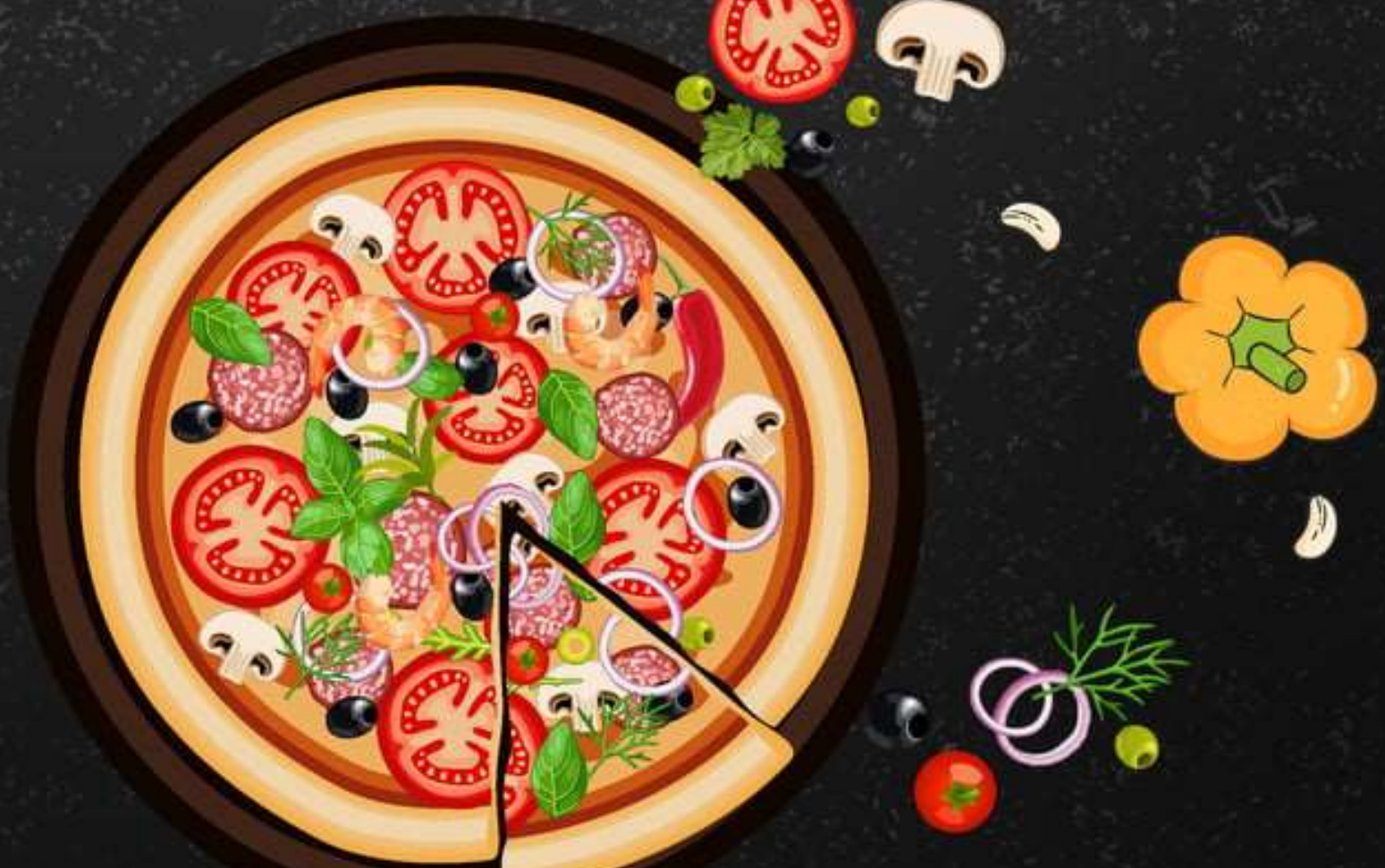
PIZZA SALES ANALYSIS

SQL



PIZZA ANALYSIS BY USING STRUCTURED QUERY LANGUAGE(SQL)

BY: PASAM NUTAN


```
#Retrieve the total number of orders placed.

select count(order_id) as total_orders from  orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_orders
21350

Result 1

Output

Action Output

#	Time	Action	Message
1	18:51:12	select count(order_id) as total_orders from orders LIMIT 0, 1000	1 row(s) returned


```sql
-- This query retrieves the names of pizza types and their corresponding prices.
-- JOIN operation link the tables and ensure the price matches the pizza type
SELECT pizza_types.name, pizzas.price
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	price
The Spinach and Feta Pizza	16
The Spinach and Feta Pizza	20.25
The Vegetables + Vegetabl...	12
The Vegetables + Vegetabl...	16
The Vegetables + Vegetabl...	20.25

Result Grid | Filter Rows: | Exp

name	price
The Southwest Chicken Pizza	16.75
The Southwest Chicken Pizza	20.75
The Thai Chicken Pizza	12.75
The Thai Chicken Pizza	16.75
The Thai Chicken Pizza	20.75

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

name	price
The Barbecue Chicken Pizza	12.75
The Barbecue Chicken Pizza	16.75
The Barbecue Chicken Pizza	20.75
The California Chicken Pizza	12.75
The California Chicken Pizza	16.75


```
#Calculate the total revenue generated from pizza sales

select round(sum(order_details.quantity * pizzas.price),2)  as total_sales from order_details join pizzas  on pizzas.
pizza_id = order_details.pizza_id ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	total_sales
▶	628080.4


```
#Identify the highest-priced pizza.
select pizza_types.name,pizzas.price from pizza_types join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id
order by pizzas.price desc limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

name	price
The Greek Pizza	35.95


```sql
#Identify the most common pizza size ordered.
SELECT pizzas.size, COUNT(order_details.order_details_id) AS order_count
FROM pizzas
JOIN order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

size	order_count
L	14247
M	11806
S	10826
XL	433
XXL	22


```sql
#List the top 5 most ordered pizza types along with their quantities.
SELECT pizzas.size, SUM(order_details.quantity) AS total_quantity
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.size
ORDER BY total_quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

size	total_quantity
L	14573
M	11987
S	11037
XL	437
XXL	22


```
#Determine the distribution of orders by hour of the day.
select hour(order_time) as hour ,count(order_id) as order_count from orders group by hour(order_time)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

hour	order_count
16	1920
17	2336
18	2399
19	2009
20	1642

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

hour	order_count
21	1198
22	663
23	28
10	8
9	1


```sql
-- This query calculates the rounded average quantity of items ordered per day.

SELECT ROUND(AVG(total_quantity), 0) AS average_quantity
FROM (
  SELECT SUM(order_details.quantity) AS total_quantity
  FROM orders
  JOIN order_details ON orders.order_id = order_details.order_id
  GROUP BY orders.order_date
) AS order_quantity;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

average_quantity
137


```sql
-- This query retrieves the top 3 pizza types by revenue, calculated as the sum of quantity ordered multiplied by price.
SELECT pizza_types.name,
       SUM(order_details.quantity * pizzas.price) AS revenue
FROM pizza_types
JOIN pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

name	revenue
The Barbecue Chicken Pizza	33354.75
The Thai Chicken Pizza	32725.75
The California Chicken Pizza	31608.75


```sql
-- This query calculates the revenue for each pizza category, summing the total revenue based on quantity ordered and pizza price.
SELECT pizza_types.category,
       SUM(order_details.quantity * pizzas.price) AS revenue
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category	revenue
Classic	169636.65000000002
Supreme	158982.99999999886
Chicken	149814
Veggie	149646.74999999956


```sql
-- This query calculates the cumulative revenue over time, with the cumulative total increasing with each order date.
SELECT
    order_date,
    SUM(revenue) OVER (ORDER BY order_date) AS cum_revenue
FROM (
    SELECT
        orders.order_date,
        SUM(order_details.quantity * pizzas.price) AS revenue
    FROM
        order_details
    JOIN
        pizzas ON order_details.pizza_id = pizzas.pizza_id
    JOIN
        SELECT
            orders.order_date,
            SUM(order_details.quantity * pizzas.price) AS revenue
        FROM
            order_details
        JOIN
            pizzas ON order_details.pizza_id = pizzas.pizza_id
        JOIN
            orders ON orders.order_id = order_details.order_id
        GROUP BY
            orders.order_date
) AS sales;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_date	cum_revenue
2015-01-01	3147.1000000000004
2015-01-02	6276.75
2015-01-03	9065.15
2015-01-04	10820.6
2015-01-05	12886.55

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_date	cum_revenue
2015-01-11	26819.65
2015-01-12	28738.7
2015-01-13	30788.300000000003
2015-01-14	33315.700000000004
2015-01-15	35300.50000000001


```sql
-- This query retrieves the top 3 pizzas by revenue within each category.
SELECT
    name,
    revenue
FROM (
    SELECT
        category,
        name,
        revenue,
        RANK() OVER (PARTITION BY category ORDER BY revenue DESC) AS rn
    FROM (
        SELECT
            pizza_types.category,
            pizza_types.name,
            SUM(order_details.quantity * pizzas.price) AS revenue
        FROM
            pizza_types
        JOIN
            pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
            order_details ON order_details.pizza_id = pizzas.pizza_id
        GROUP BY
            pizza_types.category,
            pizza_types.name
    ) AS a
) AS b
WHERE rn <= 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	revenue
The Barbecue Chicken Pizza	33354.75
The Thai Chicken Pizza	32725.75
The California Chicken Pizza	31608.75
The Classic Deluxe Pizza	29159.5
The Hawaiian Pizza	24403.25

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	revenue
The Italian Supreme Pizza	25858.5
The Sicilian Pizza	23188
The Four Cheese Pizza	24378.50000000041
The Five Cheese Pizza	20757
The Mexicana Pizza	20272.75