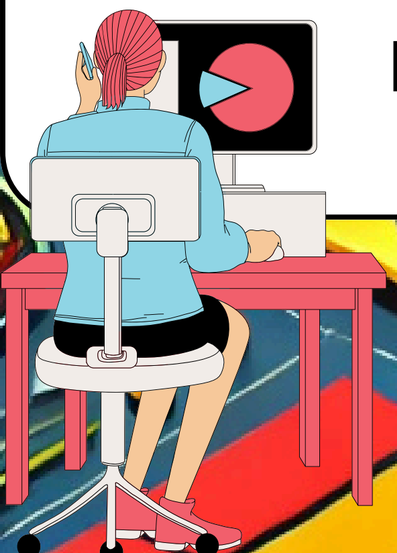
PIZZA SALES

Welcome to the Pizza Sales Analysis Project! This project aims to provide insights into pizza sales data by utilizing SQL queries to analyze various aspects of the sales performance.



By - Krishnanand Shah

Q1.

```sql
-- Solving basic question
-- 1>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

Q2.

```sql
-- 2>Calculate the total revenue generated from pizza sales.
SELECT
    SUM(order_details.quantity * pizzas.price) 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
817860.049999993

Q3.

```sql
-- 3>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

Q4.

```
44
45      -- 4>Identify the most common pizza size ordered.
46  •   SELECT
47          pizzas.size,
48          COUNT(order_details.order_details_id) AS order_count
49      FROM
50          pizzas
51              JOIN
52          order_details ON pizzas.pizza_id = order_details.pizza_id
53      GROUP BY pizzas.size
54      ORDER BY order_count DESC;
55
```

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

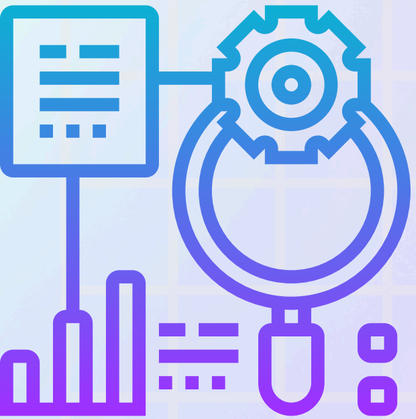
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

Q5.

```sql
-- Intermediate:
-- 5> Determine the top 3 most ordered pizza types based on revenue.
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 Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5



Q6.

```sql
/*5>List the top 5 most ordered pizza types along with their quantities*/
SELECT
    pizza_types.name, SUM(order_details.quantity) AS 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 pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```

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

name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Q7.

```sql
-- Intermediate:
-- 4>Group the orders by date and calculate the average number of pizzas ordered per day.
/*We can also try to this round off in 2 decimal places*/
SELECT
    ROUND(AVG(quantity), 0) as avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS 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:

avg_pizza_ordered_per_day
138

Q8.

```sql
-- Intermediate:
/*3>Join relevant tables to find the category-wise distribution of pizzas.*/
SELECT
    category, COUNT(category)
FROM
    pizza_types
GROUP BY category;
```

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

category	COUNT(category)
Chicken	6
Classic	8
Supreme	9
Veggie	9

Q9.

```sql
-- Intermediate:
-- 2>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
16	1920

Result 3

Q10.

```sql
-- Intermediate:
-- 1>Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS 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 pizza_types.category
ORDER BY quantity DESC;
```

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

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Q11.

```sql
-- Advanced:
-- 3>Determine the top 3 most ordered pizza types based on revenue for each pizza 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 Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25

Q12.

```sql
-- Advanced:
-- 1>Calculate the percentage contribution of each pizza type to total revenue.
select pizza_types.category,
round(sum(order_details.quantity*pizzas.price)/ (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)*100,2) 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	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

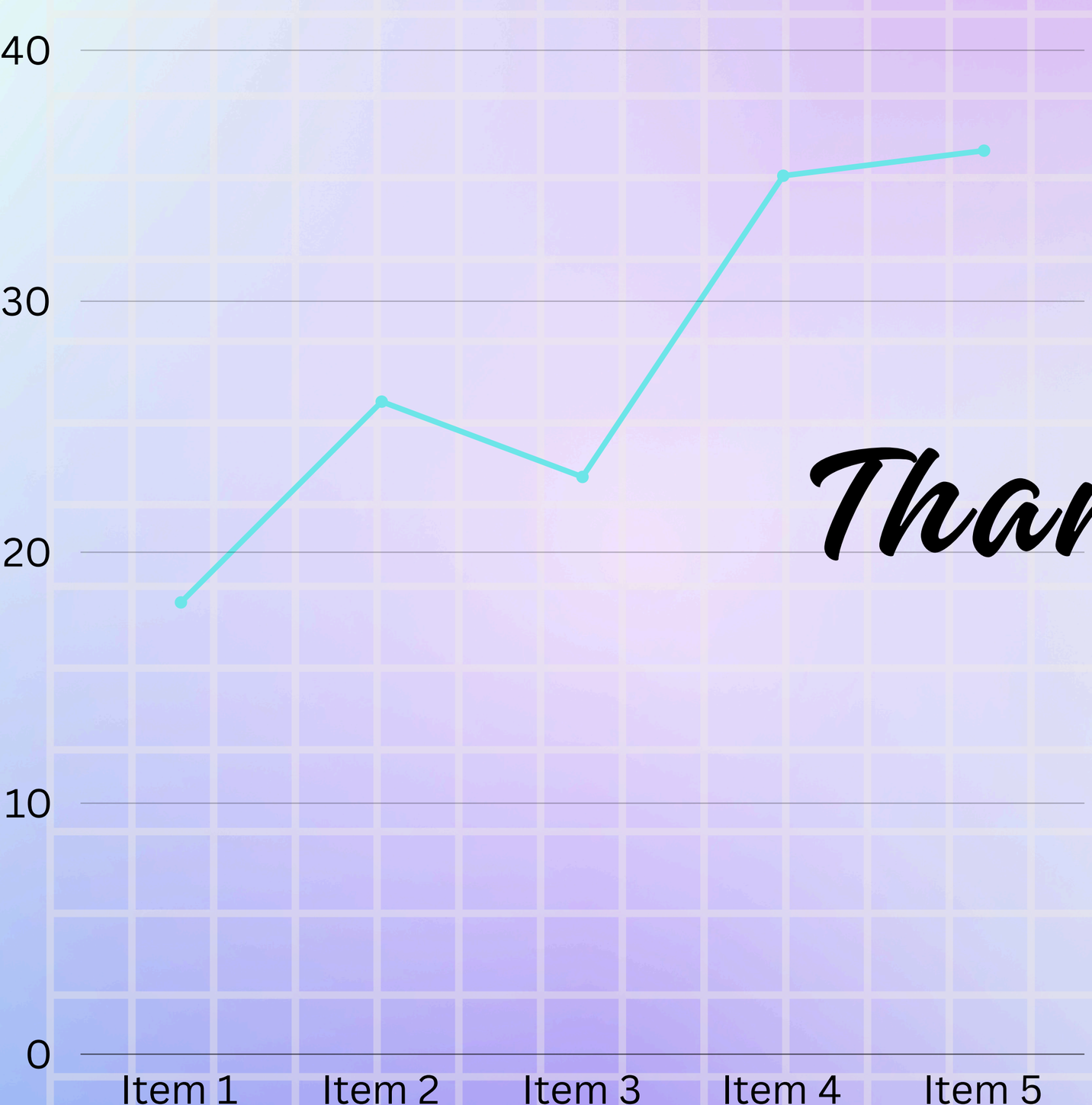
Q13.

```sql
-- Advanced:
-- 2>Analyze the cumulative revenue generated over time.
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 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	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4

Result 10



Thank You