EXPERIMENT NO.03

Name:-Ujawala Sinha
Roll no.553

```
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:

psql (18.0)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.
```

1. **PostgreSQL SQL Code**

   -- Customer Dimension

```
postgres=# create table dim_customer (customer_id SERIAL PRIMARY KEY, customer_name VARCHAR(100), country VARCHAR(50));
CREATE TABLE
```

-- Agent Dimension

```
postgres=# create table dim_agent (agent_id SERIAL PRIMARY KEY, agent_name VARCHAR(100), department VARCHAR(50));
CREATE TABLE
```

-- Time Dimension

```
postgres=# create table dim_time (time_id SERIAL PRIMARY KEY, call_date DATE, week INT, month VARCHAR(20), year INT);
CREATE TABLE
```

-- Issue Type Dimension

```
postgres=# create table dim_issue_type (issue_id SERIAL PRIMARY KEY, issue_type VARCHAR(100), severity_level VARCHAR(20));
 CREATE TABLE
```

-- Fact Table: Support Calls

```
postgres=# CREATE TABLE fact_support_calls (call_id SERIAL PRIMARY KEY,  customer_id INT REFERENCES dim_customer(customer_id),agent_id INT REFERENCES dim_agent(agent_id),time_id INT REFERENCES dim_time(time_id),issue_id INT REFERENCES dim_issue_type(issue_id),call_duration_minutes INT,issue_resolved BOOLEAN);
CREATE TABLE
```

2. Sample Data -- Insert into Dimensions

```
postgres=# INSERT INTO dim_customer (customer_name, country) VALUES ('John Doe', 'India');
INSERT 0 1
postgres=# INSERT INTO dim_agent (agent_name, department) VALUES ('Alice Smith', 'Tech Support');
```

```
INSERT 0 1
postgres=# INSERT INTO dim_time (call_date, week, month, year) VALUES ('2025-07-01', 27, 'July',
2025);
INSERT 0 1
postgres=# INSERT INTO dim_issue_type (issue_type, severity_level) VALUES ('Login Failure',
'Medium');
INSERT 0 1
postgres=# INSERT INTO fact_support_calls (customer_id, agent_id, time_id, issue_id,
call_duration_minutes,
postgres(# issue_resolved)
postgres-# VALUES (1, 1, 1, 1, 15, TRUE);
INSERT 0 1
```

Total Calls Handled by Each Agent

```
postgres=# SELECT a.agent_name, COUNT(f.call_id) AS total_calls FROM fact_support_calls f JOIN
dim_agent a ON f.agent_id = a.agent_id GROUP BY a.agent_name;
 agent_name  | total_calls
-------------+-------------
 Alice Smith |           1
(1 row)
```

2. Number of Issues Resolved vs Not Resolved

```
postgres=# SELECT issue_resolved, COUNT(*) AS call_count FROM fact_support_calls GROUP BY
issue_resolved;
 issue_resolved | call_count
----------------+------------
 t              |          1
(1 row)
```

3. Total Call Duration per Customer

```
postgres=# SELECT c.customer_name, SUM(f.call_duration_minutes) AS total_minutes FROM
fact_support_calls f JOIN dim_customer c ON f.customer_id = c.customer_id GROUP BY
c.customer_name;
 customer_name | total_minutes
---------------+---------------
 John Doe      |            15
(1 row)
```

4. Calls per Issue Type and Severity

```
postgres=# SELECT i.issue_type, i.severity_level, COUNT(*) AS total_calls FROM fact_support_calls f
JOIN dim_issue_type i ON f.issue_id = i.issue_id GROUP BY i.issue_type, i.severity_level;
  issue_type   | severity_level | total_calls
---------------+----------------+-------------
 Login Failure | Medium         |           1
(1 row)
```

5. Monthly Support Call Trend

```
postgres=# SELECT t.month, t.year, COUNT(*) AS total_calls FROM fact_support_calls f JOIN
dim_time t ON f.time_id = t.time_id GROUP BY t.month, t.year ORDER BY t.year, t.month;
 month | year | total_calls
-------+------+-------------
 July  | 2025 |           1
```

```
(1 row)
```

Snowflake Schema (DDL)

-- DIMENSION TABLES

-- Time Dimension

```
postgres=# CREATE TABLE dim_time1 ( time_id INT PRIMARY KEY, date DATE, month INT, quarter INT, year INT);
CREATE TABLE
```

-- Location Dimension

```
postgres=# CREATE TABLE dim_location (location_id INT PRIMARY KEY, city VARCHAR(50), state VARCHAR(50), country VARCHAR(50));
CREATE TABLE
```

-- Customer Dimension

```
postgres=# CREATE TABLE dim_customer1 (customer_id INT PRIMARY KEY, customer_name VARCHAR(100), email VARCHAR(100), phone VARCHAR(20), location_id INT, FOREIGN KEY (location_id) REFERENCES dim_location(location_id));
CREATE TABLE
```

-- Department Dimension

```
postgres=# CREATE TABLE dim_department (department_id INT PRIMARY KEY, department_name VARCHAR(100));
CREATE TABLE
```

-- Agent Dimension

```
postgres=# CREATE TABLE dim_agent1 (agent_id INT PRIMARY KEY, agent_name VARCHAR(100), department_id INT, FOREIGN KEY (department_id) REFERENCES dim_department(department_id));
CREATE TABLE
```

-- Issue Category Dimension

```
postgres=# CREATE TABLE dim_issue_category (issue_category_id INT PRIMARY KEY, issue_category_name VARCHAR(100));
CREATE TABLE
```

-- Issue Dimension

```
postgres=# CREATE TABLE dim_issue (issue_id INT PRIMARY KEY, issue_type VARCHAR(100), issue_category_id INT, FOREIGN KEY (issue_category_id) REFERENCES dim_issue_category(issue_category_id));
CREATE TABLE
```

-- FACT TABLE

```
postgres=# CREATE TABLE fact_support_calls1 (call_id INT PRIMARY KEY, customer_id INT, agent_id INT, issue_id INT, time_id INT, duration_minutes DECIMAL(5,2), resolution_status VARCHAR(50),  -- Resolved, Escalated, Pending FOREIGN KEY (customer_id) REFERENCES dim_customer(customer_id), FOREIGN KEY (agent_id) REFERENCES dim_agent(agent_id), FOREIGN KEY (issue_id) REFERENCES dim_issue(issue_id), FOREIGN KEY (time_id) REFERENCES dim_time(time_id));
CREATE TABLE
```

1. **Total Calls by Month**

```
postgres=# SELECT  t.year, t.month, COUNT(f.call_id) AS total_calls FROM fact_support_calls f JOIN dim_time t ON f.time_id = t.time_id GROUP BY t.year, t.month ORDER BY t.year, t.month;
 year | month | total_calls
------+-------+-------------
 2025 | July  |           1
```

```
(1 row)
```

2. Calls by Issue Category

```
postgres=# SELECT  ic.issue_category_name, COUNT(f.call_id) AS total_calls FROM fact_support_calls f JOIN dim_issue i ON f.issue_id = i.issue_id JOIN dim_issue_category ic ON i.issue_category_id = ic.issue_category_id GROUP BY ic.issue_category_name ORDER BY total_calls DESC;
 issue_category_name | total_calls
---------------------+-------------
      (0    rows)
```

3. Average Call Duration by Department

```
postgres=# SELECT  d.department_name, AVG(f.call_duration_minutes) AS avg_duration FROM fact_support_calls f  JOIN dim_agent a ON f.agent_id = a.agent_id  JOIN dim_department d ON a.department = d.department_name GROUP BY  d.department_name ORDER BY  avg_duration DESC;
 department_name | avg_duration
-----------------+--------------
(0 rows)
```

4. Resolution Status by Country

```
postgres=# SELECT  l.country,  f.issue_resolved,  COUNT(f.call_id) AS total FROM  fact_support_calls f JOIN dim_customer c ON f.customer_id = c.customer_id  JOIN dim_location l ON c.country = l.country GROUP BY l.country, f.issue_resolved ORDER BY l.country, total DESC;
 country | issue_resolved | total
---------+----------------+-------
(0 rows)
```

Example 2: Star & Snowflake Schema for Sales Data Warehouse

1. Create Tables:

```
postgres=# CREATE TABLE DimDate ( date_key INT PRIMARY KEY, full_date DATE, day_of_week VARCHAR(10), month VARCHAR(10), year INT );

CREATE TABLE

postgres=# CREATE TABLE DimProduct ( product_key INT PRIMARY KEY, product_name VARCHAR(100), category VARCHAR(50), brand VARCHAR(50) );

CREATE TABLE

postgres=# CREATE TABLE DimCustomer ( customer_key INT PRIMARY KEY, customer_name VARCHAR(100), city VARCHAR(50), country VARCHAR(50) );

CREATE TABLE

postgres=# CREATE TABLE FactSales ( sales_key SERIAL PRIMARY KEY, date_key INT REFERENCES DimDate(date_key), product_key INT REFERENCES DimProduct(product_key), customer_key INT REFERENCES DimCustomer(customer_key), quantity INT, price NUMERIC(10, 2) );

CREATE TABLE
```

2. Insert Sample Data:

```
postgres=# INSERT INTO DimDate (date_key, full_date, day_of_week, month, year) VALUES
(20230101, '2023-01-01', 'Sunday', 'January', 2023), (20230102, '2023-01-02', 'Monday', 'January',
2023);

INSERT 0 2

postgres=# INSERT INTO DimProduct (product_key, product_name, category, brand) VALUES (1,
'Laptop', 'Electronics', 'BrandA'), (2, 'Mouse', 'Electronics', 'BrandB');

INSERT 0 2

postgres=# INSERT INTO DimCustomer (customer_key, customer_name, city, country) VALUES (101,
'Alice', 'New York', 'USA'), (102, 'Bob', 'London', 'UK');

INSERT 0 2

postgres=# INSERT INTO FactSales (date_key, product_key, customer_key, quantity, price) VALUES
(20230101, 1, 101, 2, 1200.00), (20230102, 2, 102, 1, 25.00);

INSERT 0 2
```

3. Query Example:

```
postgres=# SELECT dd.year, dp.category, SUM(fs.quantity * fs.price) AS total_sales FROM FactSales fs
JOIN DimDate dd ON fs.date_key = dd.date_key JOIN DimProduct dp ON fs.product_key =
dp.product_key GROUP BY dd.year, dp.category;

 year |  category   | total_sales
------+-------------+-------------
 2023 | Electronics |     2425.00
(1 row)
```

Snowflake Schema Example

1. Create Tables:

```
postgres=# CREATE TABLE DimDate1 ( date_key INT PRIMARY KEY, full_date DATE, day_of_week
VARCHAR(10) );

CREATE TABLE

postgres=# CREATE TABLE DimMonth ( month_key INT PRIMARY KEY, month_name VARCHAR(10),
year INT );

CREATE TABLE

postgres=# CREATE TABLE DimProduct1 ( product_key INT PRIMARY KEY, product_name
VARCHAR(100) );

CREATE TABLE

postgres=# CREATE TABLE DimCategory1 ( category_key INT PRIMARY KEY, category_name
VARCHAR(50) );
```

```
CREATE TABLE

postgres=# CREATE TABLE DimBrand ( brand_key INT PRIMARY KEY, brand_name VARCHAR(50) );

CREATE TABLE


postgres=# CREATE TABLE ProductDetails ( product_key INT REFERENCES DimProduct1(product_key), category_key INT REFERENCES DimCategory1(category_key),

postgres(# brand_key INT REFERENCES DimBrand(brand_key), PRIMARY KEY (product_key) );

CREATE TABLE

postgres=# CREATE TABLE FactSalesSnowflake ( sales_key SERIAL PRIMARY KEY, date_key INT REFERENCES DimDate(date_key), month_key INT REFERENCES DimMonth(month_key), product_key INT REFERENCES DimProduct(product_key), quantity INT, price NUMERIC(10, 2) );

CREATE TABLE
```

2. Insert Sample Data:

```
postgres=# INSERT INTO DimDate (date_key1, full_date, day_of_week) VALUES (20230101, '2023-01-01', 'Sunday');

INSERT 0 1

postgres=# INSERT INTO DimMonth (month_key, month_name, year) VALUES (1, 'January', 2023);

INSERT 0 1

postgres=# INSERT INTO DimProduct1 (product_key, product_name) VALUES (1, 'Laptop');

INSERT 0 1

postgres=# INSERT INTO DimCategory1 (category_key, category_name) VALUES (10, 'Electronics');

INSERT 0 1

postgres=# INSERT INTO DimBrand (brand_key, brand_name) VALUES (100, 'BrandA');

INSERT 0 1

postgres=# INSERT INTO ProductDetails (product_key, category_key, brand_key) VALUES (1, 10, 100);

INSERT 0 1

postgres=# INSERT INTO FactSalesSnowflake (date_key, month_key, product_key, quantity, price) VALUES (20230101, 1, 1, 2, 1200.00);

INSERT 0 1
```

3. Query Example:

```
postgres=# SELECT dm.year, dc.category_name, SUM(fss.quantity * fss.price) AS total_sales FROM FactSalesSnowflake fss JOIN DimMonth dm ON fss.month_key = dm.month_key JOIN DimProduct1 dp ON fss.product_key = dp.product_key JOIN ProductDetails pd ON dp.product_key =
pd.product_key JOIN DimCategory1 dc ON pd.category_key = dc.category_key GROUP BY dm.year,
dc.category_name;

 year | category_name | total_sales
------+---------------+-------------
 2023 | Electronics   |     2400.00
(1 row)
```