Where Every Slice is a Taste of Perfection

A DEEP DIVE INTO PIZZA SALES

Start Your Slide

ABOUT THE PROJECT

This project analyzes pizza sales data using SQL to extract key insights, such as top-selling pizzas, revenue trends, and customer preferences. Through structured queries, the data is filtered, aggregated, and presented in a meaningful way to support data-driven decision-making for optimizing sales and inventory management.

Retrieve the total number of orders placed

```
5 •   select count(order_id) as total_orders from orders
6
```

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

total_orders
21350

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

MOST POPULAR PIZZAS AMONG CUSTOME

The classic deluxe

The hawaiian

The barbecue

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 pizzas.price desc limit 1;
```

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered

```sql
select pizzas.size,count(order_details.quantity) 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

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

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

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;
```

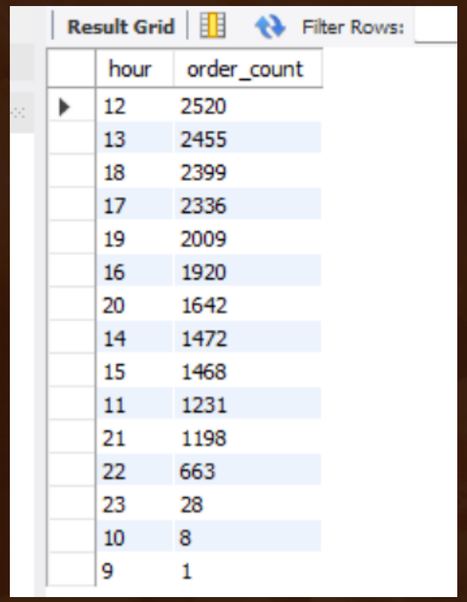
Result Grid | Filter Rows:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day

```sql
select * from orders;

select hour(order_time) as hour,count(order_id) as order_count
from orders
group by hour order by order_count desc;
```

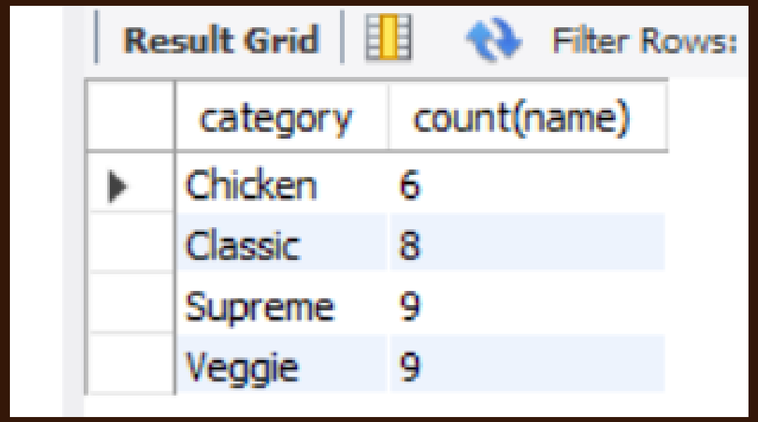


Result Grid | Filter Rows:

hour	order_count
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231
21	1198
22	663
23	28
10	8
9	1

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

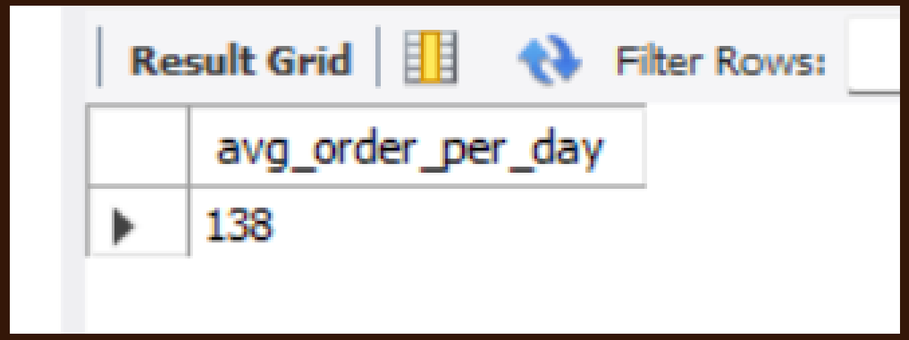
```
select category,count(name) from pizza_types
group by category;
```



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

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

```sql
select round(avg(quantity),0) as avg_order_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 order by quantity) as order_quantity;
```



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 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 5;
```

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 Spicy Italian Pizza	34831.25

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 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:

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
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65