Pizza Sales Data Analysis

```
CREATE TABLE order_details

(

 order_details_id INT PRIMARY KEY,

 order_id INT REFERENCES orders(order_id),

 pizza_id VARCHAR(100) REFERENCES pizzas(pizza_id),

 quantity INT

);

---------------------------------------------------------------

CREATE TABLE orders (

    order_id INT PRIMARY KEY,

    order_date DATE,

    order_time TIME

);

---------------------------------------------------------------

CREATE TABLE pizza_types

(

 pizza_type_id VARCHAR(50)  PRIMARY KEY,

 name VARCHAR(100),

 category VARCHAR(80),

 ingredients VARCHAR(200)

);

---------------------------------------------------------------
```

```
CREATE TABLE pizzas
(
 pizza_id VARCHAR(80) PRIMARY KEY,
 pizza_type_id VARCHAR(50)  REFERENCES pizza_types (pizza_type_id),
 size VARCHAR(20),
 price FLOAT
);
SELECT * FROM pizzas;
-----------------------------------------------------------------
```

Q1.Monthly Revenue Trend

```
SELECT
            TO_CHAR(o.order_date, 'Mon') AS month,
             ROUND(SUM(od.quantity * p.price)::numeric,2) AS revenue
FROM orders o
JOIN order_details od
ON
o.order_id = od.order_id
JOIN
pizzas p
ON
od.pizza_id = p.pizza_id
GROUP BY 1
ORDER BY 2 DESC;
--------------------------------------------------------------------------------------------------------
```

Q2.Top 5 Best-Selling Pizza Types

```
SELECT
        pt.name,
        SUM(od.quantity) AS total_sold
FROM pizza_types pt
JOIN pizzas p
ON
pt.pizza_type_id = p.pizza_type_id
JOIN order_details od
ON
od.pizza_id = p.pizza_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```

Q3.Peak Order Hours

```
SELECT
        EXTRACT(HOUR FROM order_time) AS order_hour,
        COUNT(*) AS order_count
FROM orders
GROUP BY 1
ORDER BY 2 DESC;
```

Q4.Most Demanded Ingredients

```
SELECT
            TRIM(UNNEST(STRING_TO_ARRAY(ingredients, ','))) AS ingredient,
             COUNT(*) AS frequency
FROM pizza_types
GROUP BY 1
ORDER BY 2 DESC;
```

Q5.Daily Order Volume

```
SELECT
            order_date,
             COUNT(DISTINCT order_id) AS total_orders
FROM orders
GROUP BY 1;
```

Some Other Query

Q1.Retrieve the total number of orders placed.

```
SELECT
          COUNT(*) AS total_order
FROM orders;
```

Q2.Calculate the total revenue generated from pizza sales.

```
SELECT
     ROUND(SUM(od.quantity * p.price)::numeric,2) AS total_revenue
FROM order_details od
```

```
JOIN pizzas p
ON
od.pizza_id = p.pizza_id;
```

Q3.Identify the highest price pizzas.

```
SELECT
            pt.name,
            p.price
FROM pizza_types pt
JOIN pizzas p
ON
pt.pizza_type_id = p.pizza_type_id
ORDER BY 2 DESC
LIMIT 1;
```

Q4.Identify the most common pizza size ordered.

```
SELECT
            p.size,
            COUNT(od.order_details_id) AS total_pizza_size
FROM  pizzas p
JOIN order_details od
ON
p.pizza_id = od.pizza_id
GROUP BY 1
ORDER BY 2 DESC;
```

Q5.List the top 5 most ordered pizza types along with their quantities.

```
SELECT
        pt.name,
        SUM(od.quantity) AS total_qty_sold
FROM pizza_types pt
JOIN pizzas p
ON
pt.pizza_type_id = p.pizza_type_id
JOIN order_details od
ON od.pizza_id = p.pizza_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```

Q6.Join the necessary tables to find the total quantity of each pizza category ordered.

```
SELECT
        pt.category,
        SUM(od.quantity) AS total_quantity
FROM pizza_types pt
JOIN pizzas p
ON
pt.pizza_type_id = p.pizza_type_id
JOIN order_details od
```

```
ON
od.pizza_id = p.pizza_id
GROUP BY 1;
```

Q7.Determine the distribution of orders by hour of the day.

```
SELECT
        EXTRACT(HOUR FROM order_time) AS hour,
        TO_CHAR(order_date, 'Day') AS day,
        COUNT(order_id) AS order_count
FROM orders
GROUP BY 1,2
ORDER BY 1;
```

Q8.Join relevant tables to find the category-wise distribution of pizzas.

```
SELECT
        category,
        COUNT(pizza_type_id) AS total_count
FROM pizza_types
GROUP BY 1
ORDER BY 2 DESC;
```

Q9.Group the orders by date and calculate the average number of pizzas ordered per day.

```
SELECT
    ROUND(AVG(total_quantity)::numeric,2) AS avg_quantity
FROM
  (
   SELECT
                o.order_date,
                SUM(od.quantity) AS total_quantity
FROM orders o
JOIN order_details od
ON
o.order_id = od.order_id
GROUP BY 1
   );
------------------------------------------------------------
------OR with CTE------

WITH CTE AS
(
   SELECT
                o.order_date,
                SUM(od.quantity) AS total_quantity
FROM orders o
JOIN order_details od
ON
o.order_id = od.order_id
GROUP BY 1

)

SELECT

    ROUND(AVG(total_quantity)::numeric,2) AS avg_quantity

FROM CTE;
```

Q10.Determine the top 3 most ordered pizza types based on revenue.

```
SELECT

            pt.name,

            SUM(od.quantity * p.price) AS revenue

FROM pizza_types pt

JOIN pizzas p

ON

pt.pizza_type_id = p.pizza_type_id

JOIN order_details od

ON

od.pizza_id = p.pizza_id

GROUP BY 1

ORDER BY 2 DESC

LIMIT 3;
```

Advanced Questions

Q11.Calculate the percentage contribution of each pizza type to total revenue.

```
SELECT
            pt.name,
            SUM(od.quantity * p.price) AS revenue,
                ROUND
                          (
                           (SUM(od.quantity * p.price)*100.0/
                            (SELECT SUM(od.quantity * p.price)
                            FROM order_details od
                           JOIN pizzas p
                           ON
                           od.pizza_id = p.pizza_id))::numeric,2) AS revenue_percentage
FROM pizza_types pt
JOIN pizzas p
ON
pt.pizza_type_id = p.pizza_type_id
JOIN order_details od
ON od.pizza_id = p.pizza_id
GROUP BY pt.name;
```

Q12.Analyze the cumulative revenue generated over time.

```
SELECT hour, day, SUM(revenue) OVER(ORDER BY hour) AS cumulative_revenue
FROM
(
SELECT
        EXTRACT(HOUR FROM o.order_time) AS hour,
        TO_CHAR(order_date, 'Day') AS day,
        SUM(od.quantity * p.price) AS revenue
FROM orders o
JOIN order_details od
ON
o.order_id = od.order_id
JOIN pizzas p
ON
p.pizza_id = od.pizza_id
GROUP BY 1,2
);
```

Q13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
SELECT category,name,revenue
FROM(
SELECT
            category,name,revenue,
            RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS rn
            FROM
            (
SELECT
     pt.category,
            pt.name,
            SUM(od.quantity * p.price) AS revenue
FROM pizza_types pt
JOIN pizzas p
ON
pt.pizza_type_id = p.pizza_type_id
JOIN order_details od
ON
od.pizza_id = p.pizza_id
GROUP BY 1,2)
)
WHERE rn >=3;
```