PIZZA SALES ANALYSIS

Using SQL

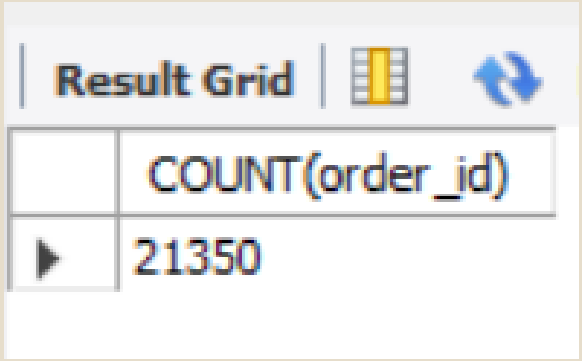
Presented By Tabish Ahmad

In this SQL project, I analysed some of the queries of Pizza Sales in Pizza Hut.

The queries with its outputs are given in the further pages.

1. Retrieve the total number of orders placed

```sql
SELECT
    COUNT(order_id)
FROM
    orders;
```



2. Calculate the total revenue generated from pizza sales

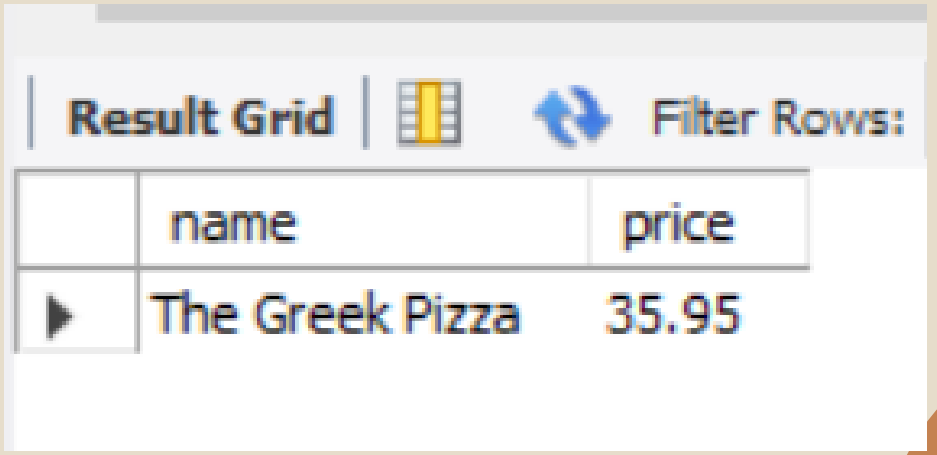
```
SELECT
    ROUND(SUM(`order_details`.quantity * pizzas.price),
            2) AS total_revenue
FROM
    `order_details`
        JOIN
    pizzas ON pizzas.pizza_id = `order_details`.pizza_id;
```

Result Grid

	total_revenue
▶	817860.05

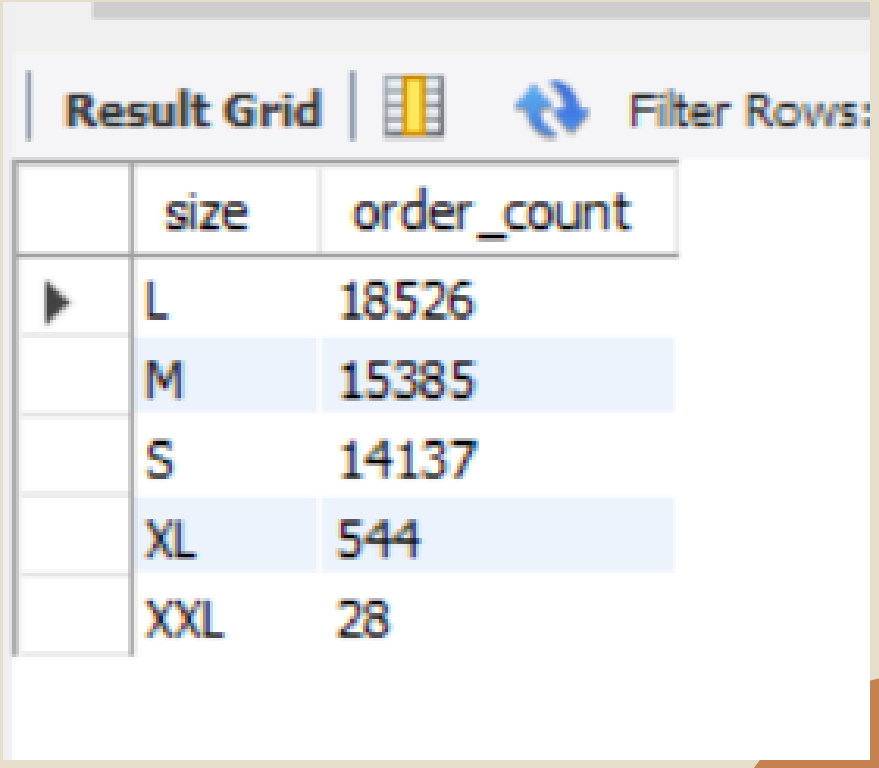
3. Identify the highest-priced pizza.

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



4. 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;
```



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

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:

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

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

```sql
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

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

7. 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 | Filte

Hour	Order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663

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

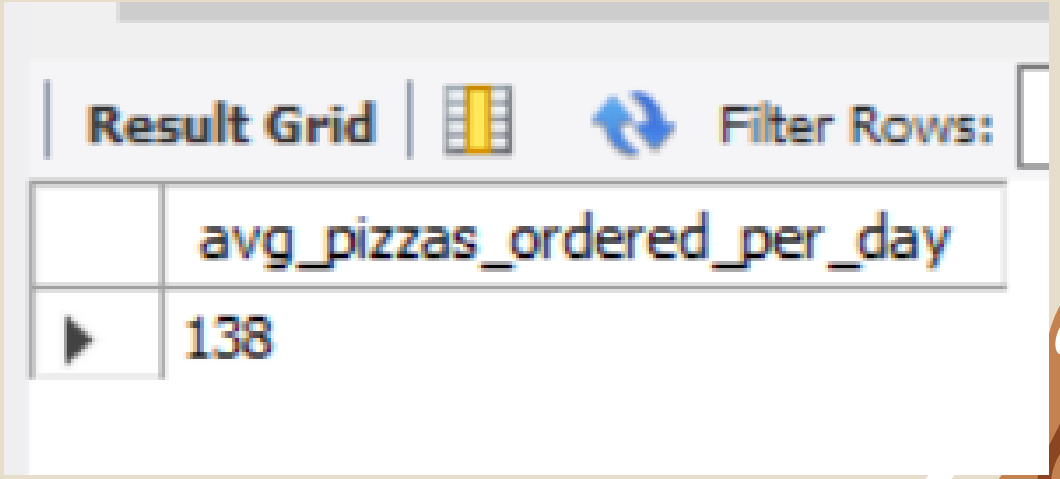
```
select category, count(name)
from pizza_types
group by category;
```

Result Grid | Filter Rows:

category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

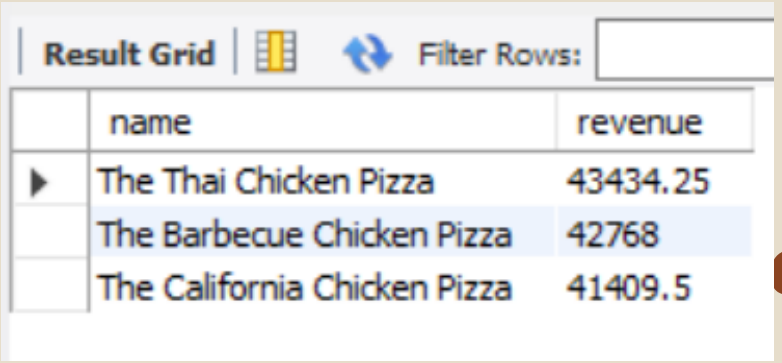
9. Group the orders by date and calculate the average number of pizzas ordered per day.

```
select round(avg(quantity),0) as avg_pizzas_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
```



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

```sql
SELECT
    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.name
ORDER BY revenue DESC
LIMIT 3;
```



Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

11. 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_revenue
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;
```

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

12. Analyze the cumulative revenue generated over time.

```sql
select order_date,
sum(revenue) over (order by order_date) as cumulative_revenue
from
(select orders.order_date, sum(order_details.quantity * pizzas.price) as revenue
from orders join order_details
on orders.order_id = order_details.order_id
join pizzas
on pizzas.pizza_id = order_details.pizza_id
group by orders.order_date) as daily_revenue;
```

Result Grid | Filter Rows:

order_date	cumulative_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

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

```sql
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
order by category) as a) as b
where rn <=3;
```

Result Grid | Filter Rows:

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

THANK YOU

Tabish Ahmad
M.Tech (Computational Mathematics)