# Lab 10 > Part B

Name: Debatreya Das Roll No. 12212070 CS A4 Data Mining Lab 10

### Create A data warehouse with given data using PostgreSQL tool and perform data extraction using SQL and OLAP.

Write the following queries:

1. Find the total sales.
2. Find total sales for each city.
3. Find total sales for each state.
4. Find total sales for each country.
5. Find sales of all cities of a specific state in a specific year.
6. Find year-wise total sales for each state.
7. Find year-wise total sales for each country.

### Importing Required Libraries for Postgres

import psycopg2  
from psycopg2 import sql  
import pandas as pd

### Create Connection

# Establish connection to PostgreSQL  
conn = psycopg2.connect(  
 dbname="datacube",  
 user="postgres",  
 password="postgres",  
 host="localhost",  
 port="5432"  
)  
cursor = conn.cursor()

### Create the tables

create\_tables = [  
 """  
 CREATE TABLE IF NOT EXISTS Products (  
 P\_Id INTEGER PRIMARY KEY,  
 P\_name VARCHAR(100),  
 Category VARCHAR(50),  
 Price REAL  
 );  
 """,  
 """  
 CREATE TABLE IF NOT EXISTS Locations (  
 Loc\_Id INTEGER PRIMARY KEY,  
 City VARCHAR(50),  
 State VARCHAR(50),  
 Country VARCHAR(50)  
 );  
 """,  
 """  
 CREATE TABLE IF NOT EXISTS Times (  
 Time\_Id INTEGER PRIMARY KEY,  
 Date VARCHAR(10),  
 Week INTEGER,  
 Month INTEGER,  
 Quarter INTEGER,  
 Year INTEGER  
 );  
 """,  
 """  
 CREATE TABLE IF NOT EXISTS Sales (  
 Loc\_Id INTEGER REFERENCES Locations(Loc\_Id),  
 P\_Id INTEGER REFERENCES Products(P\_Id),  
 Time\_Id INTEGER REFERENCES Times(Time\_Id),  
 Sale REAL,  
 PRIMARY KEY (Loc\_Id, P\_Id, Time\_Id)  
 );  
 """  
]  
  
# Drop tables if they already exist   
cursor.execute("DROP TABLE IF EXISTS Sales;")  
cursor.execute("DROP TABLE IF EXISTS Products;")  
cursor.execute("DROP TABLE IF EXISTS Locations;")  
cursor.execute("DROP TABLE IF EXISTS Times;")  
  
# Execute each CREATE TABLE statement  
for command in create\_tables:  
 cursor.execute(command)  
  
# Commit changes and close connection  
conn.commit()  
  
print("Tables created successfully!")

Tables created successfully!

### Inserting values in the Tables

# Insert data into Products table  
products\_data = [  
 (11, 'Lee Jeans', 'Apparel', 25),  
 (12, 'Zord', 'Toys', 18),  
 (13, 'Biro Pen', 'Stationery', 2)  
]  
cursor.executemany("INSERT INTO Products (P\_Id, P\_name, Category, Price) VALUES (%s, %s, %s, %s);", products\_data)  
  
# Insert data into Locations table  
locations\_data = [  
 (1, 'Madison', 'WI', 'USA'),  
 (2, 'Fresno', 'CA', 'USA'),  
 (5, 'Chennai', 'TN', 'India')  
]  
cursor.executemany("INSERT INTO Locations (Loc\_Id, City, State, Country) VALUES (%s, %s, %s, %s);", locations\_data)  
  
# Insert data into Times table  
times\_data = [  
 (1, '2023-01-01', 1, 1, 1, 2023),  
 (2, '2023-04-01', 14, 4, 2, 2023),  
 (3, '2023-07-01', 27, 7, 3, 2023)  
]  
cursor.executemany("INSERT INTO Times (Time\_Id, Date, Week, Month, Quarter, Year) VALUES (%s, %s, %s, %s, %s, %s);", times\_data)  
  
# Insert data into Sales table  
sales\_data = [  
 (1, 11, 1, 25), (1, 11, 2, 8), (1, 11, 3, 15),  
 (1, 12, 1, 30), (1, 12, 2, 20), (1, 12, 3, 50),  
 (1, 13, 1, 8), (1, 13, 2, 10), (1, 13, 3, 10),  
 (2, 11, 1, 35), (2, 11, 2, 22), (2, 11, 3, 10),  
 (2, 12, 1, 26), (2, 12, 2, 45), (2, 12, 3, 40),  
 (2, 13, 1, 20), (2, 13, 2, 20), (2, 13, 3, 40)  
]  
cursor.executemany("INSERT INTO Sales (Loc\_Id, P\_Id, Time\_Id, Sale) VALUES (%s, %s, %s, %s);", sales\_data)  
  
# Commit the transaction  
conn.commit()  
  
  
print("Data inserted successfully!")

Data inserted successfully!

### SELECT \* FROM < TABLE > to check if its inserted properly

import pandas as pd  
  
# Select all rows from the Products table  
cursor.execute("SELECT \* FROM Products;")  
products = cursor.fetchall()  
  
print("Products Table")  
print(pd.DataFrame(products, columns=['P\_Id', 'P\_name', 'Category', 'Price']))  
  
# Select all rows from the Locations table  
cursor.execute("SELECT \* FROM Locations;")  
locations = cursor.fetchall()  
  
print("Locations Table")  
print(pd.DataFrame(locations, columns=['Loc\_Id', 'City', 'State', 'Country']))  
  
# Select all rows from the Times table  
cursor.execute("SELECT \* FROM Times;")  
times = cursor.fetchall()  
  
print("Times Table")  
print(pd.DataFrame(times, columns=['Time\_Id', 'Date', 'Week', 'Month', 'Quarter', 'Year']))  
  
# Select all rows from the Sales table  
cursor.execute("SELECT \* FROM Sales;")  
sales = cursor.fetchall()  
  
print("Sales Table")  
print(pd.DataFrame(sales, columns=['Loc\_Id', 'P\_Id', 'Time\_Id', 'Sale']))

Products Table  
 P\_Id P\_name Category Price  
0 11 Lee Jeans Apparel 25.0  
1 12 Zord Toys 18.0  
2 13 Biro Pen Stationery 2.0  
Locations Table  
 Loc\_Id City State Country  
0 1 Madison WI USA  
1 2 Fresno CA USA  
2 5 Chennai TN India  
Times Table  
 Time\_Id Date Week Month Quarter Year  
0 1 2023-01-01 1 1 1 2023  
1 2 2023-04-01 14 4 2 2023  
2 3 2023-07-01 27 7 3 2023  
Sales Table  
 Loc\_Id P\_Id Time\_Id Sale  
0 1 11 1 25.0  
1 1 11 2 8.0  
2 1 11 3 15.0  
3 1 12 1 30.0  
4 1 12 2 20.0  
5 1 12 3 50.0  
6 1 13 1 8.0  
7 1 13 2 10.0  
8 1 13 3 10.0  
9 2 11 1 35.0  
10 2 11 2 22.0  
11 2 11 3 10.0  
12 2 12 1 26.0  
13 2 12 2 45.0  
14 2 12 3 40.0  
15 2 13 1 20.0  
16 2 13 2 20.0  
17 2 13 3 40.0

### Writing the SQL (without OLAP)

#### Find the total sales

query = """  
SELECT SUM(Sale) AS Total\_Sales  
FROM Sales;  
"""  
  
cursor.execute(query)  
total\_sales = cursor.fetchone()  
  
print("Total Sales: $", total\_sales[0])

Total Sales: $ 434.0

#### Find total sales for each city

query = """  
SELECT L.City, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
GROUP BY L.City;  
"""  
  
cursor.execute(query)  
city\_sales = cursor.fetchall()  
  
print("City-wise Sales")  
print(pd.DataFrame(city\_sales, columns=['City', 'Total\_Sales']))

City-wise Sales  
 City Total\_Sales  
0 Fresno 258.0  
1 Madison 176.0

#### Find total sales for each state.

query = """  
SELECT L.State, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
GROUP BY L.State;  
"""  
  
cursor.execute(query)  
state\_sales = cursor.fetchall()  
  
print("State-wise Sales")  
print(pd.DataFrame(state\_sales, columns=['State', 'Total\_Sales']))

State-wise Sales  
 State Total\_Sales  
0 WI 176.0  
1 CA 258.0

#### Find total sales for each country.

query = """  
SELECT L.Country, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
GROUP BY L.Country;  
"""  
  
cursor.execute(query)  
country\_sales = cursor.fetchall()  
  
print("Country-wise Sales")  
print(pd.DataFrame(country\_sales, columns=['Country', 'Total\_Sales']))

Country-wise Sales  
 Country Total\_Sales  
0 USA 434.0

#### Find sales of all cities of a specific state (CA) in a specific year (2023).

query = """  
SELECT T.Year, L.City, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
JOIN Times T ON S.Time\_Id = T.Time\_Id  
WHERE T.Year = 2023 AND L.State = 'CA'  
GROUP BY T.Year, L.City  
ORDER BY T.Year, L.City;  
"""  
  
cursor.execute(query)  
city\_sales\_2023 = cursor.fetchall()  
  
print("2023 City Sales in CA")  
print(pd.DataFrame(city\_sales\_2023, columns=['Year', 'City', 'Total\_Sales']))

2023 City Sales in CA  
 Year City Total\_Sales  
0 2023 Fresno 258.0

#### Find year-wise total sales for each state.

query = """  
SELECT T.Year, L.State, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
JOIN Times T ON S.Time\_Id = T.Time\_Id  
GROUP BY T.Year, L.State  
ORDER BY T.Year, L.State;  
"""  
  
cursor.execute(query)  
year\_state\_sales = cursor.fetchall()  
  
print("Year and State-wise Sales")  
print(pd.DataFrame(year\_state\_sales, columns=['Year', 'State', 'Total\_Sales']))

Year and State-wise Sales  
 Year State Total\_Sales  
0 2023 CA 258.0  
1 2023 WI 176.0

#### Find year-wise total sales for each country.

query = """  
SELECT T.Year, L.Country, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
JOIN Times T ON S.Time\_Id = T.Time\_Id  
GROUP BY T.Year, L.Country  
ORDER BY T.Year, L.Country;  
"""  
  
cursor.execute(query)  
year\_country\_sales = cursor.fetchall()  
  
print("Year and Country-wise Sales")  
print(pd.DataFrame(year\_country\_sales, columns=['Year', 'Country', 'Total\_Sales']))

Year and Country-wise Sales  
 Year Country Total\_Sales  
0 2023 USA 434.0

### OLAP Queries

#### Find the total sales

query = """  
SELECT SUM(Sale) AS Total\_Sales  
FROM Sales  
GROUP BY CUBE(Loc\_Id, P\_Id, Time\_Id);  
"""  
  
cursor.execute(query)  
total\_sales = cursor.fetchone()  
  
print("Total Sales: $", total\_sales[0])

Total Sales: $ 434.0

#### Find total sales for each city.

query = """  
SELECT L.City, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
GROUP BY ROLLUP (L.City);  
"""  
  
cursor.execute(query)  
city\_sales = cursor.fetchall()  
  
print("City-wise Sales")  
print(pd.DataFrame(city\_sales, columns=['City', 'Total\_Sales']))

City-wise Sales  
 City Total\_Sales  
0 None 434.0  
1 Fresno 258.0  
2 Madison 176.0

#### Find total sales for each state.

query = """  
SELECT L.State, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
GROUP BY ROLLUP (L.State);  
"""  
  
cursor.execute(query)  
state\_sales = cursor.fetchall()  
  
print("State-wise Sales")  
print(pd.DataFrame(state\_sales, columns=['State', 'Total\_Sales']))

State-wise Sales  
 State Total\_Sales  
0 None 434.0  
1 WI 176.0  
2 CA 258.0

#### Find total sales for each country.

query = """  
SELECT L.Country, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
GROUP BY ROLLUP (L.Country);  
"""  
  
cursor.execute(query)  
country\_sales = cursor.fetchall()  
  
print("Country-wise Sales")  
print(pd.DataFrame(country\_sales, columns=['Country', 'Total\_Sales']))

Country-wise Sales  
 Country Total\_Sales  
0 None 434.0  
1 USA 434.0

#### Find sales of all cities of a specific state (WI) in a specific year (2023).

query = """  
SELECT L.State, L.City, T.Year, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
JOIN Times T ON S.Time\_Id = T.Time\_Id  
WHERE L.State = 'WI' AND T.Year = '2023'  
GROUP BY ROLLUP (L.City, L.State, T.Year);  
"""  
  
cursor.execute(query)  
city\_sales\_2023 = cursor.fetchall()  
  
print("2023 City Sales in WI")  
print(pd.DataFrame(city\_sales\_2023, columns=['State', 'City', 'Year', 'Total\_Sales']))

2023 City Sales in WI  
 State City Year Total\_Sales  
0 None None NaN 176.0  
1 WI Madison 2023.0 176.0  
2 WI Madison NaN 176.0  
3 None Madison NaN 176.0

#### Find year-wise total sales for each state.

query = """  
SELECT T.Year, L.State, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
JOIN Times T ON S.Time\_Id = T.Time\_Id  
GROUP BY ROLLUP (T.Year, L.State)  
ORDER BY T.Year, L.State;  
"""  
  
cursor.execute(query)  
year\_state\_sales = cursor.fetchall()  
  
print("Year and State-wise Sales")  
print(pd.DataFrame(year\_state\_sales, columns=['Year', 'State', 'Total\_Sales']))

Year and State-wise Sales  
 Year State Total\_Sales  
0 2023.0 CA 258.0  
1 2023.0 WI 176.0  
2 2023.0 None 434.0  
3 NaN None 434.0

#### Find year-wise total sales for each country.

query = """  
SELECT T.Year, L.Country, SUM(S.Sale) AS Total\_Sales  
FROM Sales S  
JOIN Locations L ON S.Loc\_Id = L.Loc\_Id  
JOIN Times T ON S.Time\_Id = T.Time\_Id  
GROUP BY ROLLUP (T.Year, L.Country)  
ORDER BY T.Year, L.Country;  
"""  
  
cursor.execute(query)  
year\_country\_sales = cursor.fetchall()  
  
print("Year and Country-wise Sales")  
print(pd.DataFrame(year\_country\_sales, columns=['Year', 'Country', 'Total\_Sales']))

Year and Country-wise Sales  
 Year Country Total\_Sales  
0 2023.0 USA 434.0  
1 2023.0 None 434.0  
2 NaN None 434.0