[**Exercise 1:** 2](#_Toc125742915)

[About Dataset 2](#_Toc125742916)

[Things to be Done: 2](#_Toc125742917)

[1) Load the CSV files to the database tables using python. 2](#_Toc125742918)

[2) In SQL database 4](#_Toc125742919)

[a) Add is\_invalid\_date column which indicates True or False based on the condition where start\_day is greater than end\_day. 4](#_Toc125742920)

[c) Normalize campaign\_table into 2 tables. 5](#_Toc125742921)

[d) Combine coupon and coupon\_redempt tables into one Coupon table. 6](#_Toc125742922)

[3) Find/ Create all the primary keys and foreign keys. 6](#_Toc125742923)

[4) Build an ER model diagram using Lucid chart. 8](#_Toc125742924)

[5) Connect to Power BI and build sample visualization. 9](#_Toc125742925)

[**Exercise 2 :** 10](#_Toc125742926)

[API Integration 10](#_Toc125742927)

[Power Bi Visual of Weather Report: 12](#_Toc125742928)

**Final Internship Training Project**

(Documentation)

# **Exercise 1:**

## About Dataset

This dataset contains household level transactions over two years from a group of 2,500 households who are frequent shoppers at a retailer. It contains all of each household’s purchases, not just those from a limited number of categories. For certain households, demographic information as well as direct marketing contact history are included.

## Things to be Done:

## Load the CSV files to the database tables using python.

For this first we must import some modules to read our data in python.

For the python code I have used Jupyter notebook.

import pandas as pd

import sqlalchemy

SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that givers application developers

The full power and flexibility of SQL

* By doing read.csv is pandas we can read our file

data1 = pd.read\_csv('campaign\_table.csv')

data2 = pd.read\_csv('causal\_data.csv')

data3 = pd.read\_csv('coupon.csv')

data4 = pd.read\_csv('coupon\_redempt.csv')

data5 = pd.read\_csv('hh\_demographic.csv')

data6 = pd.read\_csv('product.csv')

data7 = pd.read\_csv('transaction\_data.csv')

* This code used for building Python connection to load data in SQL server.

So here we required Driver name, Server Name, Database Name, and the connection URL of SQL server which we get on SQL server documentation

connection\_string = ('Driver=ODBC Driver 17 for SQL Server;'

'Server=SWADKA01PC0922\SQLEXPRESS;'

'Database=Final\_Project;'

'Trusted\_Connection=yes;')

connection\_url = sqlalchemy.engine.URL.create("mssql+pyodbc",query=dict(odbc\_connect=connection\_string))

engine = sqlalchemy.create\_engine(connection\_url, fast\_executemany=True)

* Here I have used to\_sql to load data directly to the SQL server.
* In this below written code if\_exists determined as ‘replace’, which means if following uploading data is already present in selected database it will just replace it with new data.
* And index is set to ‘false’, because while uploading the data to SQL server it was also pushing the indexes, which are non-relevant for our use case.

data1.to\_sql("campaign\_table", engine, if\_exists='replace',index = False)

data3.to\_sql("coupon", engine, if\_exists='replace',index = False)

data4.to\_sql("coupon\_redempt", engine, if\_exists='replace',index = False)

data5.to\_sql("hh\_demographic", engine, if\_exists='replace',index = False)

data6.to\_sql("product", engine, if\_exists='replace',index = False)

data7.to\_sql("transaction\_data", engine, if\_exists='replace',index = False

data3[['COUPON\_UPC', 'PRODUCT\_ID', 'CAMPAIGN']].duplicated().value\_counts()

* In below code the file size was too large, its giving error to upload in one go. So to resolve that we passed that data in chunks.

data2.to\_sql("causal\_data", engine, if\_exists='replace', index = False,chunksize=10000)

## In SQL database

* To view whole data, I used the select \* query.

SELECT \*

FROM [Final\_Project].[dbo].[campaign\_table]

go

* Here we need to change the data type of START\_DAY and END\_DAY columns from string to date format.
* For that I used alter query and used ‘date’ datatype.

-- change into date format

alter table campaign\_table

alter column START\_DAY date

alter table campaign\_table

alter column END\_DAY date

go

### Add is\_invalid\_date column which indicates True or False based on the condition where start\_day is greater than end\_day.

* Created new column named as is\_invalid\_date.

ALTER TABLE campaign\_table add is\_invalid\_date nvarchar(10);

go

* Used update query using case statement to load ‘True’ or ‘False’ values as per requirement.

Update campaign\_table

SET is\_invalid\_date =(

CASE WHEN START\_DAY > END\_DAY THEN

'True'

ELSE

'False'

END

)

Go

1. Create start\_datekey and end\_datekey with yyyymmdd format based on start\_day and end\_day.

* For this I used alter query to add new column in existing table.
* Then I used update query to update the values in new column.
* Here convert function is used to change the format of that value. Value 112 is the format to change the existing value into yyyymmdd format.

alter table campaign\_table add START\_DAY\_KEY varchar(max)

update campaign\_table set

START\_DAY\_KEY = convert(varchar, START\_DAY, 112) from campaign\_table

--

alter table campaign\_table add END\_DAY\_KEY varchar(max)

update campaign\_table set

END\_DAY\_KEY = convert(varchar, END\_DAY, 112) from campaign\_table

### Normalize campaign\_table into 2 tables.

* created 2 separate tables from campaign\_table named as Description\_no & Campaign
* Created 1st table names as Description\_no.
* For that I have created a view as V\_description\_no.

drop view if exists V\_description\_no

go

create View V\_description\_no as

select Description from campaign\_table group by DESCRIPTION having count(\*) > 1;

go

* Here I created Description\_no table.
* Here I have only inserted Description column. And gave number as 1, 2, and 3 to TypeA, TypeB and TypeC in new created column as DescCamp\_id.

drop table if exists Description\_no

create table Description\_no(DescCamp\_id bigint primary key, DESCRIPTION nvarchar(20))

go

insert into Description\_no(DescCamp\_id, DESCRIPTION)

(select case when DESCRIPTION = 'TypeA' then '1'

when DESCRIPTION = 'TypeB' then '2'

when DESCRIPTION = 'TypeC' then '3' else null end

DescCamp\_id , DESCRIPTION from V\_description\_no)

go

select \* from Description\_no

--

* Created 2nd table names as Campaign.
* In this table except the Description columns all other columns are taken.
* Follows the same step as above for the DescCamp\_id Column.

drop table if exists campaign

Create table Campaign(household\_key bigint,CAMPAIGN bigint,START\_DAY date,END\_DAY date,is\_invalid\_date nvarchar(10), START\_DAY\_KEY varchar(max), END\_DAY\_KEY varchar(max), DescCamp\_id bigint FOREIGN KEY REFERENCES Description\_no(DescCamp\_id))

go

insert into Campaign(household\_key, CAMPAIGN, START\_DAY, END\_DAY, is\_invalid\_date, START\_DAY\_KEY, END\_DAY\_KEY, DescCamp\_id)

(select household\_key,CAMPAIGN,START\_DAY,END\_DAY, is\_invalid\_date, START\_DAY\_KEY, END\_DAY\_KEY,

case when DESCRIPTION = 'TypeA' then '1'

when DESCRIPTION = 'TypeB' then '2'

when DESCRIPTION = 'TypeC' then '3' else null end

DescCamp\_id from campaign\_table)

go

select \* from Campaign

select \* from Description\_no

go

### Combine coupon and coupon\_redempt tables into one Coupon table.

with cte as (SELECT

c1.CAMPAIGN, c1.COUPON\_UPC, c1.PRODUCT\_ID,c2.household\_key,c2.DAY

FROM coupon c1

full JOIN coupon\_redempt c2

ON c1.CAMPAIGN = c2.CAMPAIGN

and c1.COUPON\_UPC = c2.COUPON\_UPC )

select \* into Coupon\_Table from cte

## Find/ Create all the primary keys and foreign keys.

* Need to do alter query to change the datatype to bigint and make column as not null to make column as Primary key or Foreign key.
* To find the Keys in tables use these following pandas code, this will give results as True for duplicate, and False for Unique records.

data4[['household\_key', 'DAY', 'COUPON\_UPC', 'CAMPAIGN']].duplicated().value\_counts()

-- hh\_demographic

--

ALTER TABLE [hh\_demographic] alter column household\_key bigint NOT NULL

go

Alter table [dbo].[hh\_demographic] add primary key (household\_key);

--product

--

ALTER TABLE [dbo].[product] alter column PRODUCT\_ID bigint NOT NULL

go

Alter table [dbo].[product] add primary key (PRODUCT\_ID);

--

--Transaction data

ALTER TABLE [dbo].[transaction\_data] alter column PRODUCT\_ID bigint NOT NULL

go

ALTER TABLE [dbo].[transaction\_data] alter column [BASKET\_ID] bigint NOT NULL

go

Alter table [dbo].[transaction\_data] add primary key (PRODUCT\_ID, BASKET\_ID);

--

--campaign

-- handled orphan keys campaign

* There were some household keys which are present in the campaign table but not in the hh\_demographic, these keys are called orphan keys.
* To handle this orphan keys we need to insert those keys in to hh\_demographic table

insert into hh\_demographic (household\_key)

(select distinct household\_key from campaign where household\_key not in (select household\_key from hh\_demographic))

Alter table [dbo].[Campaign] add foreign key ([household\_key]) references [hh\_demographic] ([household\_key])

--coupon redempt

Alter table [dbo].[coupon\_redempt] add foreign key ([household\_key]) references [hh\_demographic] ([household\_key])

--coupon table

Alter table [dbo].[Coupon\_Table] add foreign key ([household\_key]) references [hh\_demographic] ([household\_key])

Alter table [dbo].[Coupon\_Table] add foreign key (PRODUCT\_ID) references [product] (PRODUCT\_ID)

--coupon

Alter table [dbo].[coupon] add foreign key (PRODUCT\_ID) references [product] (PRODUCT\_ID)

--casual table

Alter table [dbo].[causal\_data] add foreign key (PRODUCT\_ID) references [product] (PRODUCT\_ID)

-- handled orphan keys [transaction\_data]

* Same case used in this table also as campaign table.

insert into hh\_demographic (household\_key)

(select distinct household\_key from [transaction\_data] where household\_key not in (select household\_key from hh\_demographic))

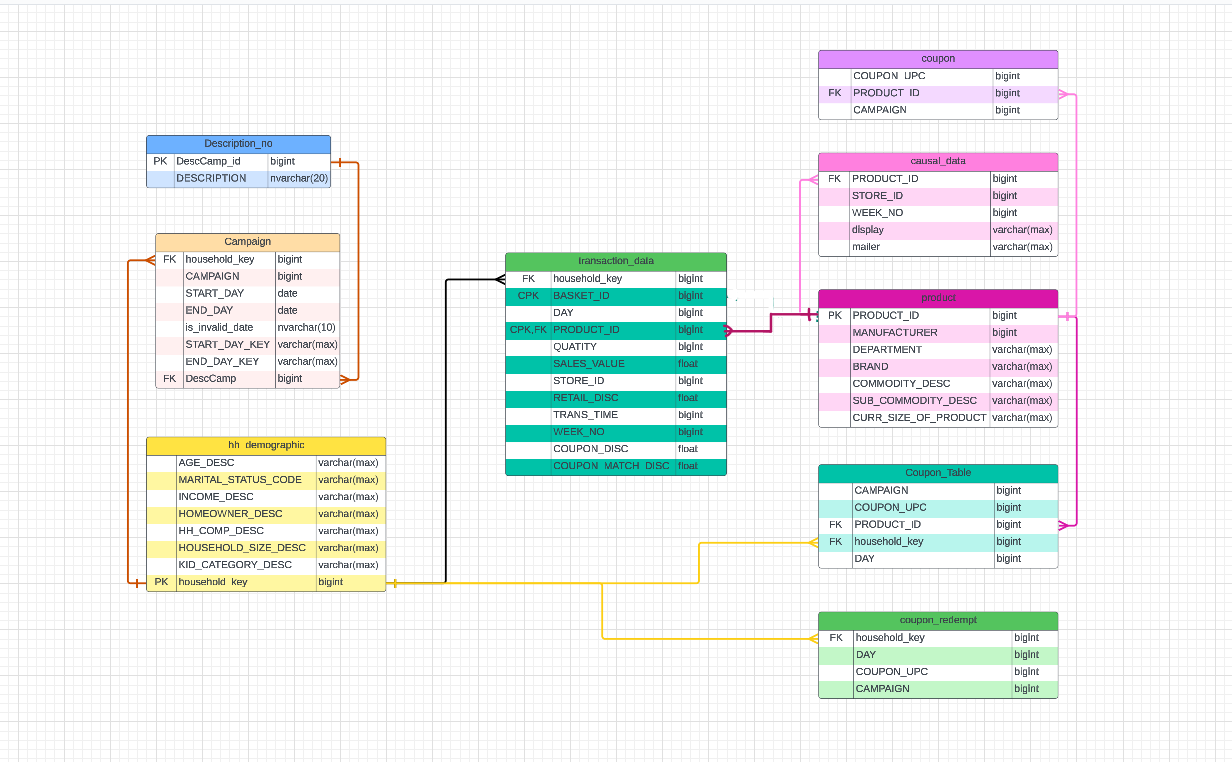
-- transaction

Alter table [dbo].[transaction\_data] add foreign key ([household\_key]) references [hh\_demographic] ([household\_key])

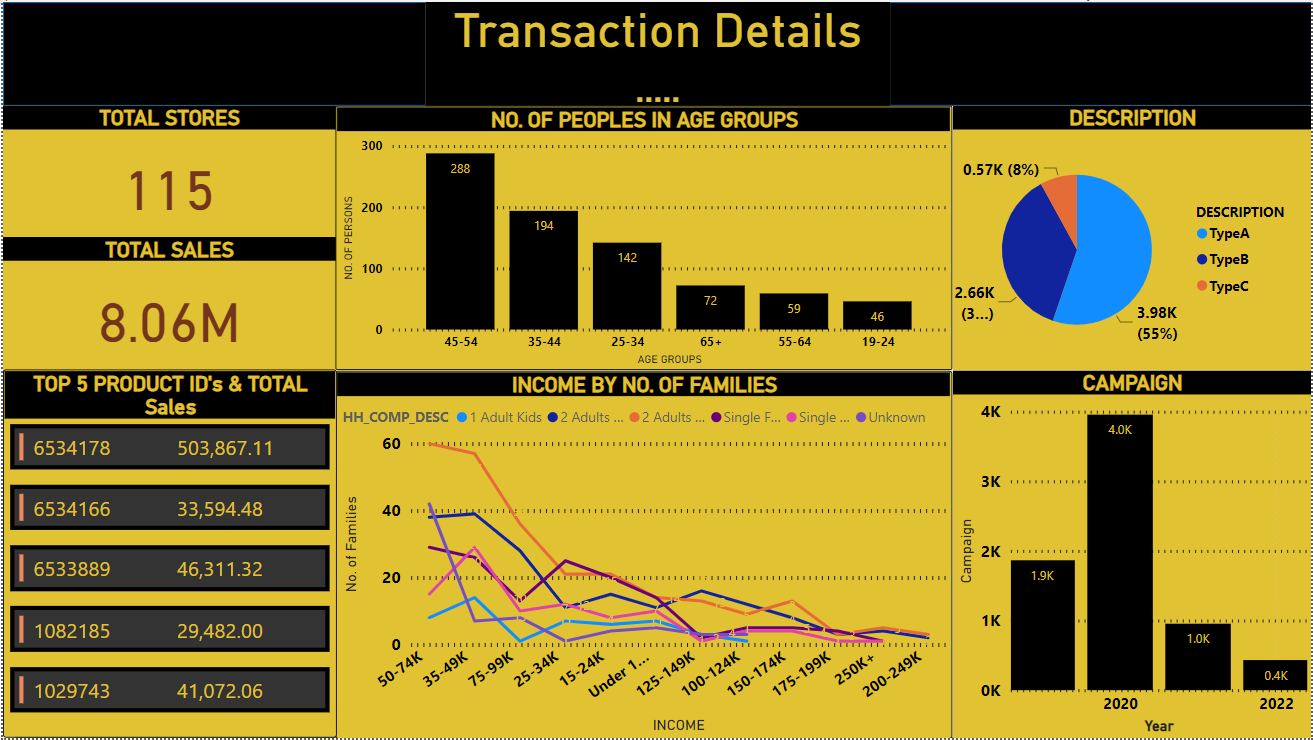
Alter table [dbo].[transaction\_data] add foreign key (PRODUCT\_ID) references [product] (PRODUCT\_ID)

select count(a.household\_key) from campaign\_table a join coupon b on a.CAMPAIGN=b.CAMPAIGN

## Build an ER model diagram using Lucid chart.



## Connect to Power BI and build sample visualization.



# **Exercise 2 :**

## API Integration

* API Integration is one of the important things to be managed when working with real time datasets. In this exercise you will be sourcing data from an api and displaying the results in the Power BI report.
* From here we can source the data from the api : https://openweathermap. org/current.
* As per requirement we can get the data from above web site like, current or week/month.
* For our requirement I got data from this site <https://api.openweathermap.org/data/2.5/forecast>?
* This will give us 5 days weather report
* For the first need to import some Modules.
* I did the exception handling for getting the data from above site.
* In that I have given the specific city name Bangalore.

import requests

from requests.exceptions import HTTPError

try:

api\_key = "5ed8bb81871d32daaa4e4daa19a61235"

City\_Name = 'Bangalore'

res = f"https://api.openweathermap.org/data/2.5/forecast?q={City\_Name}&appid={api\_key}"

response = requests.get(res)

response.raise\_for\_status()

# access json content

data1 = response.json()

print("Entire JSON response")

# print(jsonResponse)

except HTTPError as http\_err:

print(f'HTTP error occurred: {http\_err}')

except Exception as err:

print(f'Other error occurred: {err}')

# Created a dataframe of the result which we get from site

df\_weather=pd.DataFrame(columns=['city\_id','city\_name','date','time','temp','temp\_min','temp\_max','pressure','sea\_level',

'grnd\_level','humidity','weather\_id','Cloudiness','description','wind\_speed'])

for index in range(len(data1['list'])):

city\_id=data1['city']['id']

city\_name=data1['city']['name']

date=data1['list'][index]['dt\_txt'].split(" ")[0]

time=data1['list'][index]['dt\_txt'].split(" ")[1]

temp=data1['list'][index]['main']['temp']

temp\_min=data1['list'][index]['main']['temp\_min']

temp\_max=data1['list'][index]['main']['temp\_max']

pressure=data1['list'][index]['main']['pressure']

sea\_level=data1['list'][index]['main']['sea\_level']

grnd\_level=data1['list'][index]['main']['grnd\_level']

humidity=data1['list'][index]['main']['humidity']

weather\_id=data1['list'][index]['weather'][0]['id']

Cloudiness=data1['list'][index]['weather'][0]['main']

description=data1['list'][index]['weather'][0]['description']

wind\_speed=data1['list'][index]['wind']['speed']

df\_weather.loc[index]=[city\_id,city\_name,date,time,temp,temp\_min,temp\_max,pressure,sea\_level,grnd\_level,humidity,weather\_id,Cloudiness,description,wind\_speed]

df\_weather

## Power Bi Visual of Weather Report:

