CREATE table Time\_dw5 (

time\_id int PRIMARY KEY,

day DATE NOT NULL,

month varchar(255) NOT NULL,

qt varchar(255) NOT NULL,

yr varchar(255) NOT NULL

);

CREATE TABLE Product\_dw (

prod\_id INT PRIMARY KEY,

prod\_name VARCHAR(60) NOT NULL,

prod\_category VARCHAR(255) NOT NULL,

brand\_name VARCHAR(255) NOT NULL,

suppl\_name VARCHAR(255) NOT NULL,

prod\_price INT

);

CREATE TABLE Location\_dw (

loc\_id INT PRIMARY KEY,

street VARCHAR(60) NOT NULL,

city VARCHAR(255) NOT NULL,

state VARCHAR(255) NOT NULL,

country VARCHAR(255) NOT NULL

);

CREATE TABLE Fact\_sales (

prod\_id INT REFERENCES Product\_dw(prod\_id),

time\_id INT REFERENCES Time\_dw5(time\_id),

loc\_id INT REFERENCES Location\_dw(loc\_id),

Number\_of\_unit\_sold INT NOT NULL,

Total\_sales INT NOT NULL

);

INSERT INTO Time\_dw5(time\_id, day, month, qt, yr) VALUES

(101, '2021-01-17', 'January', 'Q1', '2021'),

(102, '2021-02-14', 'February', 'Q1', '2021'),

(103, '2021-05-21', 'May', 'Q2', '2021');

INSERT INTO Product\_dw(prod\_id, prod\_name, prod\_category, brand\_name, suppl\_name, prod\_price) VALUES

(1, 'Rice', 'Grocery', 'Dawat', 'Ramesh', 140),

(2, 'Sugar', 'Grocery', 'Dawat', 'Ramesh', 50),

(3, 'Kurta', 'Cloth', 'Max', 'Lila', 500),

(4, 'Jacket', 'Cloth', 'Max', 'Lila', 700),

(5, 'Biscuit', 'Snacks', 'Britannia', 'Dilip', 30);

INSERT INTO Location\_dw(loc\_id, street, city, state, country) VALUES

(201, 'MLROAD', 'Mumbai', 'Maharashtra', 'India'),

(202, 'AIROAD', 'Mumbai', 'Maharashtra', 'India'),

(203, 'BIROAD', 'Kolkata', 'West Bengal', 'India'),

(204, 'DBROAD', 'Kolkata', 'West Bengal', 'India'),

(205, 'BROAD', 'Delhi', 'Delhi', 'India');

INSERT INTO Fact\_sales(prod\_id, time\_id, loc\_id, Number\_of\_unit\_sold, Total\_sales) VALUES

(1, 101, 201, 400, 90000),

(1, 102, 201, 200, 40000),

(2, 101, 202, 300, 15000),

(3, 103, 203, 150, 75000),

(4, 103, 204, 100, 70000);

Dice

SELECT prod\_name,Total\_sales,day

FROM (Fact\_sales

INNER JOIN Product\_dw ON Product\_dw.prod\_id = Fact\_sales.prod\_id)

JOIN Time\_dw5 ON Time\_dw5.time\_id = Fact\_sales.time\_id;

Slice

SELECT prod\_name,Total\_sales

FROM Fact\_sales

INNER JOIN Product\_dw ON Product\_dw.prod\_id = Fact\_sales.prod\_id;

Rollup

SELECT Time\_dw5.yr, SUM(Total\_sales)

FROM Fact\_sales

JOIN Product\_dw ON Fact\_sales.prod\_id = Product\_dw.prod\_id

JOIN Time\_dw5 ON Fact\_sales.time\_id = Time\_dw5.time\_id

GROUP BY Time\_dw5.yr;

Drilldown

SELECT Time\_dw5.yr, Time\_dw5.day, SUM(Total\_sales)

FROM Fact\_sales

JOIN Product\_dw ON Fact\_sales.prod\_id = Product\_dw.prod\_id

JOIN Time\_dw5 ON Fact\_sales.time\_id = Time\_dw5.time\_id

GROUP BY Time\_dw5.yr, Time\_dw5.day;

Yes, the queries you've written are correct for the respective operations in OLAP (Online Analytical Processing) systems, such as slicing, dicing, rollup, and drilldown. Here's a breakdown of each:

### 1. \*\*Dice\*\*:

```sql

SELECT prod\_name, Total\_sales, day

FROM (Fact\_sales

INNER JOIN Product\_dw ON Product\_dw.prod\_id = Fact\_sales.prod\_id)

JOIN Time\_dw5 ON Time\_dw5.time\_id = Fact\_sales.time\_id;

```

- \*\*Explanation\*\*: The \*\*dice\*\* operation is used to query data across multiple dimensions. This query retrieves product names, total sales, and the day of the transaction by joining the `Fact\_sales`, `Product\_dw`, and `Time\_dw5` tables. This effectively slices across the product and time dimensions to get a more focused dataset.

### 2. \*\*Slice\*\*:

```sql

SELECT prod\_name, Total\_sales

FROM Fact\_sales

INNER JOIN Product\_dw ON Product\_dw.prod\_id = Fact\_sales.prod\_id;

```

- \*\*Explanation\*\*: The \*\*slice\*\* operation selects data based on a single dimension. Here, you're selecting product names and total sales. This limits the scope to those two columns without considering the time dimension or other details.

### 3. \*\*Rollup\*\*:

```sql

SELECT Time\_dw5.yr, SUM(Total\_sales)

FROM Fact\_sales

JOIN Product\_dw ON Fact\_sales.prod\_id = Product\_dw.prod\_id

JOIN Time\_dw5 ON Fact\_sales.time\_id = Time\_dw5.time\_id

GROUP BY Time\_dw5.yr;

```

- \*\*Explanation\*\*: The \*\*rollup\*\* operation aggregates data, usually at a higher level. This query groups the total sales by year, aggregating the `Total\_sales` for each year (`yr`). It's useful for summarizing data at the year level.

### 4. \*\*Drilldown\*\*:

```sql

SELECT Time\_dw5.yr, Time\_dw5.day, SUM(Total\_sales)

FROM Fact\_sales

JOIN Product\_dw ON Fact\_sales.prod\_id = Product\_dw.prod\_id

JOIN Time\_dw5 ON Fact\_sales.time\_id = Time\_dw5.time\_id

GROUP BY Time\_dw5.yr, Time\_dw5.day;

```

- \*\*Explanation\*\*: The \*\*drilldown\*\* operation gives you a more detailed view by breaking down the aggregated data. This query shows total sales for each day within each year (`yr`, `day`). It goes deeper into the data compared to the rollup, which only aggregated by year.

### Summary:

- \*\*Dice\*\*: Queries data across multiple dimensions (product, total sales, day).

- \*\*Slice\*\*: Queries data based on a single or limited set of columns (product, total sales).

- \*\*Rollup\*\*: Aggregates data at a higher level (total sales per year).

- \*\*Drilldown\*\*: Provides a detailed breakdown of the data (total sales per day within each year).

Your queries are accurate for these operations!

Pivot

SELECT

Time\_dw5.yr,

SUM(CASE WHEN Product\_dw.prod\_name = 'Rice' THEN Total\_sales ELSE 0 END) AS Rice,

SUM(CASE WHEN Product\_dw.prod\_name = 'Sugar' THEN Total\_sales ELSE 0 END) AS Sugar,

SUM(CASE WHEN Product\_dw.prod\_name = 'Kurta' THEN Total\_sales ELSE 0 END) AS Kurta,

SUM(CASE WHEN Product\_dw.prod\_name = 'Jacket' THEN Total\_sales ELSE 0 END) AS Jacket,

SUM(CASE WHEN Product\_dw.prod\_name = 'Biscuit' THEN Total\_sales ELSE 0 END) AS Biscuit

FROM

Fact\_sales

JOIN

Product\_dw ON Fact\_sales.prod\_id = Product\_dw.prod\_id

JOIN

Time\_dw5 ON Fact\_sales.time\_id = Time\_dw5.time\_id

GROUP BY

Time\_dw5.yr;

pip install pandas numpy matplotlib scikit-learn

https://github.com/suvidhiiiiii/dwm