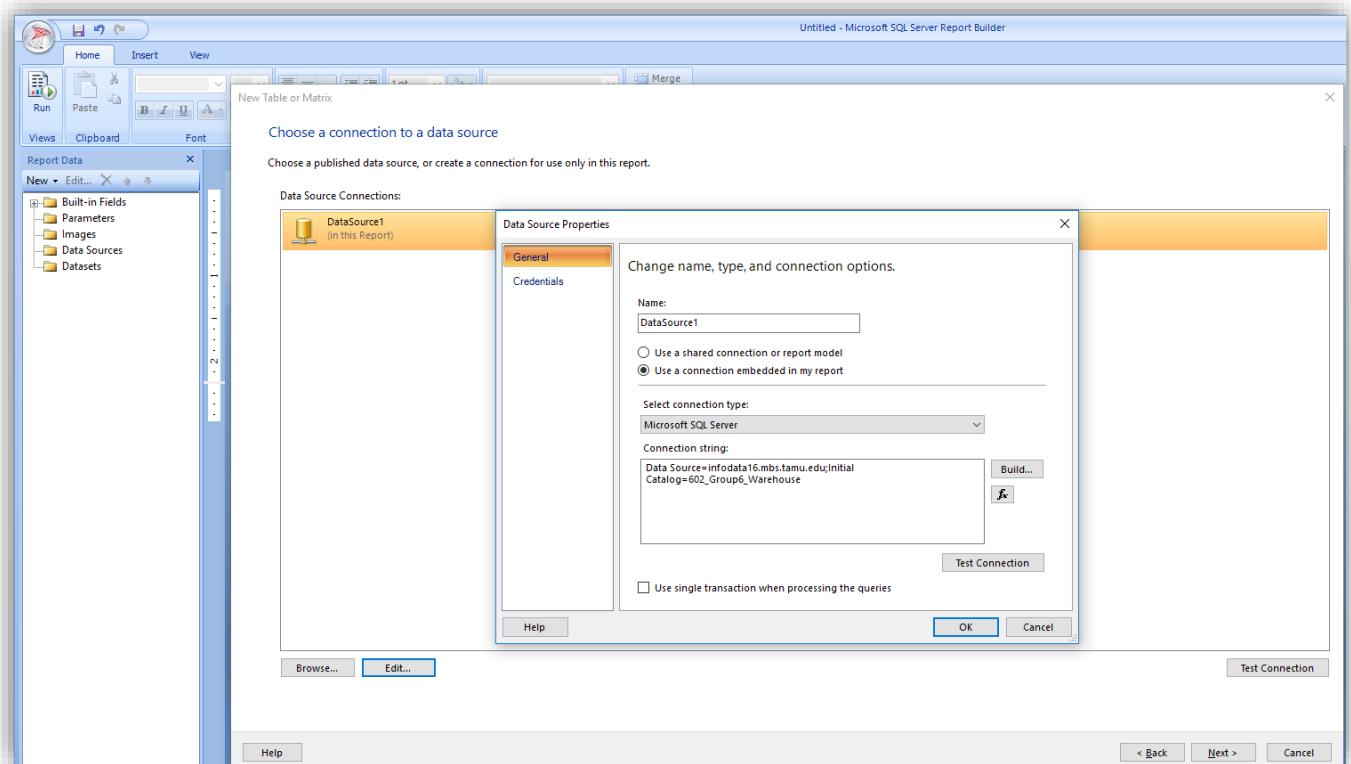


Step 7: Final Report for 1996



6.4 Report using Report Builder (Business question #2)

Step 1: Setting the data source



Step 2: Selecting tables and fields

New Table or Matrix

Design a query

Build a query to specify the data you want from the data source.

Database view

Tables

- COUPON_REDEMPTION_FACT
- CUSTOMER_COUNT_FACT
- DIM_PRODUCT
- DIM_STORE
- DIM_WEEK
- SALES_MOVE_FACT
 - SALES_MOVE_KEY
 - PRODUCT_KEY
 - week_key
 - STORE_KEY
 - MOVE
 - QTY
 - PRICE
 - PROFIT
 - CALCULATED_SALES_AMOUNT
 - CALCULATED_PROFIT
- sysdiagrams

Stored Procedures

Selected fields

Field	Aggregate
SALE	Grouped by
Avg_CALCULATED_PROFIT	Avg

Relationships

Left Table	Join Type	Right Table	Join Fields
DIM_PRODUCT	Inner	SALES_MOVE_FACT	PRODUCT_KEY = PRODUCT_KEY

Applied filters

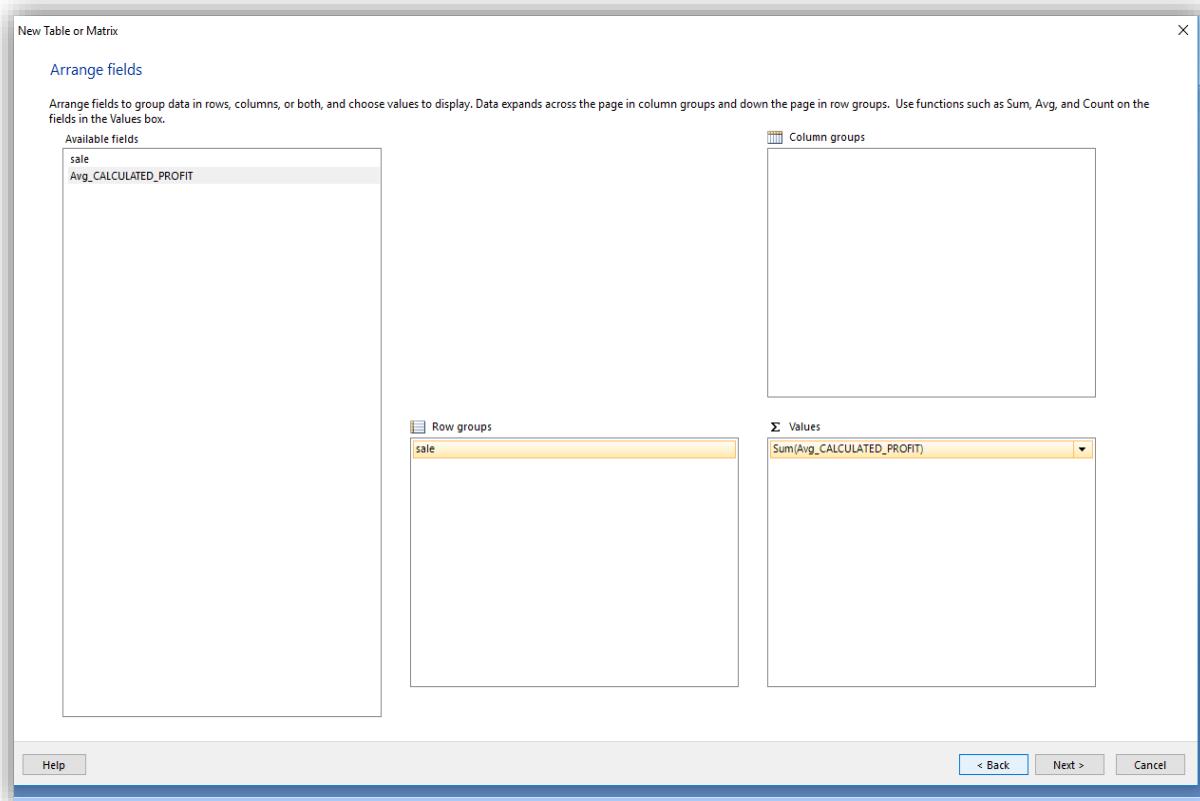
Field name	Operator	Value	Parameter
CATEGORY	like	Frozen%	<input type="checkbox"/>
SALE	is any of	S, B	<input type="checkbox"/>

Query results

Help < Back Next > Cancel

The screenshot shows the 'New Table or Matrix' dialog in Microsoft Power BI Data Explorer. In the 'Selected fields' section, 'SALE' is selected with 'Avg_CALCULATED_PROFIT' as its aggregate. A relationship is defined between 'DIM_PRODUCT' (left table) and 'SALES_MOVE_FACT' (right table) using an inner join on 'PRODUCT_KEY = PRODUCT_KEY'. Applied filters include 'CATEGORY like Frozen%' and 'SALE is any of S, B'.

Step 3: Selecting the rows, columns and Values



Step 4:

The screenshot shows a report preview window. At the top, it says 'Click to add title'. Below that is a table with two columns. The first column contains 'sale', '[sale]', and 'Total'. The second column contains 'Avg', '[Sum(Avg_CALCULATED_PROFIT)]', and '[Sum(Avg_CALCULATED_PROFIT)]'. In the bottom right corner, there is a placeholder box with the text '[&ExecutionTime]'. The entire window has a blue sidebar on the left.

Step 5: Running query for actual data

The screenshot shows the Microsoft SQL Server Report Builder interface. The top menu bar includes 'File', 'Edit', 'View', 'Insert', 'Format', 'Report Data', 'Report Parts', 'Data Regions', and 'Help'. The toolbar contains icons for Design, Views, Zoom, Refresh, Stop, Back, Print, Page Setup, and Print Layout. The main area displays a report titled 'sale' with the following data:

	Avg CALCULATED PROFIT
B	8.65394391982 356
S	11.6691213837 614
Total	20.3230653035 849

The status bar at the bottom right indicates the date and time: 4/20/2019 5:53:53 PM.

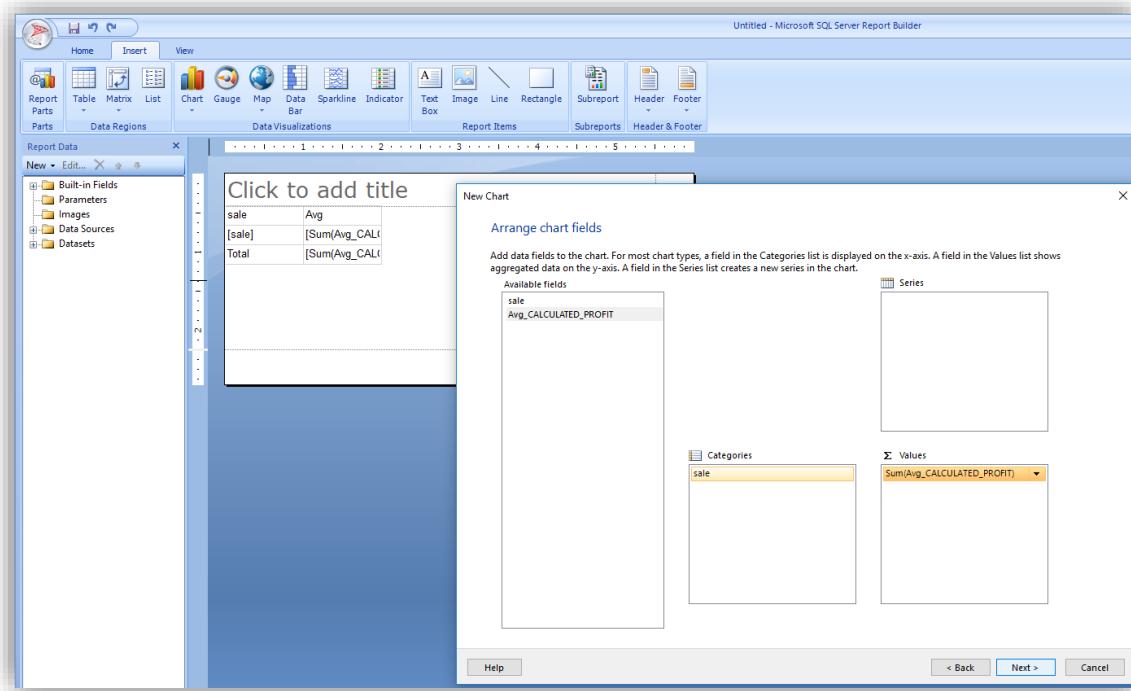
Step 6: Creating chart, selecting data source

The screenshot shows the Microsoft SQL Server Report Builder interface with the 'New Chart' dialog box open. The dialog box has the following steps:

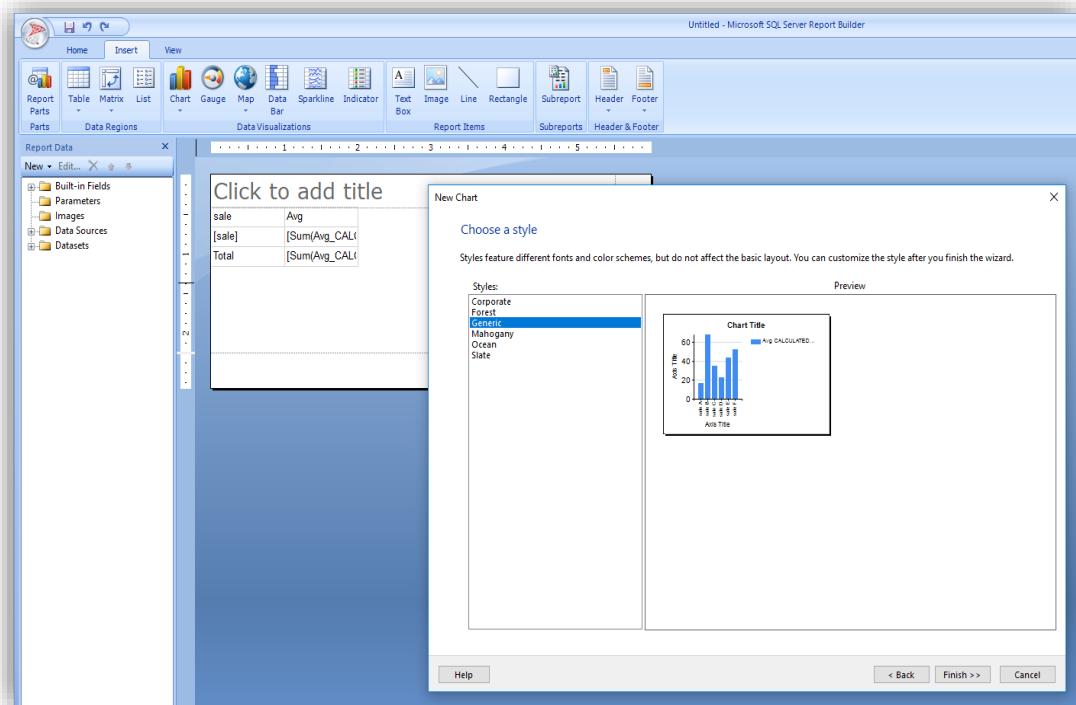
- Step 1: Choose a dataset
- Step 2: Choose a dataset
- Step 3: Choose an existing dataset in this report or a shared dataset
- Step 4: Create a dataset

The 'Choose an existing dataset in this report or a shared dataset' step is selected. A list box shows 'DataSet1 (in this Report) sale, Avg_CALCULATED_PROFIT' highlighted in orange. Other options include 'Browse...' and 'Create a dataset'. The background shows the report builder interface with a report data pane containing a table structure and a chart visualization pane.

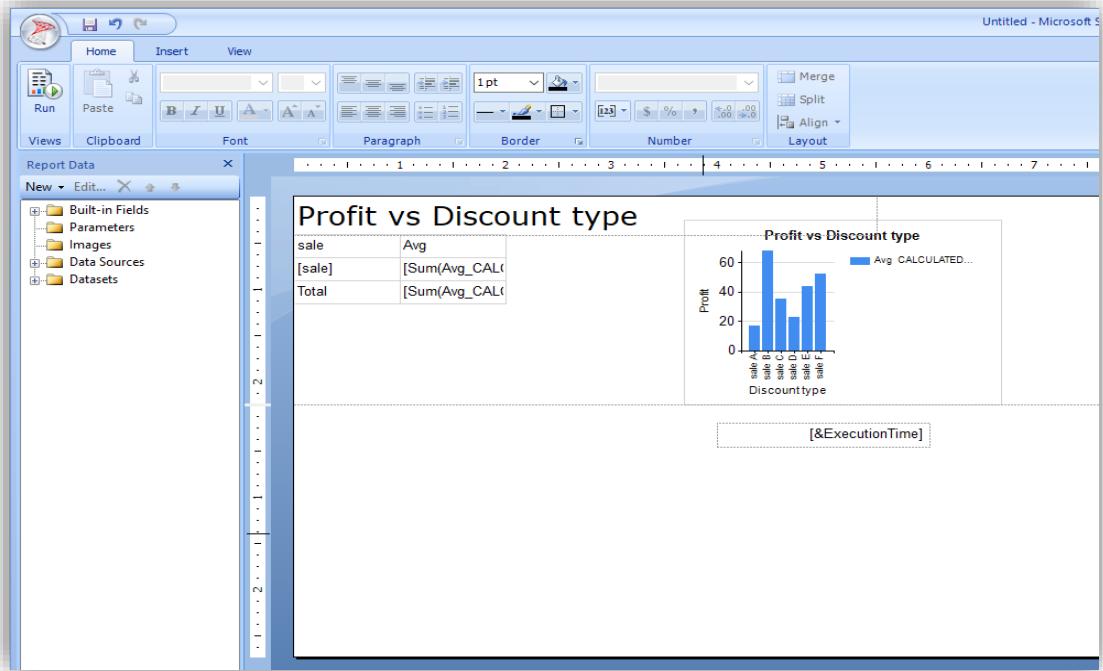
Step 7: Selecting rows, columns and values



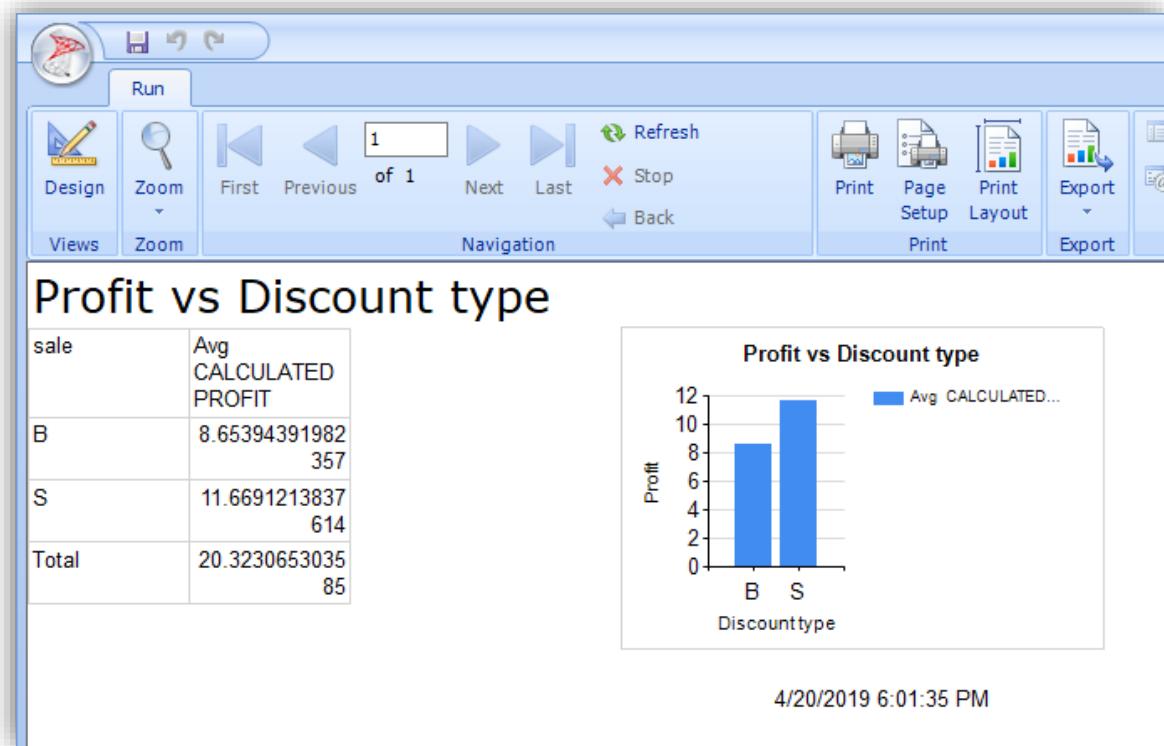
Step 8: Choosing the style



Step 9: Preview



Step 10: Final Chart



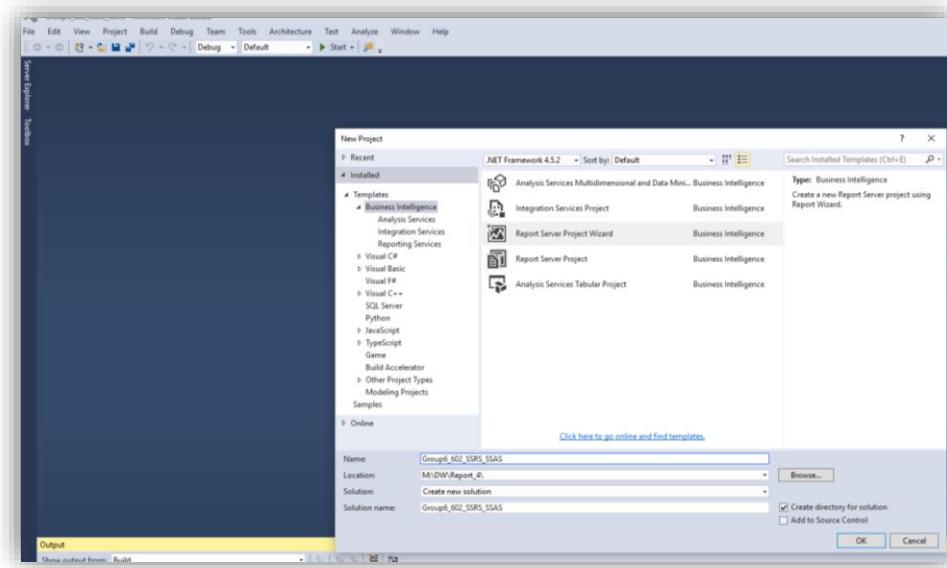
6.5 Report using SSAS and SSRS (Business question # 3)

Step 1: Creating Cube & deploying in SQL Server (6.1)

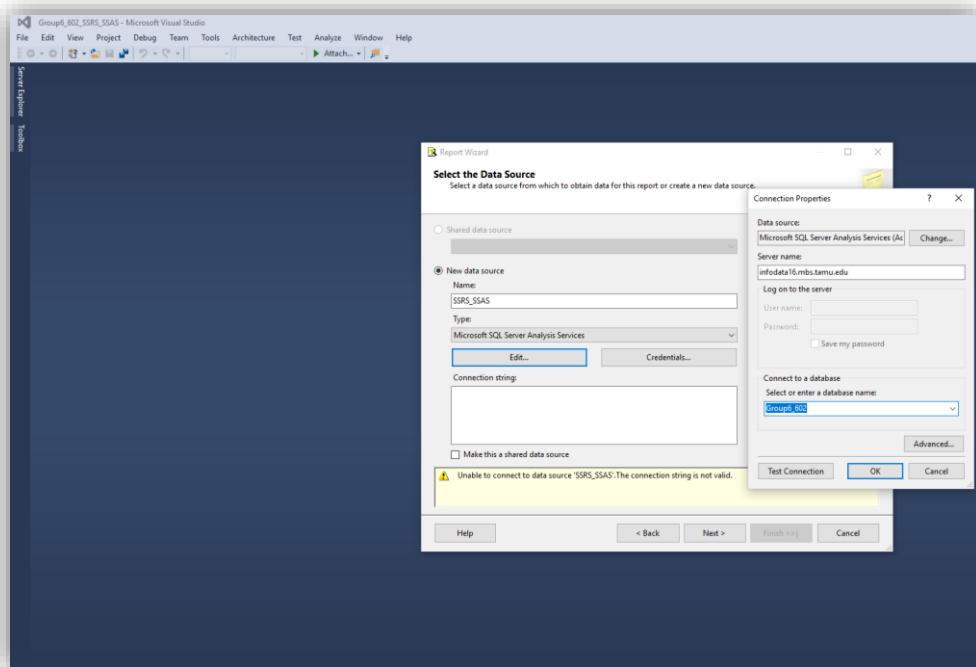
The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with various database objects like AdventureworksDW, AdventureworksLT, Analysis Services Tutorial, and several cubes and dimensions. The right pane shows the properties of the '602 Group6 Warehouse' cube, including its object name and viewer type (Cube). The central area shows the cube's structure and data.

UPC	PRICE TIER	CALCULATED SALES AMOUNT	CALCULATED PROFIT
107000039	High	9466.12	4202.04-49800284
107000039	Low	1642.55	748.1939780009
107000039	Medium	7843.5400000001	3385.95453995
107000039	None	10112.81	4290.47959997
107000039	High	12896.73	5635.04799977
107000039	Low	11000.0999999985	4695.01920002
107000039	Medium	27895.299999999	1072.35139998
107000039	None	2562.41	1111.16238002
107000018	High	311.333333333333	89.632666666667
107000018	Low	193.333333333333	55.660666666667
107000018	Medium	197	56.691031333333
107000018	CubFighter	474.666666666667	136.631266666667
107000018	High	979.666666666667	282.046013333333
107000018	Low	587.666666666667	169.1387
107000018	Medium	1541.666666666667	443.700333333333
107000018	None	97	27.9263
107000285	High	257.436666666667	128.541183333333
107000285	Low	126.286666666667	72.0068
107000285	Medium	249.5	123.257496
107000285	CubFighter	358.296666666667	182.149236666667
107000285	High	799.033333333333	416.411783
107000285	Low	538.42	308.779715
107000285	Medium	1420.023333333333	707.464541333333
107000285	None	20.55	5.55505
107000286	High	309.95	90.05347
107000286	Low	200.666666666667	57.771933333333
107000286	Medium	200.666666666667	57.771933333333
107000286	CubFighter	747.333333333333	216.742166666667
107000286	High	1365.556666666667	397.146183333333
107000286	Low	760.983333333333	220.064496666667
107000286	Medium	2053.403333333333	594.532866666666
107000286	None	98.866666666667	28.233333333333
107000185	High	1377.379	462.1549472
107000185	Low	473.191666666667	158.902666666667
107000185	Medium	1197.6	397.883025

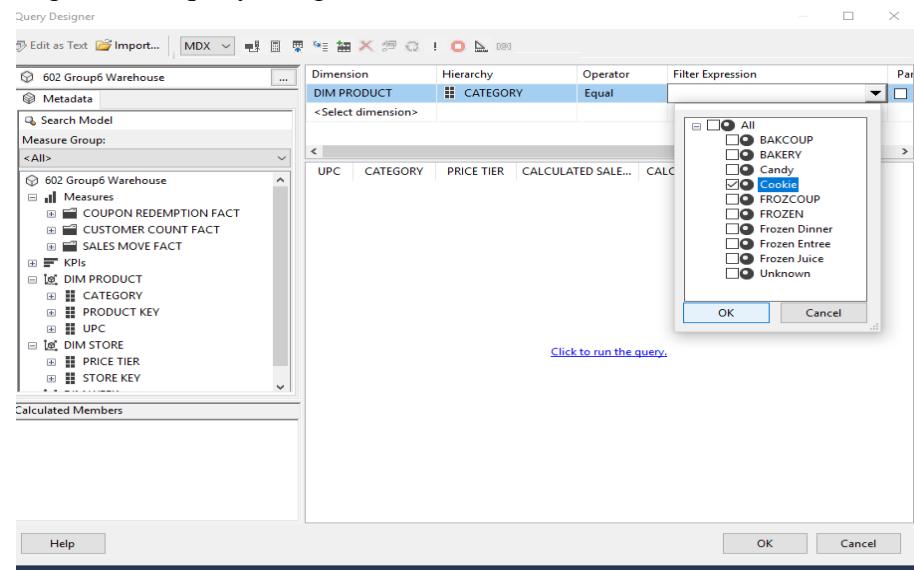
Step 2: Creating a reporting project



Step 3: Selecting data source

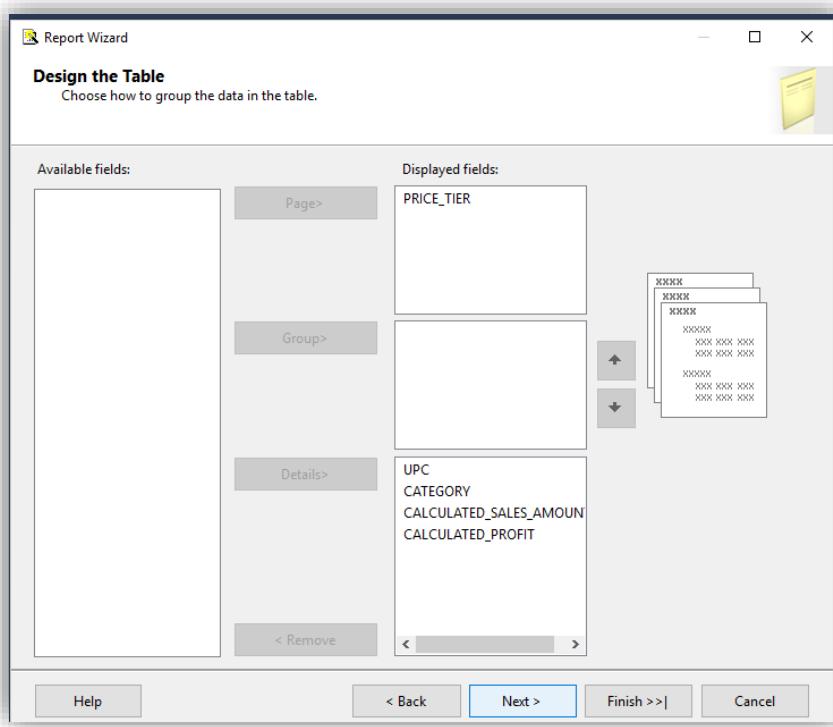


Step 4: Selecting data via query designer



UPC	CATEGORY	PRICE TIER	CALCULATED SALE...	CALCULATED PRO...
1410007012	Cookie	High	411.84	117.219996
1410007012	Cookie	Low	85.77	23.846148
1410007012	Cookie	Medium	465.03	124.339041
1410007012	Cookie	CubFighter	643.73	181.360112
1410007012	Cookie	High	1088.33	305.739860000001
1410007012	Cookie	Low	622.02	175.77657
1410007012	Cookie	Medium	1232.06	342.951386
1410007012	Cookie	None	167.94	46.652004
1410007016	Cookie	High	3777.05000000001	988.580671000001
1410007016	Cookie	Low	550.86	136.517207
1410007016	Cookie	Medium	4223.71000000001	1093.156017
1410007016	Cookie	CubFighter	3924.49000000001	1019.55080
1410007016	Cookie	High	11587.4	3030.31704500001
1410007016	Cookie	Low	3852.02000000001	991.921634
1410007016	Cookie	Medium	14013.30999999999	3644.38645700002
1410007016	Cookie	None	1435.11	378.417162
1410007056	Cookie	High	2081.64	481.811885
1410007056	Cookie	Low	171.67	42.707888

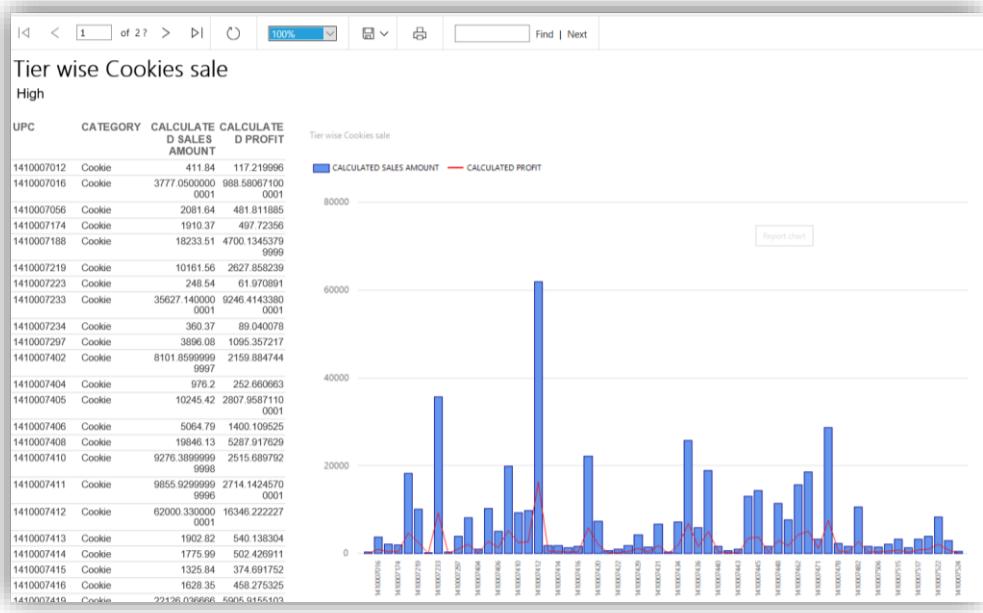
Step 5: Designing tables



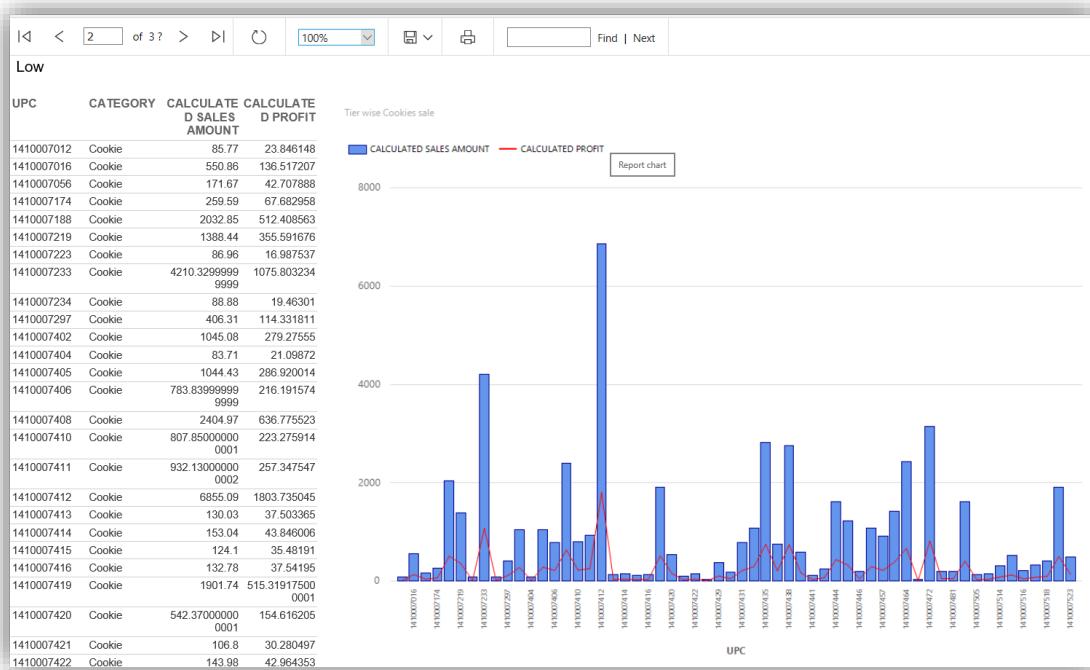
Step 6: Report (for low medium and high tier stores)

Low		Medium		High							
UPC	CATEGORY	CALCULATE D SALES AMOUNT	CALCULATE D PROFIT	UPC	CATEGORY	CALCULATE D SALES AMOUNT	CALCULATE D PROFIT	UPC	CATEGORY	CALCULATE D SALES AMOUNT	CALCULATE D PROFIT
1410007012	Cookie	85.77	23.865148	1410007012	Cookie	465.03	124.339041	1410007012	Cookie	411.84	117.219996
1410007016	Cookie	590.86	136.517207	1410007016	Cookie	4223.71000000	1093.156017	1410007016	Cookie	3777.05000000	988.580671000
1410007056	Cookie	171.67	42.707888	1410007056	Cookie	1865.26	454.509972	1410007056	Cookie	2081.64	481.811895
1410007114	Cookie	259.89	67.689583	1410007114	Cookie	1834.35000000	478.73470700	1410007114	Cookie	1910.37	497.72356
1410007168	Cookie	2032.85	512.409563	1410007168	Cookie	19042.98	4921.3495700	1410007168	Cookie	18233.51	4700.1345799
1410007191	Cookie	1300.86	365.591676	1410007191	Cookie	1001	001	1410007191	Cookie	1001	001
1410007223	Cookie	86.96	26.967537	1410007223	Cookie	8751.96099999	2211.73956300	1410007223	Cookie	10161.56	2627.66239
1410007233	Cookie	4210.32999999	1076.803234	1410007233	Cookie	406.31	114.331811	1410007233	Cookie	248.51	61.970991
1410007234	Cookie	88.88	19.46301	1410007234	Cookie	1045.08	279.27555	1410007234	Cookie	35627.14000000	9246.41432800
1410007297	Cookie	406.97	114.331811	1410007297	Cookie	34619.43000000	8950.057538	1410007297	Cookie	001	001
1410007402	Cookie	1045.08	279.27555	1410007402	Cookie	604.57999999	148.851328	1410007402	Cookie	360.37	89.040078
1410007404	Cookie	83.71	21.09872	1410007404	Cookie	3265.4	915.372664000	1410007404	Cookie	3896.08	1095.357217
1410007405	Cookie	1044.43	286.920014	1410007405	Cookie	7009.33999999	1874.33707	1410007405	Cookie	8101.85999999	2159.884744
1410007406	Cookie	783.83999999	216.191574	1410007406	Cookie	999	001	1410007406	Cookie	976.2	252.660663
1410007408	Cookie	2404.97	636.775523	1410007408	Cookie	927.09000000	238.334596	1410007408	Cookie	10245.42	2807.9571100
1410007410	Cookie	807.85000000	223.275914	1410007410	Cookie	5379.92999999	1469.632675	1410007410	Cookie	5064.79	1400.199525
1410007411	Cookie	932.13000000	257.347547	1410007411	Cookie	3892.17999999	1076.976205	1410007411	Cookie	19846.13	5287.917629
1410007412	Cookie	6855.09	1803.735045	1410007412	Cookie	999	001	1410007412	Cookie	9276.38999999	2515.689792
1410007413	Cookie	130.03	37.503365	1410007413	Cookie	153.04	43.846006	1410007413	Cookie	9855.92999999	2714.14245700
1410007415	Cookie	124.1	35.48191	1410007415	Cookie	132.78	37.54195	1410007415	Cookie	999	001
1410007419	Cookie	1901.74	515.319175000	1410007419	Cookie	4905.25999999	1339.9247	1410007419	Cookie	62000.33000000	16346.222227
1410007420	Cookie	542.37000000	154.516205	1410007420	Cookie	6560.19999999	1805.13230400	1410007420	Cookie	1902.82	540.138304
1410007421	Cookie	106.8	30.280497	1410007421	Cookie	994	001	1410007421	Cookie	1775.99	502.426911
1410007422	Cookie	143.98	42.964353	1410007422	Cookie	1000.56	372.244322	1410007422	Cookie	1325.84	374.691752
1410007424	Cookie	34.42	9.169746	1410007424	Cookie	1167.92	331.689589	1410007424	Cookie	1628.35	458.275325
1410007425	Cookie	383.31999999	108.359156	1410007425	Cookie	960.47600000	271.213421	1410007425	Cookie	22126.03666666	5905.91551033
1410007430	Cookie	184.11	50.119032	1410007430	Cookie	1050.45	306.611307	1410007430	Cookie	7300.12999999	2030.145346

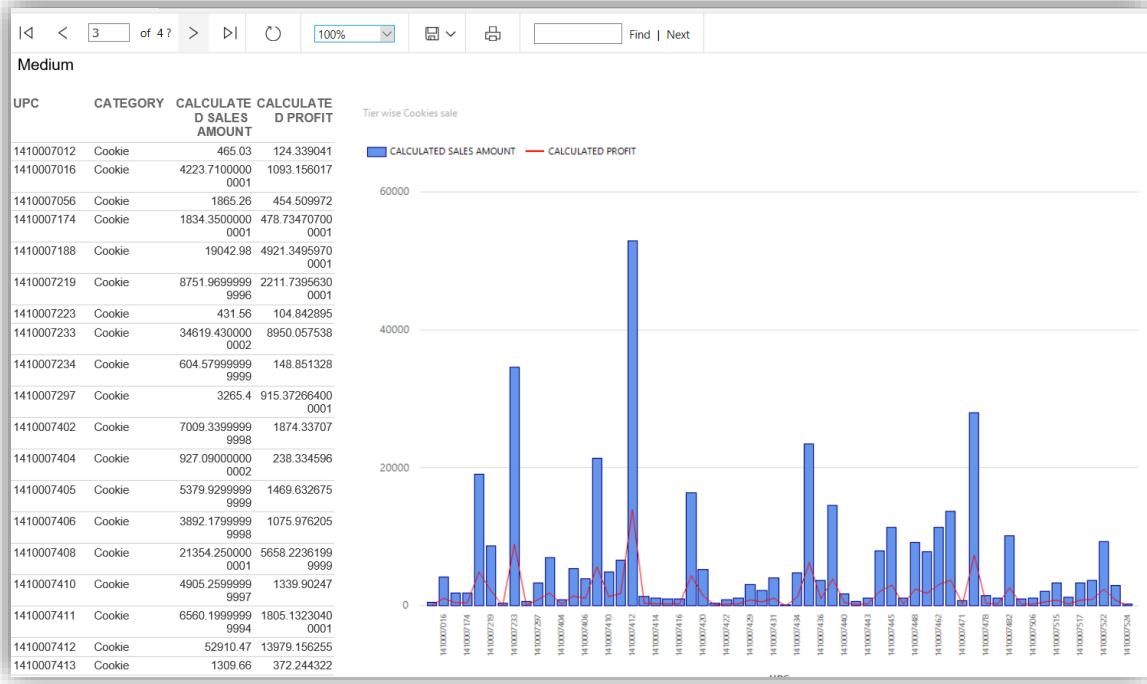
Step 7: Graph for High Tier



Step 8: Generating graph for low tier



Step 9: Generating report for Medium Tier



6.6 Report using SSAS (Cube + Excel) (Business Question #4)

Step 1: Generating cube (6.1)

The screenshot shows the SSAS cube editor for the "602 Group6 Warehouse" cube. The left pane displays the cube structure with dimensions (DIM PRODUCT, DIM WEEK), measures (CALCULATED SALES AMOUNT, CALCULATED PROFIT), and KPIs. The middle pane shows a data grid with data for the "Candy" category across various years and weeks. The right pane shows the Solution Explorer with the project structure.

CATEGORY	Year	Special Events	Week Key	CALCULATED SALES AMOUNT	CALCULATED PROFIT
Candy	1989	Halloween	7	6999.22	3376.322796
Candy	1990	Halloween	59	6651.26333333333	2535.38211666667
Candy	1991	Halloween	112	591.45	2404.6478
Candy	1992	Halloween	164	5754.04	2654.709147
Candy	1993	Halloween	216	7104.81	3005.810681
Candy	1994	Halloween	268	5910.68	2790.009179
Candy	1995	Halloween	320	4712.39	2063.50918
Candy	1996	Halloween	372	4671.76	2018.529682

Step 2: Creating pivot

Book5 - Excel

PivotTable Name: Active Field

PivotTable Fields

Choose fields to add to report:

- SALES MOVE FACT
- CALCULATED PROFIT
- CALCULATED SALES AMOUNT

Fields

- DIM PRODUCT
- CATEGORY
- PRODUCT KEY
- DIM STORE
- STORE KEY
- DIM WEEK
- Special Events
- Week Key
- Year

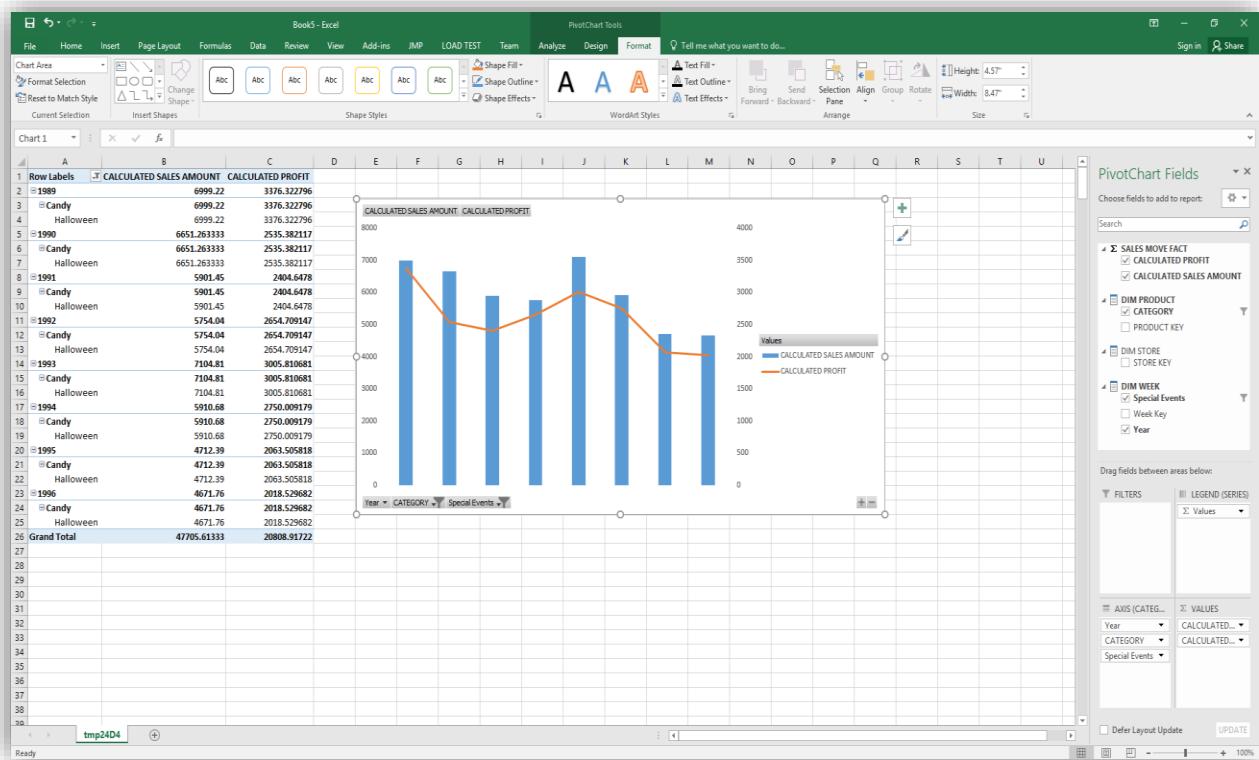
Drag fields between areas below:

FILTERS COLUMNS

ROWS VALUES

VALUES

	Calculated Sales Amount	Calculated Profit
1 Row Labels	CALCULATED PROFIT	CALCULATED SALES AMOUNT
2 1989	3376.322796	6999.22
3 Candy	3376.322796	6999.22
4 Halloween	3376.322796	6999.22
5 1990	2535.382117	6651.263333
6 Candy	2535.382117	6651.263333
7 Halloween	2535.382117	6651.263333
8 1991	2404.6478	5901.45
9 Candy	2404.6478	5901.45
10 Halloween	2404.6478	5901.45
11 1992	2654.709147	5754.04
12 Candy	2654.709147	5754.04
13 Halloween	2654.709147	5754.04
14 1993	3005.810681	7104.81
15 Candy	3005.810681	7104.81
16 Halloween	3005.810681	7104.81
17 1994	2750.009179	5910.68
18 Candy	2750.009179	5910.68
19 Halloween	2750.009179	5910.68
20 1995	2063.505818	4712.39
21 Candy	2063.505818	4712.39
22 Halloween	2063.505818	4712.39
23 1996	2018.529682	4671.76
24 Candy	2018.529682	4671.76
25 Halloween	2018.529682	4671.76
26 Grand Total	47705.61333	20808.91722



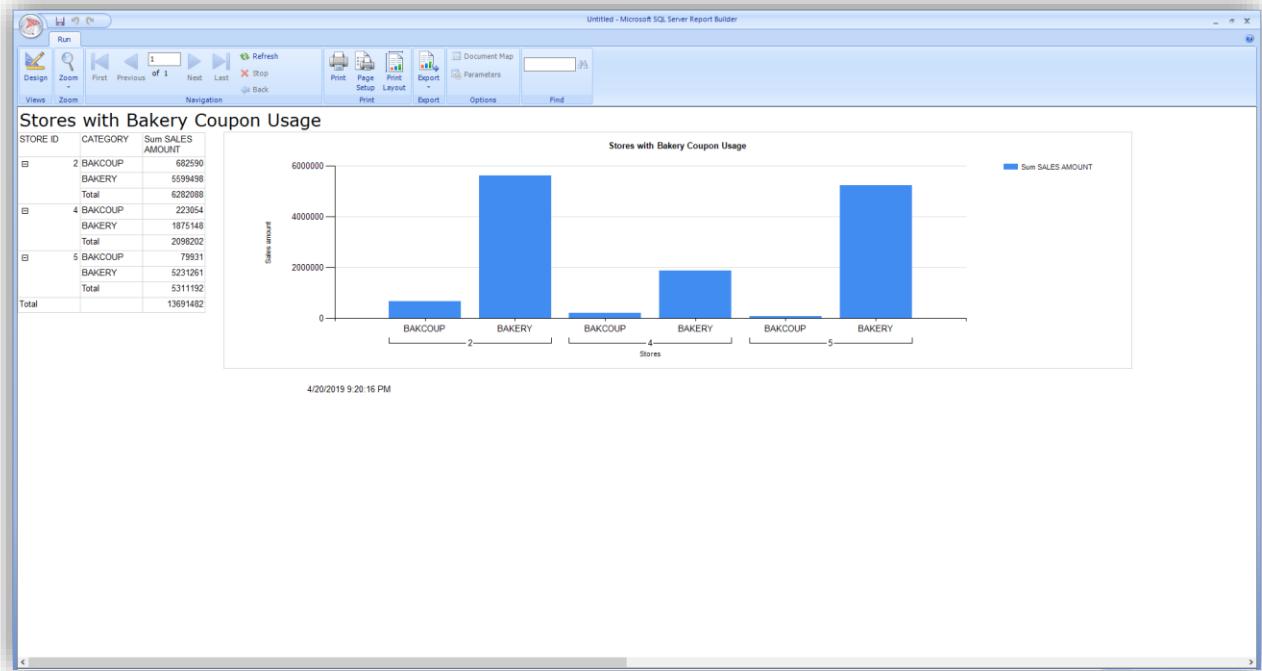
6.5 Report using Report Builder (Business question #5)

Step 1: Filtering required categories

The screenshot shows the Microsoft SQL Server Report Builder interface. The main window title is "Untitled - Microsoft SQL Server Report Builder". The left sidebar includes "Home", "Insert", "View", "Views", "Clipboard", "Font", "Paragraph", "Report Data", "New", and "Edit". The "Report Data" section lists "Built-in Fields", "Parameters", "Images", "Data Sources", and "Datasets". The central area is titled "Design a query" with the sub-instruction "Build a query to specify the data you want from the data source." A "Database view" pane on the left lists tables such as COUPON_REDEMPTION_FACT, DIM_PRODUCT, and DIM_STORE. The "Selected fields" pane on the right shows a query definition:
Field: STORE_ID
Aggregate: Sum
Grouped by: Sum_SALES_AMOUNT, CATEGORY
Relationships pane shows joins between COUPON_REDEMPTION_FACT and DIM_PRODUCT, and between COUPON_REDEMPTION_FACT and DIM_STORE.
Applied filters pane shows two conditions: CATEGORY is any of BAKCOUP, BAKERY and SALES_AMOUNT is not 0.
A "Query results" pane at the bottom displays the following data:

STORE_ID	Sum_SALES_AM...	CATEGORY
2	682590	BAKCOUP
4	223054	BAKCOUP
5	79931	BAKCOUP
2	5599498	BAKERY
4	1875148	BAKERY
5	5231261	BAKERY

Step 2: Report



7. References:

- [1] <https://www.forbes.com/sites/lauraheller/2014/01/31/this-is-what-a-failed-supermarket-looks-like/#4677ddb54866>
- [2] <https://en.wikipedia.org/wiki/Retail>
- [3] <https://www.marketingdonut.co.uk/customer-care/the-five-principles-of-retail>
- [4] <https://rsmus.com/what-we-do/industries/consumer-products/retail/top-trends-and-issues-for-retail-in-2019.html>
- [5] <https://www.inc.com/guides/2010/06/defining-your-target-market.html>
- [6] <https://extension.psu.edu/understanding-your-customers-how-demographics-and-psychographics-can-help>
- [7] <https://www.forbes.com/sites/rogerdooley/2014/01/28/h-e-b/#48e78c8632b8>
- [8] <https://pdfs.semanticscholar.org/ed03/e6a1f4fb6fe0b185938e45ef69dc64b4bb69.pdf>