SQL for Data Analysis (Assignment)

Conduct a comprehensive sales performance analysis for a pizza sale using SQL queries.

Ratnesh Satyarthi

Business Objective : -

- The objective is to gain insights into sales trends, revenue distribution, and top performing products to inform strategic decision-making and optimize sales strategies.

  - Extract and aggregate sales data from the pizza sales dataset to calculate key metrics such as total sales revenue, average order value, Total Orders.
  - Analyse sales trends over time (hourly, daily, monthly) to identify peak sales periods and seasonal variations.
  - Calculate sales performance metrics for individual products, including best-selling pizzas, popular pizza sizes, and revenue contribution by product category.

1. Orders Trend: -

- To measure and analyze various aspects of pizza sales data to assess sales performance.
    1. Total no. of orders placed?
    2. Distribution of orders per month or per day or per hour?
    3. Total Quantity of all pizzas ordered based on size?
    4. Maximum quantity of pizzas ordered based on size.
    5. How much pizzas were ordered from each category?
    6. During which time most no of pizza sales happened?
    7. Aggregated sales Hour wise.
    8. Group the orders by date and calculate the average number of pizzas ordered per day.

2. Sales Trend: -

- To analyze how sales of pizza products vary seasonally or over specific time period.

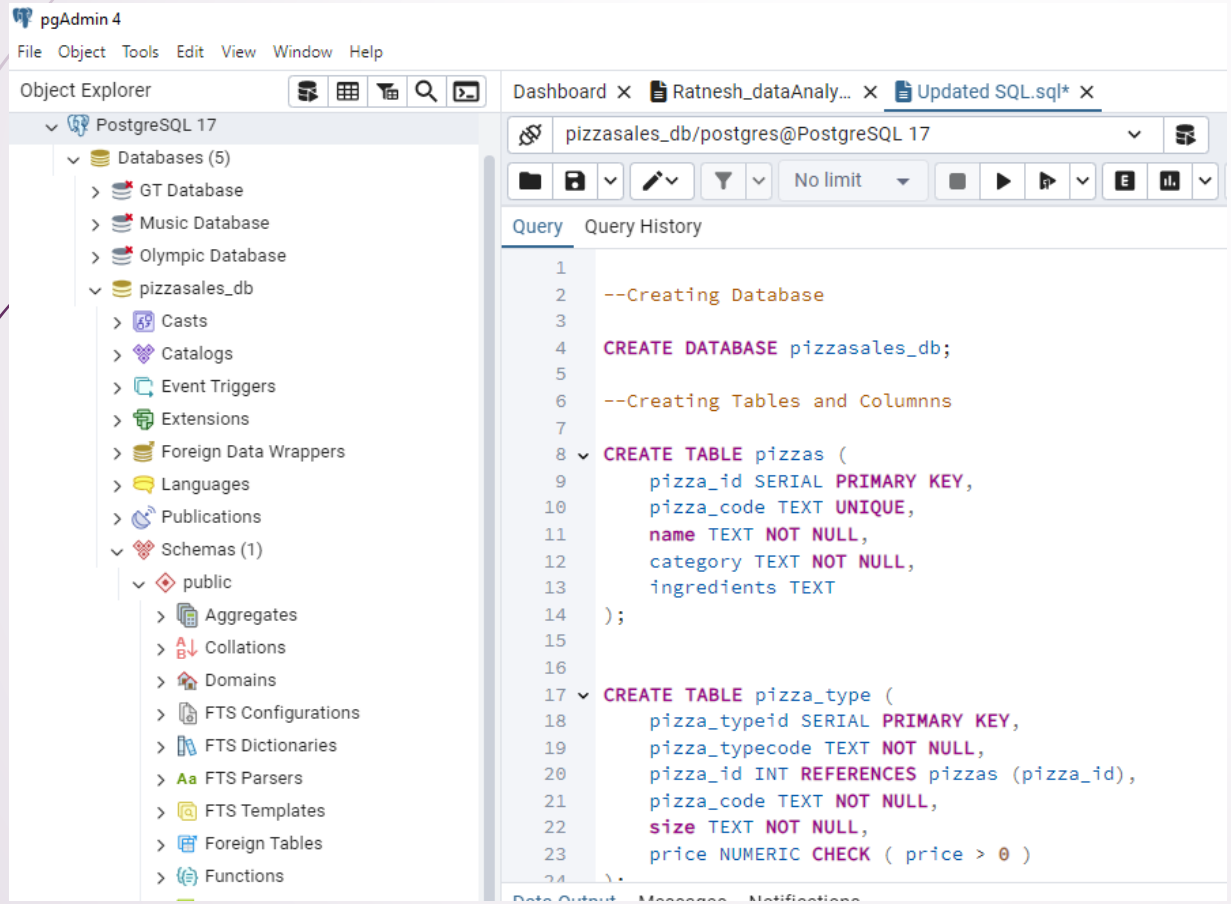
1. Total revenue generated from pizza sales?
2. Revenue based on each pizza type/ size ordered
3. Total Revenue generated per hour or per day
4. For which date the pizza sales were maximum ?
5. Top 3 most ordered pizza types based on revenue
6. How much Revenue was generated from sales of pizzas from each category
7. Total Revenue generated per month by sales of all pizzas
8. TOP 5 months which generated the most revenue by sales of pizza

3. Product Popularity: -

- To measure the popularity and demand for different pizza products.
  1. Most expensive and Least expensive pizza ?
  2. Total Quantity of each pizza type ordered based on size ?
  3. Type of pizza based on size which was ordered the most ?
  4. Most ordered pizza and how much quantity ?
  5. Find How many pizzas which have price above than average price or below than average price ?
  6. Percentage contribution of each pizza type to Revenue ?

Database used: -

- Postrgess SQL – version 8.11



Tables: -

Pizzas	Pizza Types	Orders	Order Details
• Pizza id (PK) • Pizza code • Name • Category • Ingredient	• Pizza type id (PK) • Pizza type code • Pizza id (FK) • Pizza code • Size • Price	• Order id (PK) • Date • Time	• Order details id (PK) • Order id (FK) • Pizza type code • Quantity • Pizza type id

PK – Primary Key

FK – Foreign Key

DDL Queries used: -

- CREATE: Creates a new table or database.
  - Eg. - `CREATE DATABASE pizzasales_db;`
- ALTER: Modifies an existing database object.
  - Eg. - `ALTER TABLE pizza_type ADD COLUMN pizza_id INT REFERENCES pizzas (pizza_id);`
- DROP: Deletes an entire table, database, or other objects.
  - Eg. - `DROP TABLE pizza_types;`
- TRUNCATE: Removes all records from a table, deleting the space allocated for the records.
  - Eg. - `TRUNCATE TABLE order_details;`

DML Queries used: -

- SELECT: Retrieves data from the database.
  - Eg. - `SELECT * FROM pizzas;`
- INSERT: Adds new data to a table.
  - Eg. - `INSERT INTO pizzas ( pizza_code, name, category, ingredients) VALUES('bbq_ckn', 'The Barbecue Chicken Pizza', 'Chicken', 'Barbecued Chicken, Red Peppers, Green Peppers, Tomatoes, Red Onions, Barbecue Sauce');`
- UPDATE: Modifies existing data within a table.
  - Eg. - `UPDATE pizza_type SET pizza_id = 32 WHERE pizza_code = 'veggie_veg';`
- DELETE: Removes data from a table.
  - Eg. - `DELETE pizza_type WHERE pizza_code = 'veggie_veg';`

Create Database & Table structure: -

```sql
CREATE DATABASE pizza_salesdb;
```

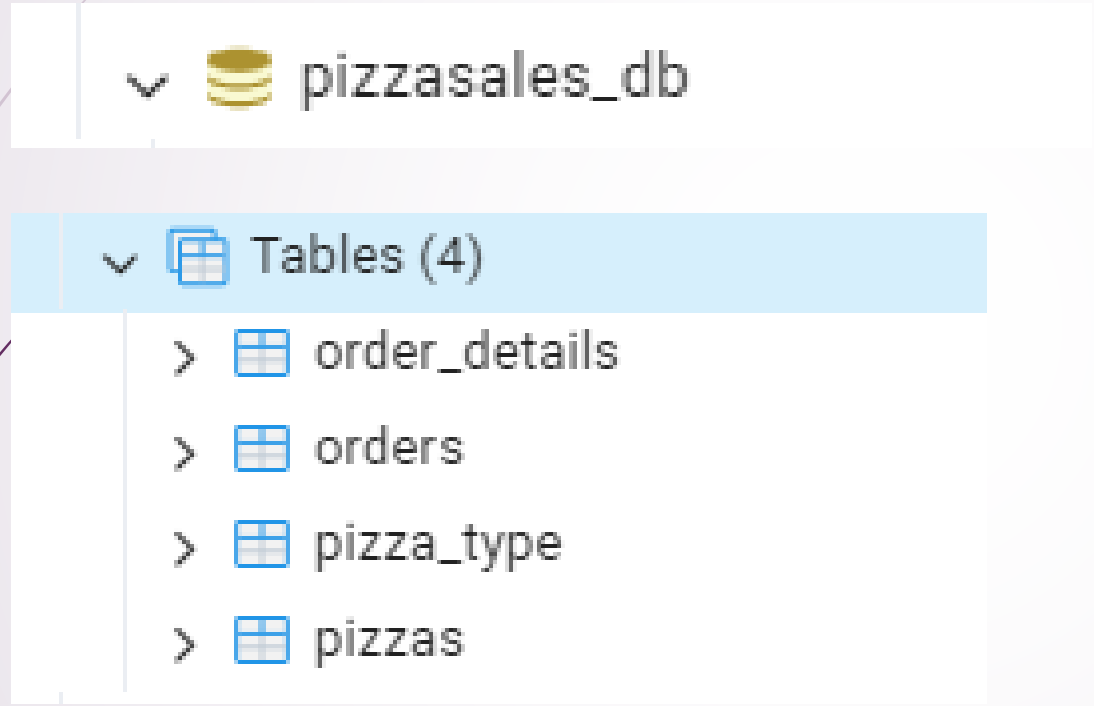
```sql
CREATE TABLE pizzas (
    pizza_id SERIAL PRIMARY KEY,
    pizza_code TEXT UNIQUE,
    name TEXT NOT NULL,
    category TEXT NOT NULL,
    ingredients TEXT
);
```

```sql
CREATE TABLE pizza_type (
    pizza_typeid SERIAL PRIMARY KEY,
    pizza_typecode TEXT NOT NULL,
    pizza_id INT REFERENCES pizzas (pizza_id),
    pizza_code TEXT NOT NULL,
    size TEXT NOT NULL,
    price NUMERIC CHECK ( price > 0 )
);
```

```sql
CREATE TABLE orders (
    order_id SERIAL PRIMARY KEY,
    date DATE NOT NULL,
    time TIME NOT NULL
);
```

```sql
CREATE TABLE order_details (
    order_details_id SERIAL PRIMARY KEY,
    order_id INT NOT NULL,
    pizza_typecode TEXT NOT NULL,
    quantity INT CHECK ( quantity > 0 ),
    pizza_typeid INT REFERENCES pizza_type (pizza_typeid),
    FOREIGN KEY (order_id) REFERENCES orders (order_id)
);
```

Tables & Database: -



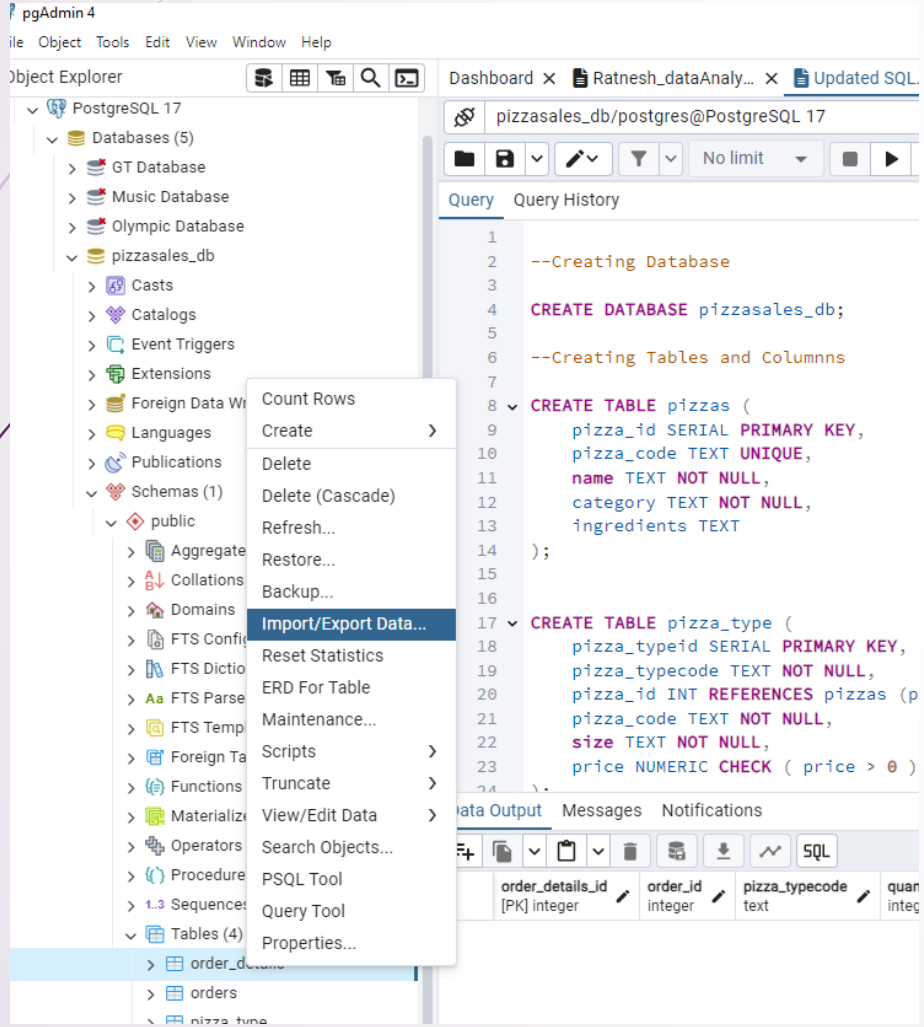
Insert Records in Tables: -

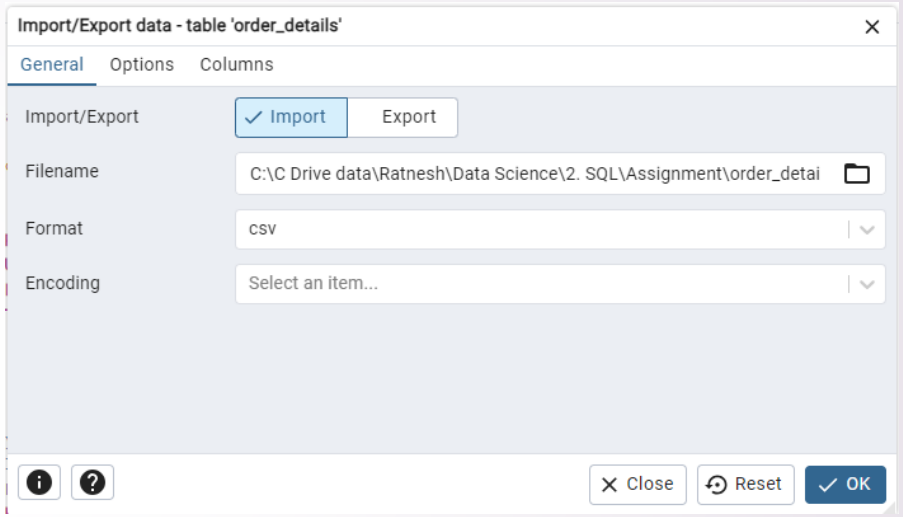
```
INSERT INTO pizzas (
    pizza_code,
    name,
    category,
    ingredients
) VALUES
('bbq_ckn',     'The Barbecue Chicken Pizza',   'Chicken',  'Barbecued Chicken, Red Peppers, Green Peppers, Tomatoes, Red Onions, Barbecue Sauce'),
('cali_ckn',    'The California Chicken Pizza',     'Chicken',  'Chicken, Artichoke, Spinach, Garlic, Jalapeno Peppers, Fontina Cheese, Gouda Cheese'),
('ckn_alfredo', 'The Chicken Alfredo Pizza',    'Chicken',  'Chicken, Red Onions, Red Peppers, Mushrooms, Asiago Cheese, Alfredo Sauce'),
('ckn_pesto',   'The Chicken Pesto Pizza', 'Chicken',  'Chicken, Tomatoes, Red Peppers, Spinach, Garlic, Pesto Sauce'),
```

```
INSERT INTO pizza_type (
    pizza_typecode,
    pizza_id,
    pizza_code,
    size,
    price
) VALUES
('bbq_ckn_s',   '1',    'bbq_ckn',  'S',    '255'),
('bbq_ckn_m',   '1',    'bbq_ckn',  'M',    '335'),
('bbq_ckn_l',   '1',    'bbq_ckn',  'L',    '415'),
('cali_ckn_s',  '2',    'cali_ckn', 'S',    '255'),
('cali_ckn_m',  '2',    'cali_ckn', 'M',    '335'),
('cali_ckn_l',  '2',    'cali_ckn', 'L',    '415'),
```

```
INSERT INTO orders (date,   time)
VALUES
('01-01-2015',  '11:38:36'),
('01-01-2015',  '11:57:40'),
('01-01-2015',  '12:12:28'),
('01-01-2015',  '12:16:31'),
('01-01-2015',  '12:21:30'),
('01-01-2015',  '12:29:36'),
('01-01-2015',  '12:50:37'),
('01-01-2015',  '12:51:37'),
('01-01-2015',  '12:52:01'),
```

Importing Records in Tables: -





Orders queris: -

```sql
--ORDERS

/* Total no. of orders placed? */
SELECT COUNT(*) as Total_Orders
    FROM orders;

/* Total orders per day */

SELECT
    date,
    COUNT(order_id) as total_orders
    FROM orders
    GROUP BY date
    ORDER BY date;

/* Total orders per hour */

SELECT
    EXTRACT ('hour' FROM time) as hours,
    COUNT(order_id) as total_orders
    FROM orders
    GROUP BY hours
    ORDER BY hours;

/* Total Quantity of all pizzas ordered based on size */

SELECT
    t.size,
    SUM (d.quantity) as total_quantity
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    GROUP BY t.size
    ORDER BY total_quantity DESC;
```

```sql
/* Maximum quantity of pizzas ordered based on size */

SELECT
    t.size,
    SUM (d.quantity) as total_quantity
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    GROUP BY t.size
    ORDER BY total_quantity DESC
    LIMIT 1;

/* How much pizzas were ordered from each category? */

SELECT
    p.category,
    SUM (d.quantity) as total_quantity
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON t.pizza_typeid = d.pizza_typeid
    GROUP BY p.category
    ORDER BY total_quantity DESC;

/* During which time most no of pizza sales happened? */

SELECT
    EXTRACT ('hour' FROM time) as hours,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY hours
    ORDER BY total_revenue DESC
    LIMIT 1;
```

Orders queries: -

```sql
/* Determine the distribution of orders by hour of the day */

SELECT
    EXTRACT ('hour' FROM time) as hours,
    COUNT (d.quantity ) as total_quantity
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY hours
    ORDER BY hours;

/* During which hour most no of pizza were ordered? */

SELECT
    EXTRACT ('hour' FROM time) as hours,
    COUNT (d.quantity ) as total_quantity
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY hours
    ORDER BY total_quantity DESC
    LIMIT 1;
```

```sql
/* Group the orders by date and calculate the average number of pizzas ordered per day. */

SELECT
    ROUND(AVG(total_quantity), 0) as avg_pizza_ordered_per_day
    FROM
        (
            SELECT
                o.date,
                SUM(d.quantity) as total_quantity
            FROM
                orders as o
            JOIN order_details as d
                ON o.order_id = d.order_id
            GROUP BY o.date
            ORDER BY o.date
        ) as order_quantity;

--SALES
```

Output | Messages | Notifications

avg_pizza_ordered_per_day numeric
138

Sales queries: -

```sql
--SALES

/* Total Revenue from all pizza sales */

SELECT
    SUM (t.price * d.quantity) as total_revenue
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid;

/* Total Revenue generated based on each pizza size */

SELECT
    t.size,
    --d.pizza_typecode,
    SUM (t.price * d.quantity) as revenue
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    GROUP BY t.size
    ORDER BY revenue DESC;

/* Total Revenue based on each pizza type ordered of different size */

SELECT
    t.pizza_typeid,
    d.pizza_typecode,
    SUM (t.price * d.quantity) as revenue
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    GROUP BY t.pizza_typeid, d.pizza_typecode
    ORDER BY t.pizza_typeid;
```

```sql
/* Total Revenue generated per hour */

SELECT
    EXTRACT ('hour' FROM time) as hours,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY hours
    ORDER BY hours;

/* Total Revenue generated per day by sale of pizzas */

SELECT
    date,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY date
    ORDER BY date;
```

Sales queries: -

```sql
SELECT
    date,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY date
    ORDER BY date DESC
    LIMIT 1;

/* Top 3 most ordered pizza types based on revenue */

SELECT
    p.name,
    --t.pizza_code,
    SUM (t.price * d.quantity) as revenue
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    INNER JOIN pizzas as p
        ON p.pizza_id = t.pizza_id
    GROUP BY p.name,t.pizza_code
    ORDER BY revenue DESC
    LIMIT 3;
```

```sql
/* How much Revenue was generated from sales of pizzas from each category ? */
SELECT
    p.category,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON t.pizza_typeid = d.pizza_typeid
    GROUP BY p.category
    ORDER BY total_revenue DESC;

/* Total Revenue generated per month by sales of all pizzas */
SELECT
    EXTRACT (MONTH FROM date) as months,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY months
    ORDER BY months;

/* TOP 5 months which generated the most revenue by sales of pizza */
SELECT
    EXTRACT (MONTH FROM date) as months,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY months
    ORDER BY total_revenue DESC
    LIMIT 5;
```

Product queries: -

```sql
/* Most ordered pizza and how much quantity ? */

SELECT
    p.name,
    t.pizza_code,
    SUM (d.quantity) as total_quantity
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    INNER JOIN pizzas as p
        ON p.pizza_code = t.pizza_code
    GROUP BY p.name,t.pizza_code
    ORDER BY total_quantity DESC
    LIMIT 1;

/* Find which pizza price is above than average price or below than average price */

SELECT ROUND(AVG(price),2) FROM pizza_type;

SELECT
    pizza_code,
    size,
    price,
        CASE
            WHEN price > (SELECT ROUND(AVG(price),2) FROM pizza_type)
                THEN 'Price above than Average'
            WHEN price < (SELECT ROUND(AVG(price),2) FROM pizza_type)
                THEN 'Price below than Average'
            ELSE 'Price same as Average'
        END as Product_price
    FROM pizza_type;
```

```sql
--PRODUCT

/* Most expensive pizza */

SELECT *
    FROM pizza_type
    WHERE price IN (
        SELECT MAX(price) FROM pizza_type);

/* Least expensive pizza */

SELECT *
    FROM pizza_type
    WHERE price IN (
        SELECT MIN(price) FROM pizza_type);

/* Total Quantity of each pizza type ordered based on size */

SELECT pizza_typeid,
    pizza_typecode,
    SUM (quantity) as total_quantity
    FROM order_details
    GROUP BY pizza_typeid, pizza_typecode
    ORDER BY pizza_typeid;

/* Type of pizza based on size which was ordered the most? */

SELECT
    p.name,
    t.size,
    d.pizza_typecode,
    SUM (d.quantity) as total_quantity
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON t.pizza_typeid = d.pizza_typeid
    GROUP BY p.name,t.size,d.pizza_typecode
    ORDER BY total_quantity DESC
    LIMIT 1;
```

Output: -

```sql
/* Percentage contribution of each pizza type to Revenue. */
WITH cte_revenue as (
        SELECT
            p.category,
            SUM (d.quantity * t.price) as revenue
            FROM pizzas as p
                INNER JOIN pizza_type as t
                    ON p.pizza_id = t.pizza_id
                INNER JOIN order_details as d
                    ON t.pizza_typeid = d.pizza_typeid
            GROUP BY p.category
            ORDER BY revenue DESC
    ),

    cte_total_revenue as (
        SELECT
            SUM (t.price * d.quantity) as total_revenue
        FROM pizza_type as t
        INNER JOIN order_details as d
            ON t.pizza_typeid = d.pizza_typeid
    )

SELECT
    cr.category,
    cr.revenue,
    ctr.total_revenue,
    ROUND((cr.revenue / ctr.total_revenue),2) * 100 as percentage_of_total_revenue
    FROM cte_revenue as cr,cte_total_revenue as ctr;
```

Output | Messages | Notifications

category text	revenue numeric	total_revenue numeric	percentage_of_total_revenue numeric
Classic	4401062	16357201	27.00
Supreme	4163940	16357201	25.00
Chicken	3918390	16357201	24.00
Veggie	3873809	16357201	24.00

Output: -

```sql
/* Find How many pizzas which have price above than average price or below than average price. */

SELECT
    COUNT(pizza_typeid),
        CASE
            WHEN price > (SELECT ROUND(AVG(price),2) FROM pizza_type)
                THEN 'Price above than Average'
            WHEN price < (SELECT ROUND(AVG(price),2) FROM pizza_type)
                THEN 'Price below than Average'
            ELSE 'Price same as Average'
        END as Product_price
    FROM pizza_type
    GROUP BY Product_price;


/* Percentage contribution of each pizza type to Revenue  */
```

a Output | Messages | Notifications

SQL

count bigint	product_price text
47	Price above than Average
49	Price below than Average

Output: -

```
70507  /* Most ordered pizza and how much quantity ? */
70508
70509  SELECT
70510      p.name,
70511      t.pizza_code,
70512      SUM (d.quantity) as total_quantity
70513      FROM pizza_type as t
70514      INNER JOIN order_details as d
70515          ON t.pizza_typeid = d.pizza_typeid
70516      INNER JOIN pizzas as p
70517          ON p.pizza_code = t.pizza_code
70518      GROUP BY p.name,t.pizza_code
70519      ORDER BY total_quantity DESC
70520      LIMIT 1;
```

Data Output | Messages | Notifications

	name text	pizza_code text	total_quantity bigint
1	The Classic Deluxe Pizza	classic_dlx	2453

```
70493  SELECT
70494      p.name,
70495      t.size,
70496      d.pizza_typecode,
70497      SUM (d.quantity) as total_quantity
70498      FROM pizzas as p
70499          INNER JOIN pizza_type as t
70500              ON p.pizza_id = t.pizza_id
70501          INNER JOIN order_details as d
70502              ON t.pizza_typeid = d.pizza_typeid
70503      GROUP BY p.name,t.size,d.pizza_typecode
70504      ORDER BY total_quantity DESC
70505      LIMIT 1;
70506
```

Data Output | Messages | Notifications

	name text	size text	pizza_typecode text	total_quantity bigint
1	The Big Meat Pizza	S	big_meat_s	1914

Output: -

```
70180  /* Total no. of orders placed? */
70181  SELECT COUNT(*) as Total_Orders
70182      FROM orders;
70183
70184  /* Total orders per day */
```

Data Output | Messages | Notifications

	total_orders bigint
1	21350

```
70186  SELECT
70187      date,
70188      COUNT(order_id) as total_orders
70189      FROM orders
70190      GROUP BY date
70191      ORDER BY date;
70192
```

Data Output | Messages | Notifications

	date date	total_orders bigint
1	2015-01-01	69
2	2015-01-02	67
3	2015-01-03	66

```
0466  --PRODUCT
0467
0468  /* Most expensive pizza */
0469
0470  SELECT *
0471      FROM pizza_type
0472      WHERE price IN (
0473          SELECT MAX(price) FROM pizza_type);
```

Data Output | Messages | Notifications

	pizza_typecode text	pizza_code text	size text	price numeric	pizza_typeid [PK] integer	pizza_id integer
	the_greek_xxl	the_greek	XXL	719	44	14

```
70477  SELECT *
70478      FROM pizza_type
70479      WHERE price IN (
70480          SELECT MIN(price) FROM pizza_type);
```

Data Output | Messages | Notifications

	pizza_typecode text	pizza_code text	size text	price numeric	pizza_typeid [PK] integer	pizza_id integer
1	pepperoni_s	pepperoni	S	195	37	13

Output: -

```
70448  /* TOP 5 months which generated the most revenue by sales of pizza */
70449
70450  SELECT
70451      EXTRACT (MONTH FROM date) as months,
70452      SUM (d.quantity * t.price) as total_revenue
70453      FROM pizzas as p
70454          INNER JOIN pizza_type as t
70455              ON p.pizza_id = t.pizza_id
70456          INNER JOIN order_details as d
70457              ON d.pizza_typeid = t.pizza_typeid
70458          INNER JOIN orders as o
70459              ON o.order_id = d.order_id
70460      GROUP BY months
70461      ORDER BY total_revenue DESC
70462      LIMIT 5;
70463
70464
70465
```

Data Output | Messages | Notifications

	months numeric	total_revenue numeric
1	7	1451158
2	5	1428055
3	3	1407942
4	11	1407907
5	1	1395866

```
70407  SELECT
70408      p.name,
70409      --t.pizza_code,
70410      SUM (t.price * d.quantity) as revenue
70411      FROM pizza_type as t
70412      INNER JOIN order_details as d
70413          ON t.pizza_typeid = d.pizza_typeid
70414      INNER JOIN pizzas as p
70415          ON p.pizza_id = t.pizza_id
70416      GROUP BY p.name,t.pizza_code
70417      ORDER BY revenue DESC
70418      LIMIT 3;
70419
```

Data Output | Messages | Notifications

	name text	revenue numeric
1	The Thai Chicken Pizza	868685
2	The Barbecue Chicken Pizza	855360
3	The California Chicken Pizza	828190

Output: -

```sql
/* How much Revenue was generated from sales of pizzas from each category ? */

SELECT
    p.category,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON t.pizza_typeid = d.pizza_typeid
    GROUP BY p.category
    ORDER BY total_revenue DESC;
```

Data Output | Messages | Notifications

	category text	total_revenue numeric
1	Classic	4401062
2	Supreme	4163940
3	Chicken	3918390
4	Veggie	3873809

```sql
/* For which date the pizza sales were maximum ? */

SELECT
    date,
    SUM (d.quantity * t.price) as total_revenue
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY date
    ORDER BY date DESC
    LIMIT 1;
```

Data Output | Messages | Notifications

	date date	total_revenue numeric
1	2015-12-31	58320

Output: -

```
70337  SELECT
70338      t.size,
70339      --d.pizza_typecode,
70340      SUM (t.price * d.quantity) as revenue
70341      FROM pizza_type as t
70342      INNER JOIN order_details as d
70343          ON t.pizza_typeid = d.pizza_typeid
70344      GROUP BY t.size
70345      ORDER BY revenue DESC;
```

Data Output | Messages | Notifications

	size text	revenue numeric
1	L	7506374
2	M	4987645
3	S	3561530
4	XL	281520
5	XXL	20132

```
70374  /* Total Revenue generated per day by sale of pizzas */
70375
70376  SELECT
70377      date,
70378      SUM (d.quantity * t.price) as total_revenue
70379      FROM pizzas as p
70380          INNER JOIN pizza_type as t
70381              ON p.pizza_id = t.pizza_id
70382          INNER JOIN order_details as d
70383              ON d.pizza_typeid = t.pizza_typeid
70384          INNER JOIN orders as o
70385              ON o.order_id = d.order_id
70386      GROUP BY date
70387      ORDER BY date;
70388
```

Data Output | Messages | Notifications

	date date	total_revenue numeric
1	2015-01-01	54277
2	2015-01-02	54638
3	2015-01-03	53248
4	2015-01-04	35109
5	2015-01-05	41319
6	2015-01-06	48579
7	2015-01-07	44044
8	2015-01-08	56767
9	2015-01-09	42547

Output: -

```sql
/* Group the orders by date and calculate the average number of pizzas ordered per day. */

SELECT
    ROUND(AVG(total_quantity), 0) as avg_pizza_ordered_per_day
    FROM
        (
            SELECT
                o.date,
                SUM(d.quantity) as total_quantity
            FROM
                orders as o
            JOIN order_details as d
                ON o.order_id = d.order_id
            GROUP BY o.date
            ORDER BY o.date
        ) as order_quantity;
```

Data Output | Messages | Notifications

	avg_pizza_ordered_per_day numeric
1	138

```sql
/* During which hour most no of pizza were ordered? */

SELECT
    EXTRACT ('hour' FROM time) as hours,
    COUNT (d.quantity ) as total_quantity
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON d.pizza_typeid = t.pizza_typeid
        INNER JOIN orders as o
            ON o.order_id = d.order_id
    GROUP BY hours
    ORDER BY total_quantity DESC
    LIMIT 1;
```

Data Output | Messages | Notifications

	hours numeric	total_quantity bigint
1	12	6543

Output: -

```sql
/* How much pizzas were ordered from each category? */

SELECT
    p.category,
    SUM (d.quantity) as total_quantity
    FROM pizzas as p
        INNER JOIN pizza_type as t
            ON p.pizza_id = t.pizza_id
        INNER JOIN order_details as d
            ON t.pizza_typeid = d.pizza_typeid
    GROUP BY p.category
    ORDER BY total_quantity DESC;
```

Data Output | Messages | Notifications

	category text	total_quantity bigint
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

```sql
/* Maximum quantity of pizzas ordered based on size */

SELECT
    t.size,
    SUM (d.quantity) as total_quantity
    FROM pizza_type as t
    INNER JOIN order_details as d
        ON t.pizza_typeid = d.pizza_typeid
    GROUP BY t.size
    ORDER BY total_quantity DESC
    LIMIT 1;
```

Data Output | Messages | Notifications

	size text	total_quantity bigint
1	L	18956

Output: -

```
70202  /* Total Quantity of all pizzas ordered based on size */
70203
70204  SELECT
70205      t.size,
70206      SUM (d.quantity) as total_quantity
70207      FROM pizza_type as t
70208      INNER JOIN order_details as d
70209          ON t.pizza_typeid = d.pizza_typeid
70210      GROUP BY t.size
70211      ORDER BY total_quantity DESC;
70212
```

Data Output | Messages | Notifications

	size text	total_quantity bigint
1	L	18956
2	M	15635
3	S	14403
4	XL	552
5	XXL	28

```
70193  /* Total orders per hour */
70194
70195  SELECT
70196      EXTRACT ('hour' FROM time) as hours,
70197      COUNT(order_id) as total_orders
70198      FROM orders
70199      GROUP BY hours
70200      ORDER BY hours;
70201
```

Data Output | Messages | Notifications

	hours numeric	total_orders bigint
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455
6	14	1472
7	15	1468
8	16	1920
9	17	2336
10	18	2399
11	19	2009
12	20	1642
13	21	1198
14	22	663
15	23	28

Business Insights: -

- Percentage share of total sales by all four Pizza Categories are almost same with CLASSIC category being the highest (27%) followed by SUPREME category (25%).
- Among pizzas, THAI CHICKEN PIZZA contributes to maximum sales while CLASSIC DELUXE PIZZA was the most ordered one among the customers and BRIE CARRE PIZZA holds the position of least favoured pizza.
- Most Expensive was The Greek Pizza (Rs 719) while Least expensive was The Pepperoni Pizza (Rs 195)
- JULY month generated the highest revenue whereas MAXIMUM pizza sale was on 31 December.
- Average pizza ordered per day was 138.
- The busiest hours for pizza orders are between 12:00 PM to 1:00 PM and 5:00 PM to 6:00 PM, indicating high demand during lunch time and evening time.
- LARGE-sized pizzas are the most popular choice among customers followed by MEDIUM and then SMALL sizes.

Recommendations: -

- VEG category pizzas should be promoted with discounts and variety option to increase their sales similar to THAI CHICKEN PIZZA .
- Some Ingredients of CLASSIC DELUXE PIZZA can be mixed with BRIE CARRE PIZZA sales to make it more tasty.
- Discounts should be reduced during June-July season to tap more revenue, whereas offer heavy discounts for other months.
- Since people prefer pizzas to celebrate days like 31 Dec so for other occasions promotional offers should be increased.
- Sales Target should be set to increase the Average pizza ordered per day from 138 to 160.
- More manpower should be deployed to handle the busiest hours for pizza orders between 12:00 PM to 1:00 PM and 5:00 PM to 6:00 PM so that maximum sales can be generated.
- More variety options should be increased for SMALL and MEDIUM pizzas to attract more customers.
- Rates of pizzas can also be optimized and gap between maximum and minimum pizzas can be reduced to tap more customers.

Thank You