PRACTICAL NO. 1

Aim: Create tables using different applications

Oracle and MySQL->

1.Create table Student: Roll_No(Primary Key), Name, Class, Course_ID(Foreign Key).

CREATE TABLE Student (Roll_No INT NOT NULL AUTO_INCREMENT PRIMARY KEY, Name VARCHAR(50) NOT NULL, Class VARCHAR(20) NOT NULL, Course_ID INT NOT NULL, FOREIGN KEY (Course_ID) REFERENCES Courses(Course_ID));

```
mysql> desc Student;
+-----------+-------------+------+-----+---------+----------------+
| Field     | Type        | Null | Key | Default | Extra          |
+-----------+-------------+------+-----+---------+----------------+
| Roll_No   | int(11)     | NO   | PRI | NULL    | auto_increment |
| Name      | varchar(50) | NO   |     |         |                |
| Class     | varchar(20) | NO   |     |         |                |
| Course_ID | int(11)     | NO   | MUL |         |                |
+-----------+-------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

2. Create table Library: Course_ID, Course_Name(Primary Key).

```
mysql> CREATE TABLE Courses (
    ->   Course_ID INT NOT NULL PRIMARY KEY,
    ->   Course_Name VARCHAR(50) NOT NULL
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> desc Courses;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| Course_ID   | int(11)     | NO   | PRI |         |       |
| Course_Name | varchar(50) | NO   |     |         |       |
+-------------+-------------+------+-----+---------+-------+
2 rows in set (0.02 sec)
```

3. Insert 5 records in each of the above tables.

```
mysql> select * from Courses;
+------------+--------------------+
| Course_ID  | Course_Name        |
+------------+--------------------+
|        101 | Computer Science   |
|        102 | Biotechnology      |
|        103 | Mathematics        |
|        104 | Physics            |
|        105 | BAF                |
+------------+--------------------+
5 rows in set (0.00 sec)
```

```
mysql> select * from student1;
+--------+---------+-------+-------------+
| rollno | name    | class | course      |
+--------+---------+-------+-------------+
|     12 | sumit   | tycs  | cs          |
|     14 | chirag  | fyai  | ai          |
|     16 | shubham | sy    | datascience |
|     18 | ansh    | tyit  | it          |
|     20 | yash    | sybms | bms         |
+--------+---------+-------+-------------+
```

4. Add a column "Credits" in the Courses table and update with values in range (2 to 4).

```
mysql> update Courses set Credits=2 where Course_ID=101;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Courses set Credits=3 where Course_ID=102;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Courses set Credits=4 where Course_ID=103;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Courses set Credits=3 where Course_ID=105;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Courses set Credits=2 where Course_ID=104;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Courses;
+-----------+-----------------+---------+
| Course_ID | Course_Name     | credits |
+-----------+-----------------+---------+
|       101 | Computer Science |       2 |
|       102 | Biotechnology    |       3 |
|       103 | Mathematics      |       4 |
|       104 | Physics          |       2 |
|       105 | BAF              |       3 |
+-----------+-----------------+---------+
```

5. Add a column "Marks" in the Student table and update with values range(0 to 100).

```
mysql> update student set Marks=floor(RAND()*101);
Query OK, 5 rows affected (0.01 sec)
Rows matched: 5  Changed: 5  Warnings: 0

mysql> select *from student;
+--------+---------+-------+-----------+-------+
| rollno | name    | class | Course_ID | Marks |
+--------+---------+-------+-----------+-------+
|     12 | sumit   | ty    |       101 |    17 |
|     13 | ansh    | sy    |       102 |    68 |
|     14 | chirag  | fy    |       103 |    84 |
|     15 | shubham | ty    |       104 |    18 |
|     16 | harsh   | sy    |       105 |    39 |
+--------+---------+-------+-----------+-------+
5 rows in set (0.00 sec)
```

6. Increase the marks of the 3rd student in the Student table by 10.

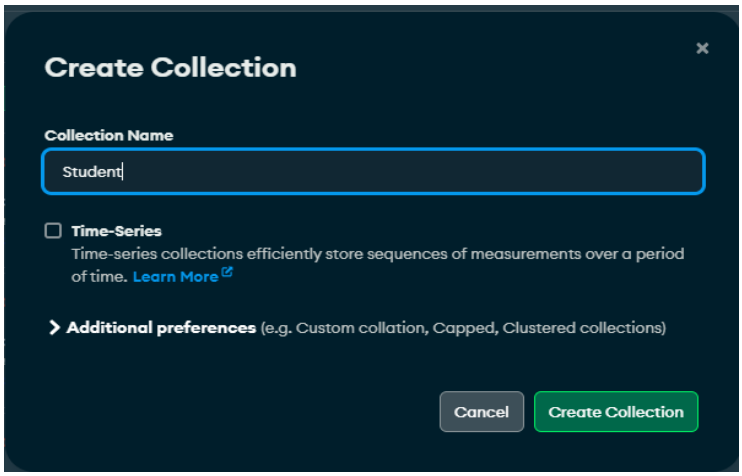
```
mysql> update student set Marks=Marks+10 where rollno=13;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from student;
+--------+---------+-------+-----------+-------+
| rollno | name    | class | Course_ID | Marks |
+--------+---------+-------+-----------+-------+
|     12 | sumit   | ty    |       101 |    17 |
|     13 | ansh    | sy    |       102 |    78 |
|     14 | chirag  | fy    |       103 |    84 |
|     15 | shubham | ty    |       104 |    18 |
|     16 | harsh   | sy    |       105 |    39 |
+--------+---------+-------+-----------+-------+
5 rows in set (0.00 sec)
```

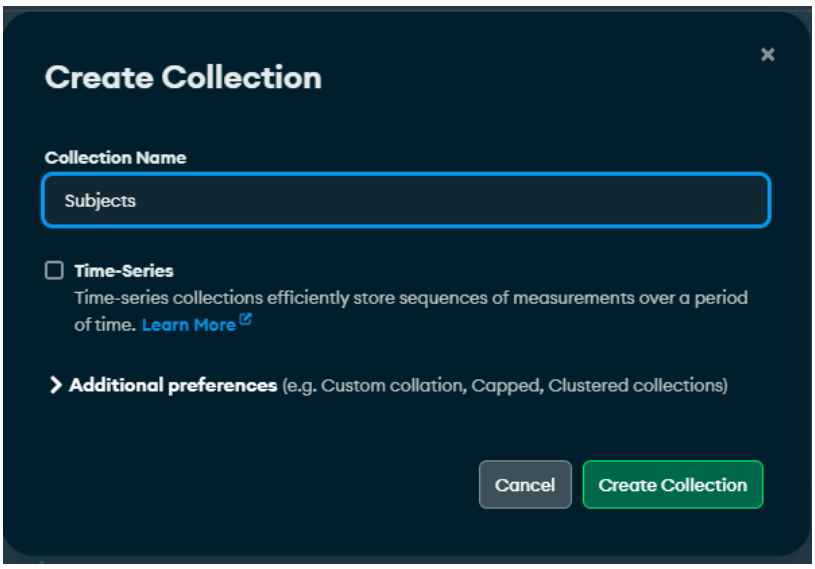
7. Find out the Course name of the 5th student in the Student table.

```
ysql> select Course_Name from Courses where Course_ID=(select Course_ID f
 student where rollno=16);
-------------+
 Course_Name |
-------------+
 BAF         |
-------------+
 row in set (0.00 sec)
```

MongoDB->

Step 1. Create a database : TYCS with 2 collections: Student and Subjects.





Step 2. Student documents will contain following fields: Roll_No, Name, Courses(May contain multiple values for subject names), CGPA, Address(Contains : Flat_No, Street_Name, City, State, Country)

Student

	_id ObjectId	Roll Int32	Name String	Courses Array	CGPA Double
1	ObjectId('669491ce34d96c…	28	"Dhruv Kushvaha"	[] 3 elements	9.9
2	ObjectId('6694925d34d96c…	31	"Manoj Maurya"	[] 3 elements	9.65
3	ObjectId('669492e334d96c…	38	"Ankit Prajapati"	[] 3 elements	9.45
4	ObjectId('6694934534d96c…	20	"Sonali Ingale"	[] 3 elements	7.45
5	ObjectId('6694936e34d96c…	3	"Shravani Anbhule"	[] 3 elements	8.76

Step 3. Subject document will contain following fields: Name, Reference_Book, Credits.

Subject

	_id ObjectId	Name String	Credits Int32	Reference_books Array	Class String
1	ObjectId('6694947c34d96c…	"Android Development"	2	[] 3 elements	"TYCS"
2	ObjectId('6694965a34d96c…	"Research Methodology"	2	[] 3 elements	"TYCS"
3	ObjectId('6694965a34d96c…	"Project Implementation"	1	[] 3 elements	"TYCS"
4	ObjectId('6694965a34d96c…	"Soft Skills"	1	[] 2 elements	"TYCS"
5	ObjectId('6694965a34d96c…	"Java Programming"	2	[] 2 elements	"TYCS"

Step 4. Update the Student documents to add College name(VES).

Update 5 documents

TYCS.Student

Filter

None

Update

Learn more about Update syntax

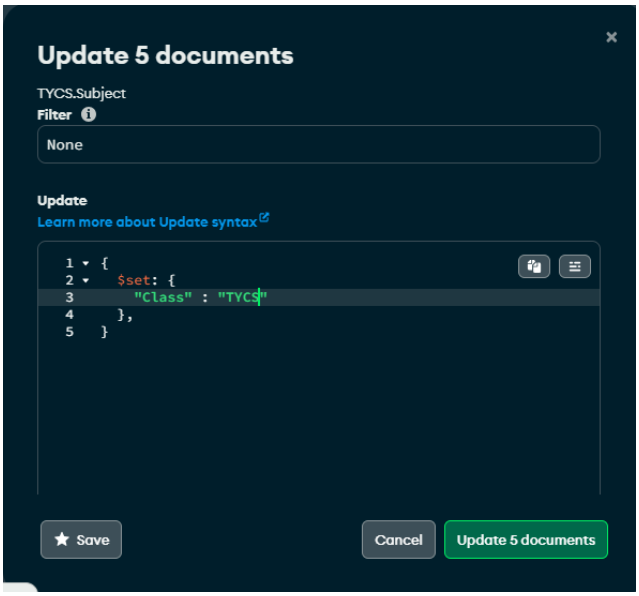
```
{
  $set: {
    "College":"VES"
  },
}
```

Save

Cancel

Update 5 documents

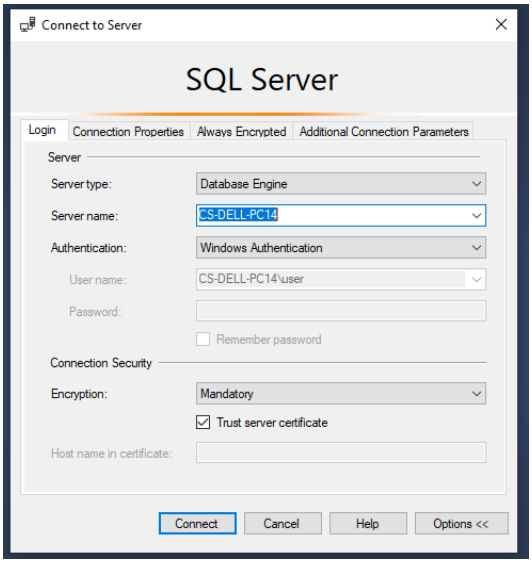
Step 5. Update the Subject documents to add Class(TYCS).



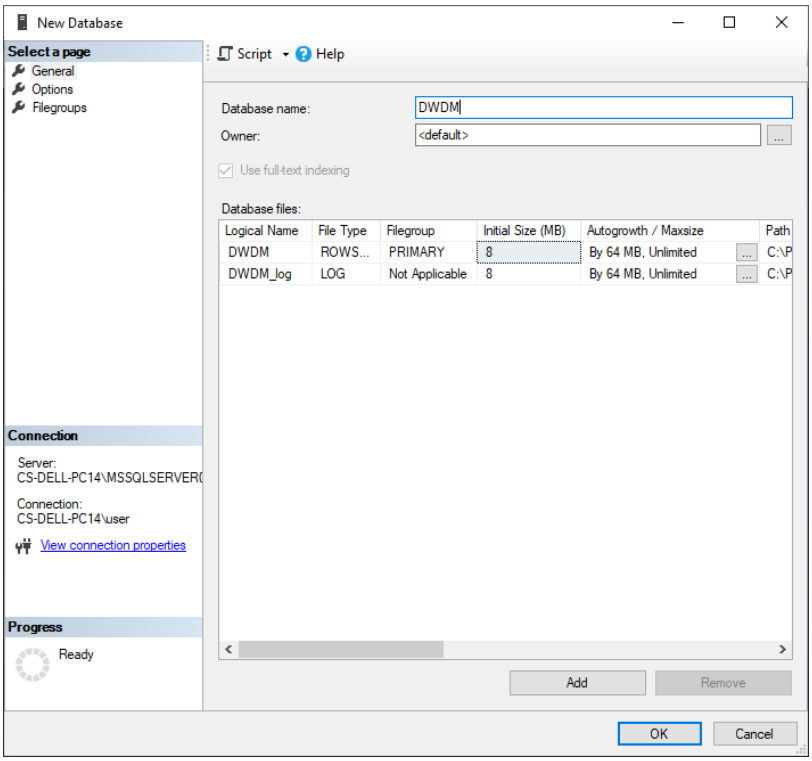
PRACTICAL NO 2

Aim: Develop an application to design a warehouse by importing various tables from the external sources.

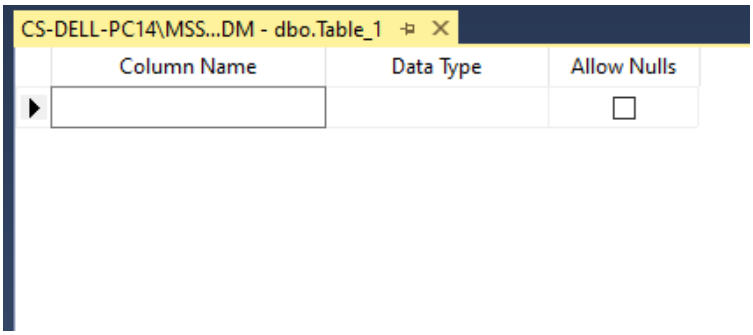
Step 1: Connect to the server in the SQL Server Management studio



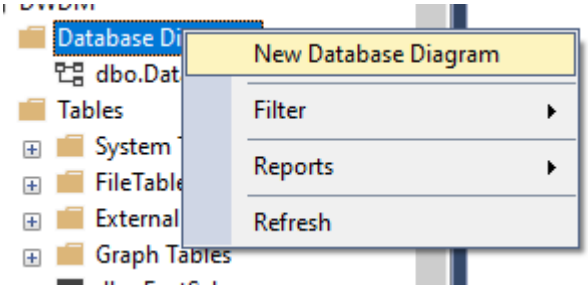
Step 2: Create database and give name DWDM



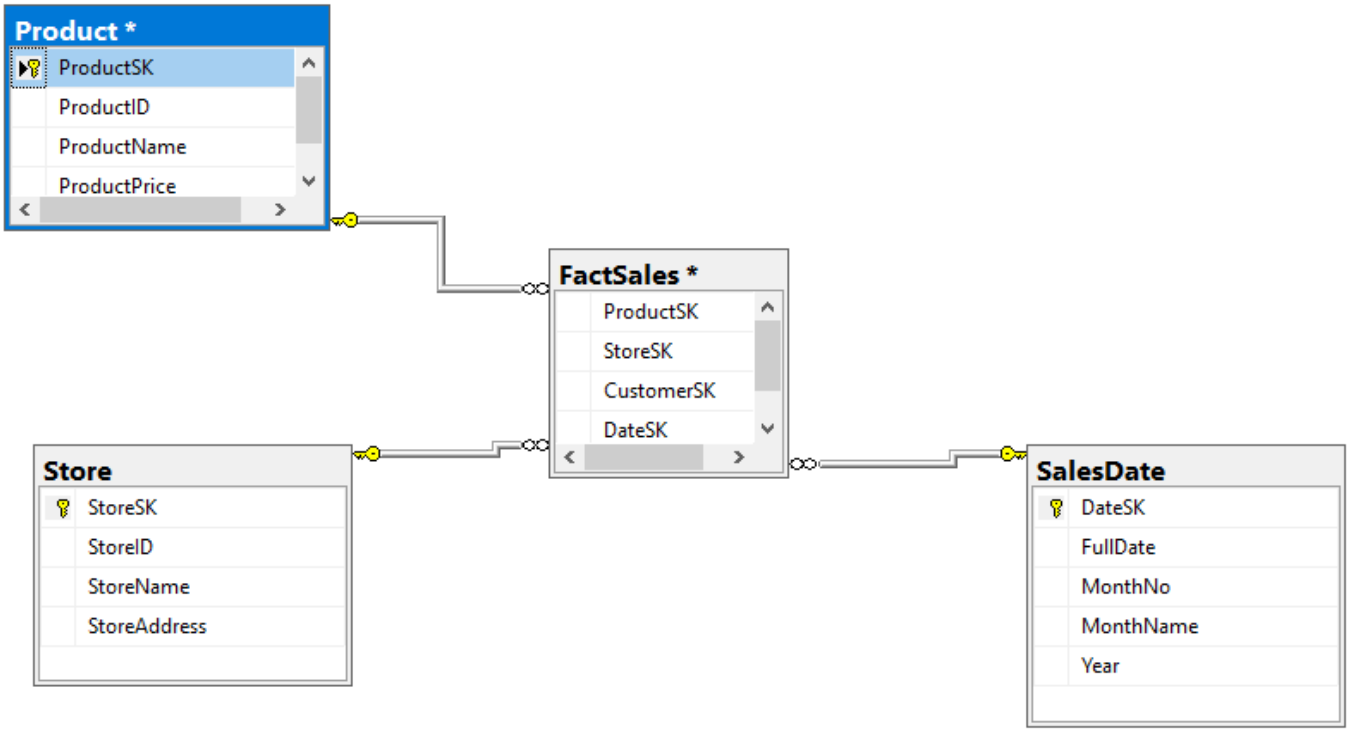
Step 3: Create four tables name them as Products, SalesDate, FactSales and Stores



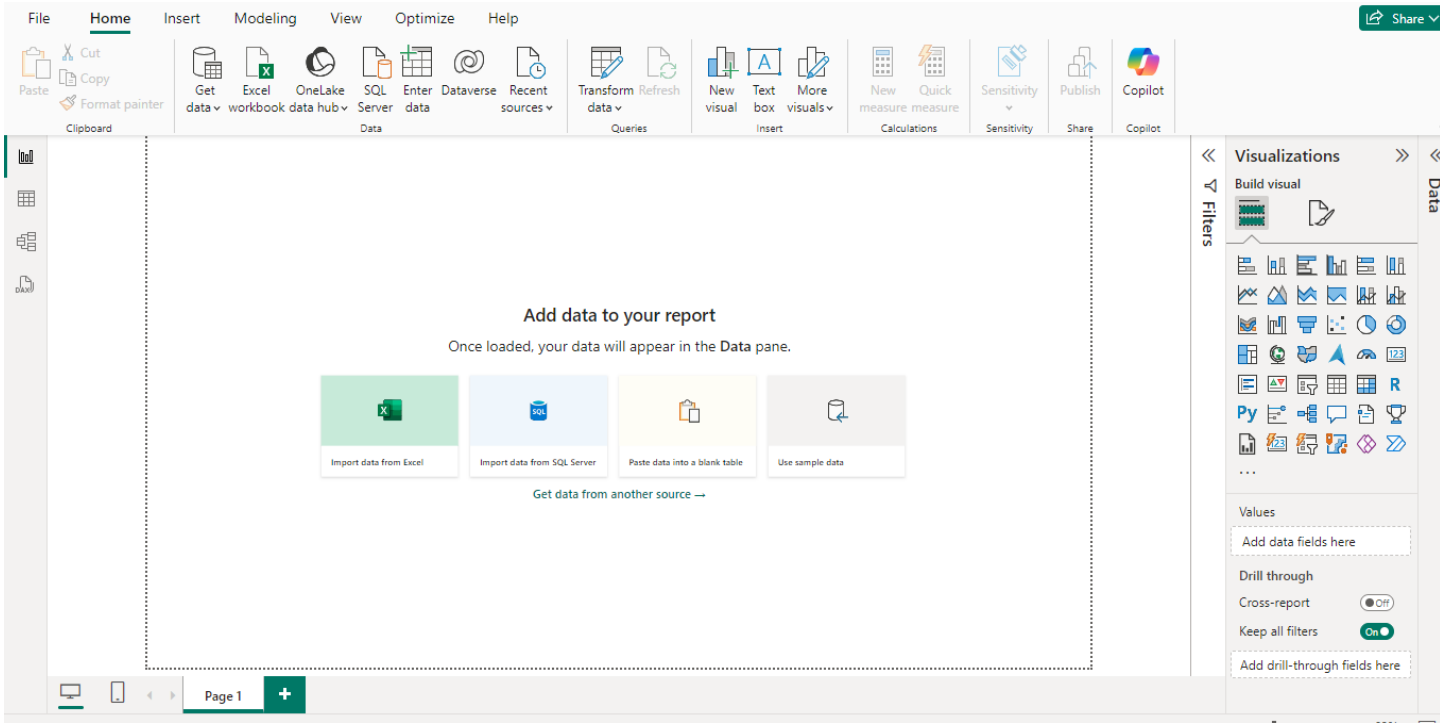
Step 4: Under the DWDM Database under database diagrams select New Database diagram



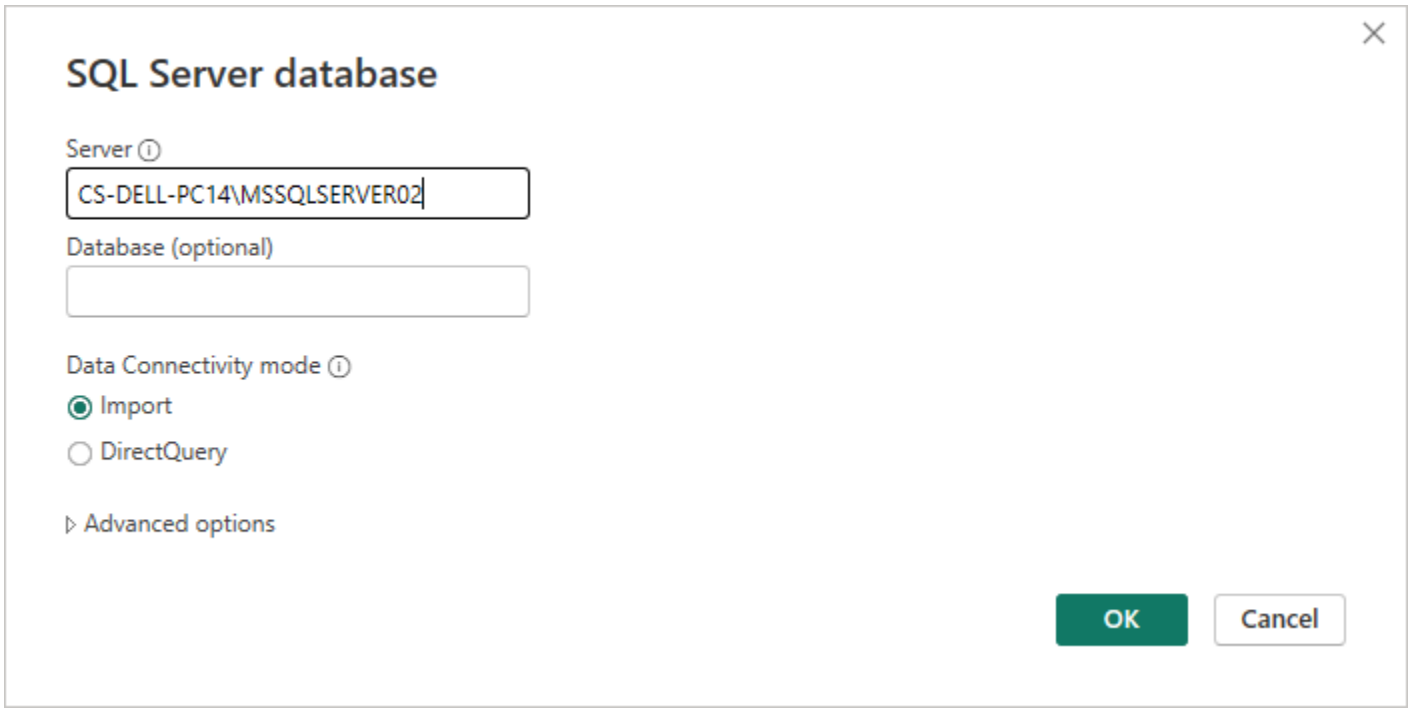
Step 5: Choose the respectives table and make the necessary relationship among them



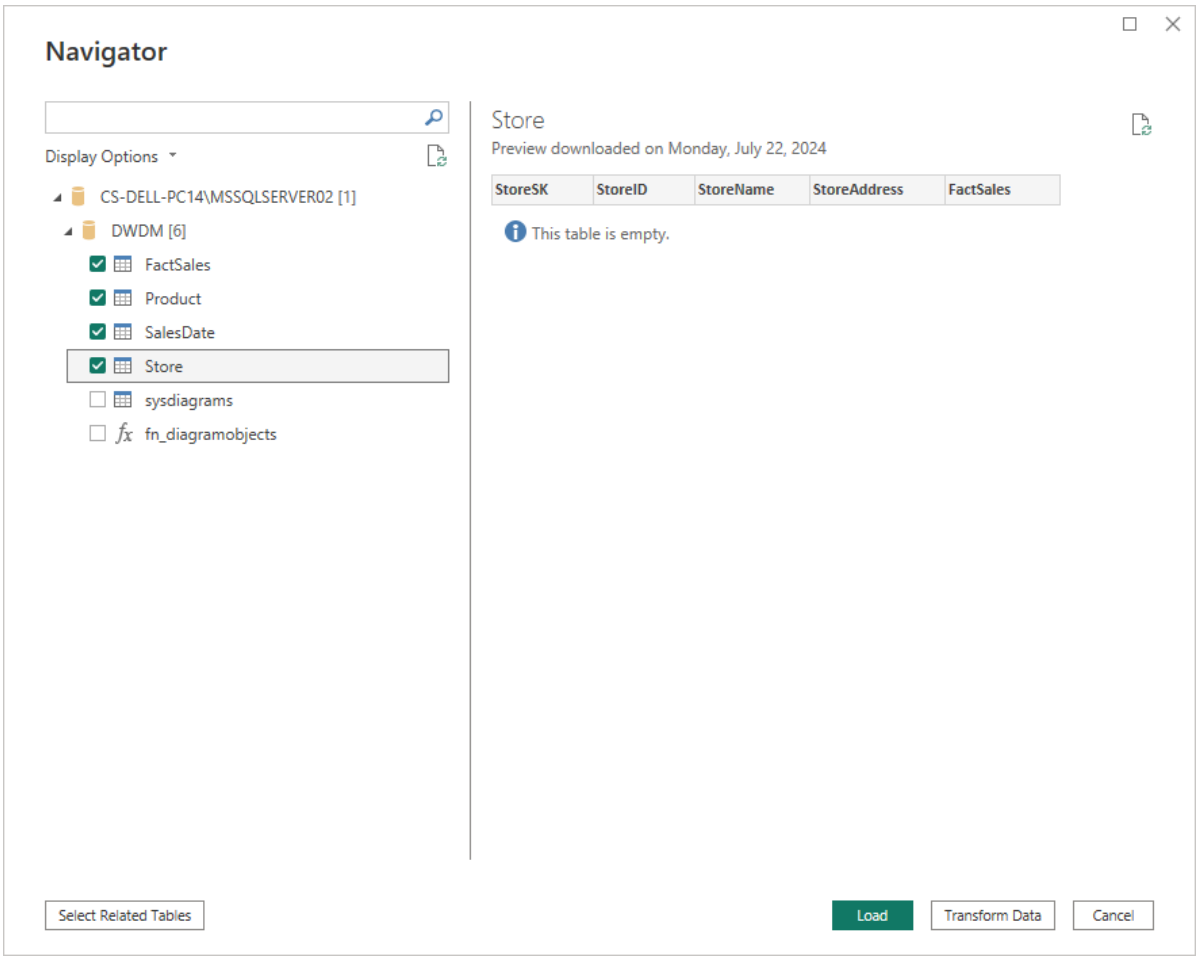
Step 6: Open Power BI Desktop and select blank report



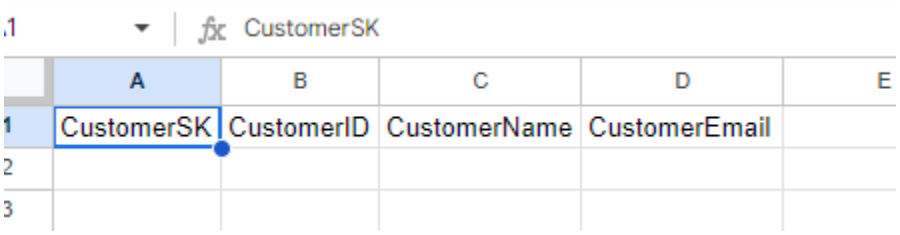
Step 7: Select SQL Server and enter the server name from the management studio



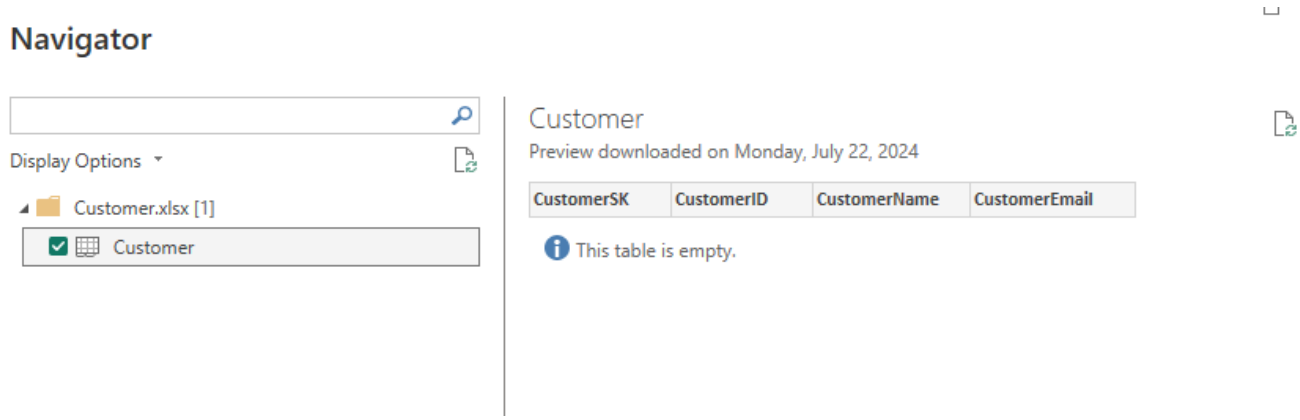
Step 8: Select the respective tables from the DWDM database in the navigator



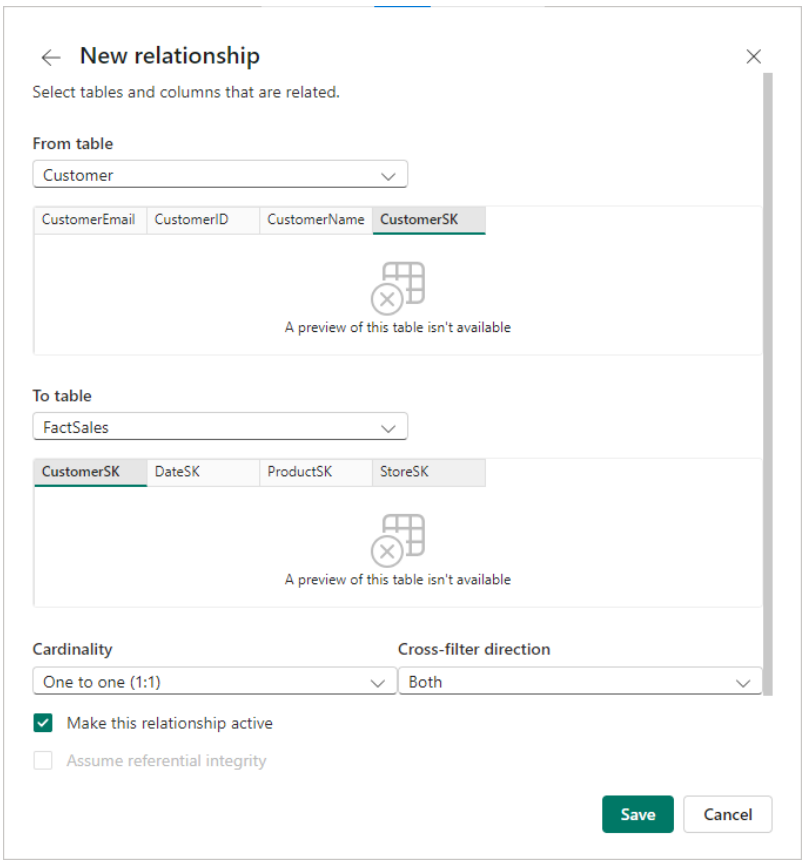
Step 9: Create an excel sheet name it as Customer with necessary columns



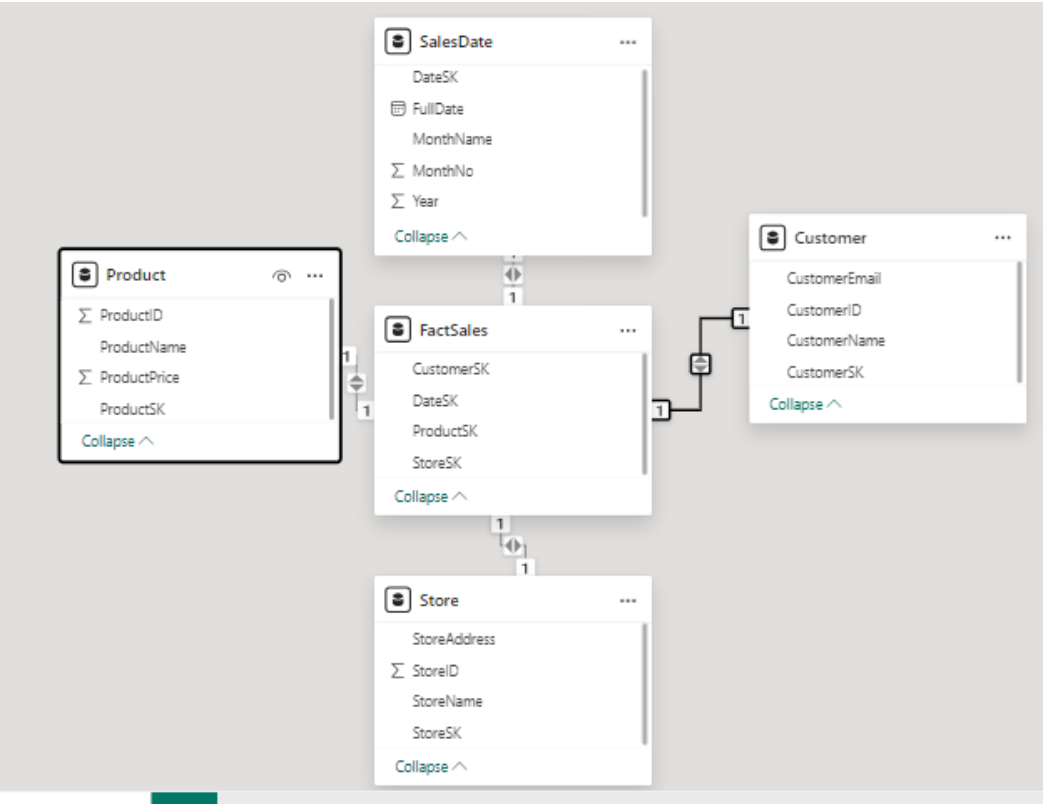
Step 10: Import the excel in the power bi project



Step 11: Make the necessary relation for the customer table



Step 12: Check the warehouse



PRACTICAL NO. 3

Aim: Execute a code for implementing ETL in Python. (PETL - Python's ETL Library)

Part A: Excel to SQL

Step 1: Create an excel sheet and name it as Employee with respective columns

EMP_ID	FirstName	LastName
101	Sumit	Bhatia
102	Ansh	Methwani
103	Harsh	Basantani
104	Nikhil	Bhatia
105	Nakul	Mangwani
106	Shubham	Jhadhav
107	Chirag	Gangwani
108	Aman	Diwedi
109	Amit	Singh
110	Yash	Gawde

Step 2: Write a python script to load this excel and send it to the SQL Server in the SSM Studio

Code :-

```
import pandas as pd
import sqlalchemy as sa
import pyodbc
print(pyodbc.drivers())

#extracting data from excel
data=pd.read_excel("D:\\TYCS\\DWDM\\employee.xlsx")
print(data)


#transforming data into new clm
data['full name']=data["first name"]+' '+data["last name"]
print(data)

#loading data in sql engine
engine=sa.create_engine('mssql+pyodbc://ASUS-27/DWDM?driver=ODBC Driver 17 for SQL Server')
data.to_sql(name='emp',con=engine,index=False,if_exists='fail')
```

Step 3: Check the SSM Studio's SQL Server for the Employee table with the new column "Full Name"

Part B: Excel to Excel

Step 1: Create new excel file with missing values and inconsistent data and name it as Sample

ID	A	B	C
100	1	45	1.2
100	2	56	1.4
101	3	48	1.1
102	4	47	1.8
103	5	65	
104	2	5000	1.4
105		57	1.6
106	5	78	1.5

Step 2: Write the python script to perform ETL and save transformed data in new excel file

code:-

```
import pandas as pd
df =
pd.read_excel("D:\\TYCS\\DWDM\\sa
mple.xlsx")
print("original dataset")
print(df)

def fill_missing_values(df):
  for col in
df.select_dtypes(include=["int","float"]
).columns:
    val = df[col].mean()
    df.fillna({col:val},inplace=True)
  return df

def drop_duplication(df,column_list):
  df =
df.drop_duplicates(subset=column_li
st)
  return df

def remove_outliners(df,column_list):
  for col in column_list:
    avg=df[col].mean()
    std=df[col].std()
    low=avg-2*std
    high=avg+2*std

df=df[df[col].between(low,high,inclusi
ve="both")]
  return df
```

```
def_processed =
(df.pipe(fill_missing_values)
.pipe(drop_duplication,"id")

.pipe(remove_outliners,["sem1","sem
2","sem2"])
)
def_processed.to_excel("dwdh2.xlsx"
)
```

Step 3: Check the new processed data in newly form excel file

PRACTICAL NO 5

Aim: Implementing OLAP using Python

Step 1: Create 4 types of below data, which can be converted to the data frame using Pandas:

1. Person
2. Sales
3. Quarter
4. Country

Step 2:

1. Get the country wise total sales, 2. Find Sales by both the person and the country, 3. Print Maximum individual sale by country

Code :-

```
import pandas as pd
data={
'persons':['manoj','sonali','manish','dhruv','ankit','amruta'],
'sales':[2000,3000,4000,5000,6000,7000],

'quarter':['Q1','Q2','Q3','Q4','Q5','Q6'],
'country':['India','UK','japan','korea','london','UK']
    }
print(data)
df=pd.DataFrame(data)
print(df)

country_wise_sales=df.groupby('country')['sales'].sum().reset_index()
print('/n country_wise sales:/n')
print(country_wise_sales)

person_country=df.groupby(['persons','country'])['sales'].sum().reset_index()
print('/n sales by both person and country :/n')
print(person_country)

pivot_data=df.pivot_table(index='country',values='sales',aggfunc='max')
print('/n maximum sales of each country')
print(pivot_data)
```

Output:-

```
{'persons': ['manoj', 'sonali', 'manish',
'dhruv', 'ankit', 'amruta'], 'sales':
[2000, 3000, 4000, 5000, 6000,
7000], 'quarter': ['Q1', 'Q2', 'Q3', 'Q4',
'Q5', 'Q6'], 'country': ['India', 'UK',
'japan', 'korea', 'london', 'UK']}
  persons  sales quarter country
0   manoj   2000      Q1   India
1  sonali   3000      Q2      UK
2  manish   4000      Q3   japan
3   dhruv   5000      Q4   korea
4   ankit   6000      Q5  london
5  amruta   7000      Q6      Uk
```

```
/n country_wise sales:/n
  country  sales
0   India   2000
1      UK  10000
2   japan   4000
3   korea   5000
4  london   6000
```

```
/n sales by both person and country
:/n
  persons country  sales
0  amruta      UK   7000
1   ankit  london   6000
2   dhruv   korea   5000
3  manish   japan   4000
4   manoj   India   2000
5  sonali      UK    300
```

```
/n maximum sales of each country
         sales
country
India     2000
UK        7000
japan     4000
korea     5000
london    6000
```

Step 3: create a data frame using the following columns-
1.stud_names,
2. category(online/offline),
3. Gender,
4. Fees

Code:-

```
import pandas as pd
data={
'Stud_names':['amruta','shubham','sa
niya','manoj','ankit','dhruv'],
```

```
'Category':['Online','Offline','Online','Offline','Online','Offline'],
'Fees':[3400,5000,2500,5500,2000,1500],
'Gender':['female','Male','Female','Male','male','male']
}
#Preparing Dataframe
df=pd.DataFrame(data)
print("Dataframe: \n")
print(df)
#pivot table
pivot_data=df.pivot_table(index='Category',columns='Gender',values="Stud_names",aggfunc="count")
print("\n")
pivot_data['Male/Female Ratio']=pivot_data['Male']/pivot_data['Female']
print(pivot_data[['Male','Female','Male/Female Ratio']]
```

Step 4: find the male/female ratio under OLAP using Python

```
print(pivot_data[['Male','Female','Male/Female Ratio']])
```

Output:-

```
Dataframe:
  Stud_names Category  Fees  Gender
0     amruta   Online  3400  female
1    shubham  Offline  5000    Male
2     saniya   Online  2500  Female
3      manoj  Offline  5500    Male
4      ankit   Online  2000    male
5      dhruv  Offline  1500    male

Gender    Male  Female  Male/Female Ratio
Category
Offline    2.0     NaN                NaN
Online     NaN     1.0                NaN
```

PRACTICAL NO 6

Aim: Implementation to perform data mining using WEKA and Python separately

Part A: Data mining using Python

Step 1: Get the dataset about the titanic passengers in the excel file and name it as titanic.csv

	A	B	C	D	E	F	G	H	I
1	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
2	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171
3	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599
4	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2.
5	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803
6	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450
7	6	0	3	Moran, Mr. James	male		0	0	330877
8	7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463

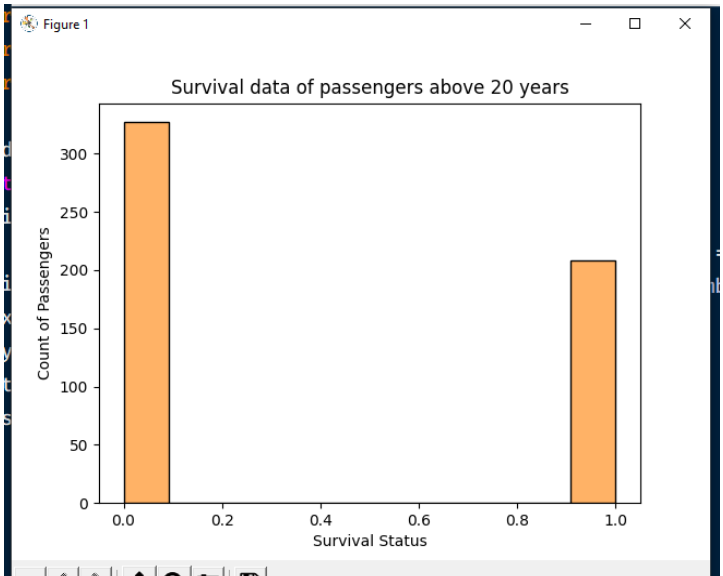
Step 2: Perform the data mining

1. **Find the number of people survived above the age 20**

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("D:\\TYCS\\DWDM\\titanic (1).csv")

survived_above_20 = df[df['Age'] > 20]['Survived']

sns.histplot(data=survived_above_20, color='#ff9933')
plt.xlabel('Survival Status')
plt.ylabel('Count of  Passenger')
plt.title('Survival data of passengers above 20 years')
plt.show()
```

Output:

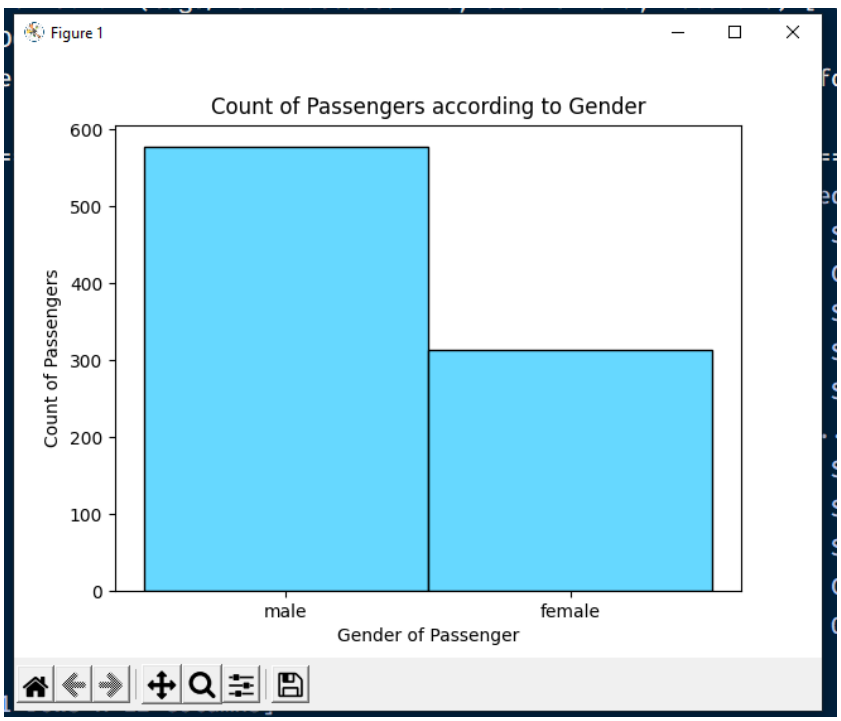


2. Visualize the distribution of gender using a histogram

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data =
pd.read_csv("D:\\TYCS\\DWDM\\titanic
(1).csv")
sns.histplot(data=data, x='Sex',
color='#33ccff')
plt.xlabel('Gender of Passenger')
plt.ylabel('Count of Passenger')
plt.title('Count of Passenger according to
gender”')
plt.show()
```

Output:

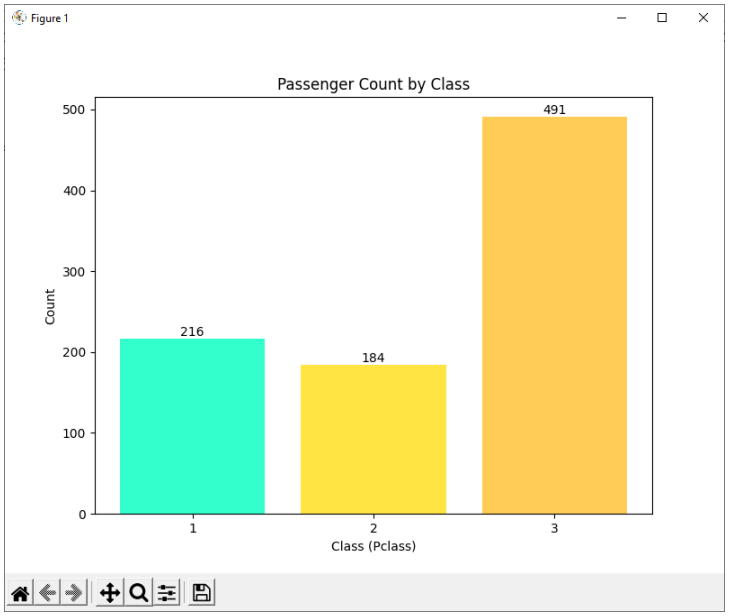


3. Show the count of passengers per class (1,2,3 etc)

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data =
pd.read_csv("D:\\TYCS\\DWDM\\titanic
(1).csv")

class_counts =
data['Pclass'].value_counts().sort_index()
plt.figure(figsize=(8, 6))
plt.bar(class_counts.index,
class_counts.values, color=['skyblue',
'orange', 'teal'])
plt.title('Passenger Count by Class')
plt.xlabel('Class (Pclass)')
plt.ylabel('Count')

for i, count in enumerate(class_counts):
    plt.text(i + 1, count, str(count),
ha='center', va='bottom')
plt.xticks(class_counts.index)
plt.show()
```

Output:



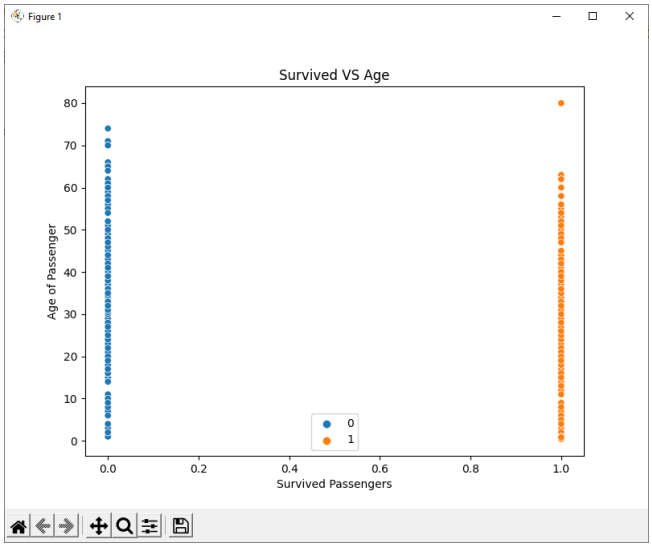
4. Show the Scatter plot of age Vs survived

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data =
pd.read_csv"D:\\TYCS\\DWDM\\titanic
(1).csv")
plt.figure(figsize=(8, 6))
```

```
sns.scatterplot(x='Survived', y='Age',
hue='Survived', data=data)
plt.title("Survived VS Age")
plt.xlabel("Survived Passengers")
plt.ylabel("Age of Passenger")
plt.legend()
plt.show()
```

Output:



Part B: Data mining using WEKA

PRACTICAL NO 7

Aim: Develop an application to pre-process data imported from external sources .

Step 1: Load the csv file

Code:

```
import pandas as pd
df = pd.read_csv("airbnbData - airbnbData.csv")
```

Step 2: Show the first few rows

Code:

```
import pandas as pd


#1.display first 10 rows.
data=pd.read_csv("airbnbData.csv",encoding='latin-1')
print(data.head().to_string())
#print(data.to_string())
```

Output:

```
   ListingID  ...                                          ShortDesc
0     281552  ...  \n    Entire home/apt Мве6 24 reviews Мве6 Har...
1     182613  ...  \n    Entire home/apt Мве6 17 reviews Мве6 Cha...
2    1587540  ...  \n    Entire home/apt Мве6 5 reviews Мве6 Char...
3     469506  ...  \n    Entire home/apt Мве6 60 reviews Мве6 Bro...
4    3937268  ...  \n    Private room Мве6 11 reviews Мве6 Brookl...

[5 rows x 65 columns]
```

Step 3: Show the values as NaN where the values are empty under hostname

Code:

```
print("------------------------- null values ---------------------------")
empty=pd.isnull(data['HostName'])
print(empty)
```

Output:

```
------------------------- null values --------------------------
0       False
1       False
2       False
3       False
4       False
       ...
2017    False
2018    False
2019    False
2020    False
2021    False
Name: HostName, Length: 2022, dtype: bool
```

Step 4: Show the data types of each column

Code:

```
print("------------------------- sum of the null values ---------------------------")
abt_lst=data['AboutListing'].isnull().sum()
print(abt_lst)
print(data.dtypes)
```

Output :

```
------------------------- sum of the null values ---------------------------
3
ListingID      int64
Title         object
UserID         int64
baseurl       object
Price         int64
             ...
S_CheckIn     object
S_Checkout    object
S_NumBeds     object
S_PropType    object
ShortDesc     object
Length: 65, dtype: object
```

Step 5: Set index to id

Code :

air_df = df.set_index("HostName", append=False)

print(air_df.head())

Output:

```
                ListingID  ...                                          ShortDesc
HostName                   ...
Mary Catherine     281552  ...  \n    Entire home/apt Мвеб 24 reviews Мвеб Har...
Max                182613  ...  \n    Entire home/apt Мвеб 17 reviews Мвеб Cha...
Finola            1587540  ...  \n    Entire home/apt Мвеб 5 reviews Мвеб Char...
Rupal              469506  ...  \n    Entire home/apt Мвеб 60 reviews Мвеб Bro...
Natasha           3937268  ...  \n    Private room Мвеб 11 reviews Мвеб Brookl...

[5 rows x 64 columns]
```

Step 6: Find the location of Brooklyn under neighborhood group

Code:

```
brooklyn_location = df.loc[df["neighbourhood group"] == "Brooklyn"]
print(brooklyn_location)
```

Output:

```
            id                                NAME  ...                                house_rules license
0      1001254   Clean & quiet apt home by the park  ...  Clean up and treat the home the way you'd like...     NaN
3      1002755                                  NaN  ...                                                NaN     NaN
6      1004650                        BlissArtsSpace!  ...  Please no shoes in the house so bring slippers...     NaN
7      1005202                        BlissArtsSpace!  ...  House Guidelines for our BnB We are delighted ...     NaN
16     1010173      Only 2 stops to Manhattan studio  ...  Absolutely no smoking in the building, handlin...     NaN
...        ...                                   ...  ...                                                ...     ...
102587 6088571    Adorable One-Bed in Williamsburg!  ...  * Check-in time is 2PM. Check-out time is 11am...     NaN
102589 6089676  Lrg room 1 block from Prospect Park  ...  House Rules 1. Check-in is 4 pm local time. If...     NaN
102590 6090228  Wonderful artists' loft in Brooklyn  ...                                            #NAME?     NaN
102594 6092437            Spare room in Williamsburg  ...  No Smoking No Parties or Events of any kind Pl...     NaN
102596 6093542        Cozy, bright room in Brooklyn  ...                                                NaN     NaN

[41842 rows x 26 columns]
```

Step 7: Find out how many null values are there under host_identifity_verified

Code:

```
null_count = df["host_identity_verified"].isnull().sum()
print("Total null values in host_identity_verified are column: "+ null_count)
```

Output :

```
Total null values in host_identity_verified are column :  289
```

Step 8: How many hotels are instant_bookable

Code:

```
df = pd.read_csv("airbnbData - airbnbData.csv")
instant_bookable_hotels = df[df["BookInstantly"] == "Yes"]
count_instant_bookable = len(instant_bookable_hotels)
print("Number of instant bookable hotels", count_instant_bookable)
```

Out

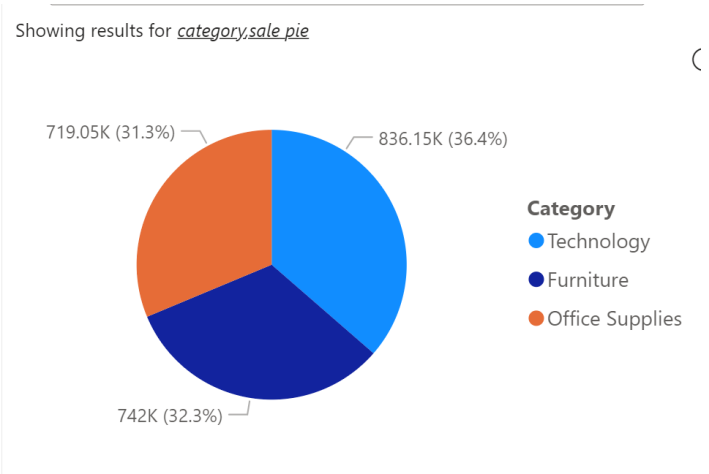
```
Number of instant bookable hotels 142
```

PRACTICAL NO 8

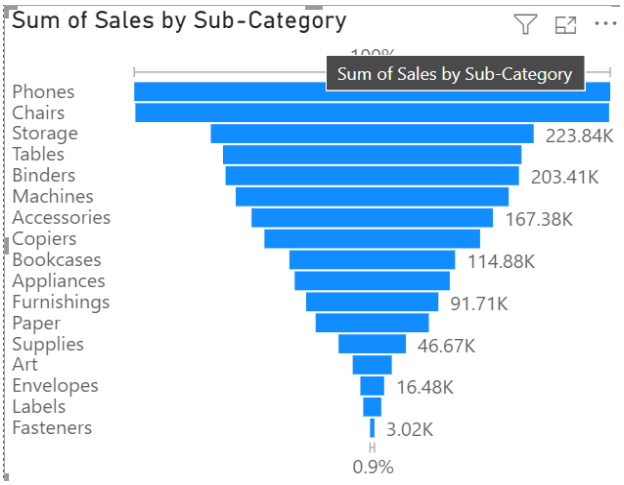
Aim: Build a BI report using various tables created in the warehouse on Tableau Public/PowerBI

Steps :-

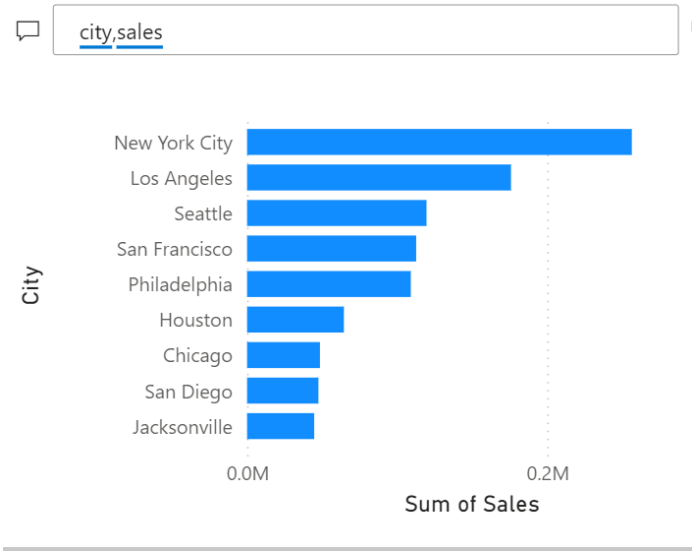
1. **Open Power BI Desktop**:
2. **Load Your Data**:
   - Click on **'Get Data'** in the Home ribbon.
   - Select the type of data source you are using (e.g., Excel, SQL Server, etc.).
   - Connect to your data source and load the data into Power BI.
3. **Create Visualizations**:
   - **Category, Sales (Pie Chart)**:
     - Drag the **'Category'** field to the **Axis** area.
     - Drag the **'Sales'** field to the **Values** area.
     - Select the **Pie Chart** visualization from the Visualizations pane.



   - **Subcategory, Sales**:
     - Drag the **'Subcategory'** field to the **Axis** area.
     - Drag the **'Sales'** field to the **Values** area.
     - Choose an appropriate chart type (e.g., Bar Chart).

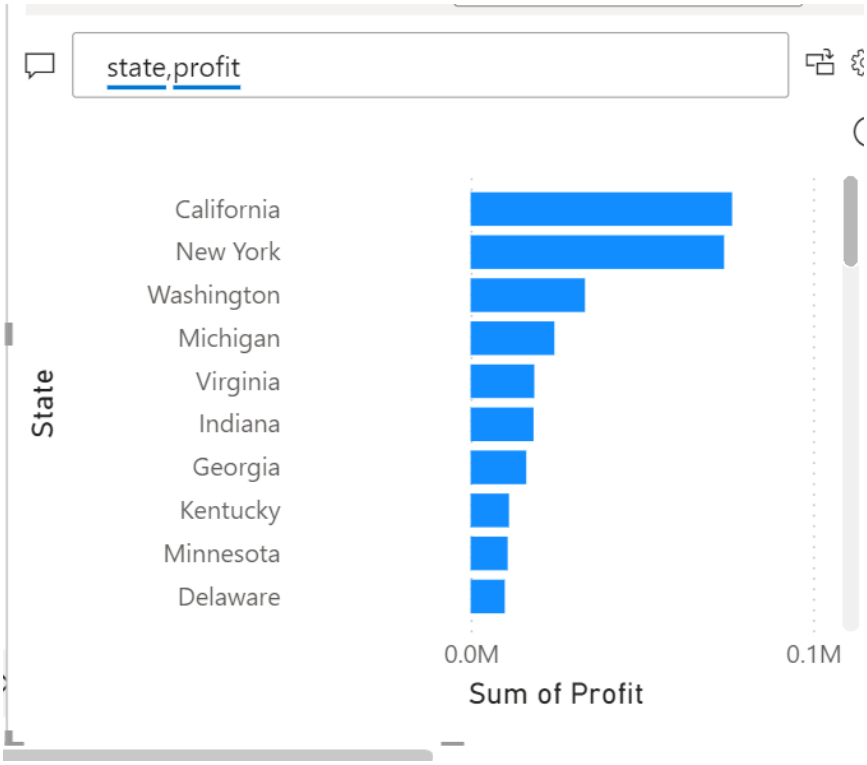


- **City, Sales**:
  - Drag the **'City'** field to the **Axis** area.
  - Drag the **'Sales'** field to the **Values** area.
  - Choose an appropriate chart type (e.g., Column Chart).

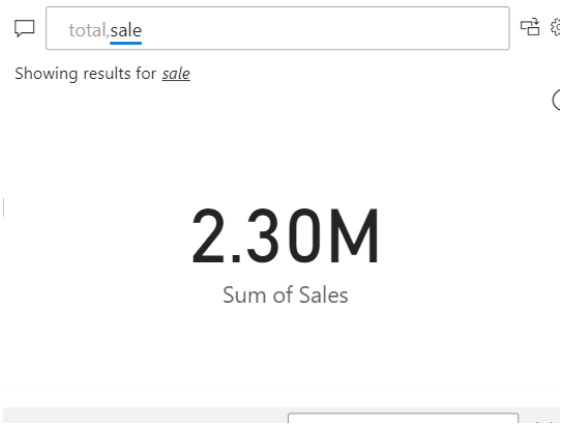


- **Country, Sales (Map)**:
  - Drag the **'Country'** field to the **Location** area.
  - Drag the **'Sales'** field to the **Size** area.
  - Select the **Map** visualization from the Visualizations pane.

- **State, Profit**:
  - Drag the **'State'** field to the **Axis** area.
  - Drag the **'Profit'** field to the **Values** area.
  - Choose an appropriate chart type (e.g., Bar Chart).



- **Total, Sales**:
  - Create a card visualization to display the total sales.
  - Drag the **'Sales'** field to the **Values** area.
  - Select the **Card** visualization from the Visualizations pane.



PRACTICAL NO 9

Aim: Execute a wordcount problem using Spark and NLTK.

Code:-

```
import nltk
nltk.download('punkt')
from nltk.tokenize import word_tokenize
statment=str(input("enter a sentence:"))
tokens=word_tokenize(statment)
print("the no of words in the given sentence is:",len(tokens))
```

Output:-

```
enter a sentence:this is amruta
the no of words in the given sentence is: 3
PS D:\TYCS 2024\dwdm>
```

Code for the collocations :-

```
from nltk.util import ngrams #ngrams is the pair of  words (collocations)
from nltk.tokenize import word_tokenize,sent_tokenize
#from nltk.collocations import*

statment=['sun','rises','in','the','east','it','sets','in','the','west']
bigrams=ngrams(statment,2)
bigrams_count={}
for b in bigrams:
    if b not in bigrams_count:
        bigrams_count[b]=1
    else:
        bigrams_count[b]+=1

print(statment)
print("Biggrams:",bigrams_count)
```

Output:-

```
['sun', 'rises', 'in', 'the', 'east', 'it', 'sets', 'in', 'the', 'west']
Biggrams: {('sun', 'rises'): 1, ('rises', 'in'): 1, ('in', 'the'): 2, ('the', 'east'): 1, ('east',
'it'): 1, ('it', 'sets'): 1, ('sets', 'in'): 1, ('the', 'west'): 1}
```