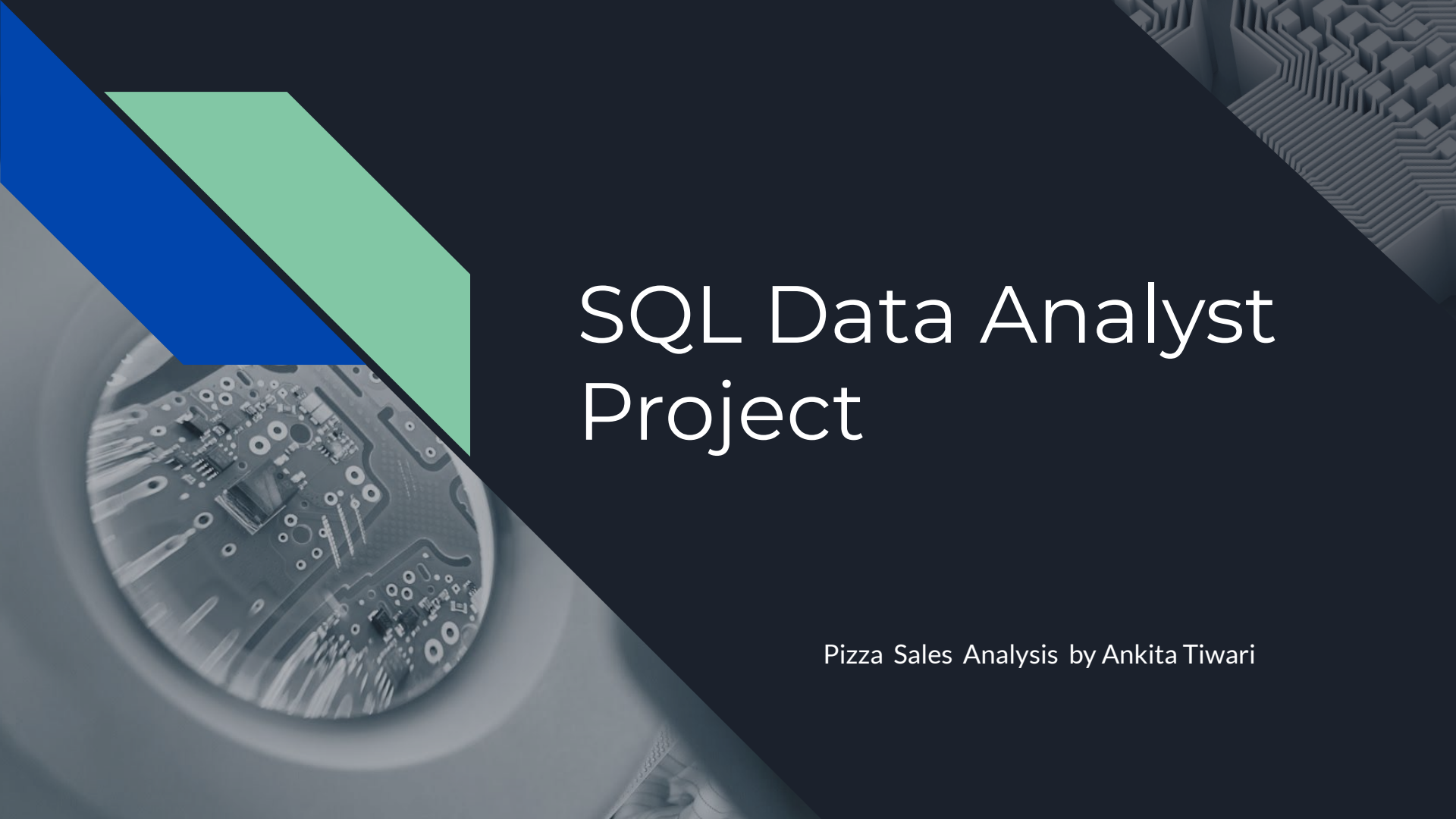


SQL Data Analyst Project

Pizza Sales Analysis by Ankita Tiwari

Overview of Pizza Sales Analysis

Objective:

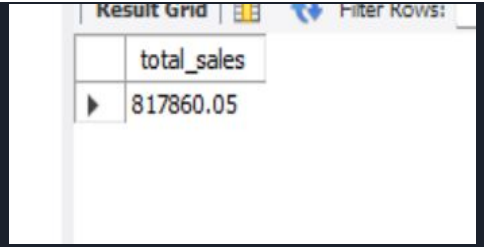
Pizza sales analysis involves examining the performance of a pizza business by analyzing sales data to uncover trends, measure efficiency, and improve decision-making. It helps identify key drivers of revenue, customer preferences, and operational strengths or weaknesses, enabling businesses to optimize their strategies.

1. Retrieve the total no of order placed.

```
3 • select count(order_id)  as total_orders
4   from orders;
```

2.Calculate total revenue generated from pizza sale

```sql
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
```



3 . identify the highest priced pizza

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

Result Grid | Filter Rows:

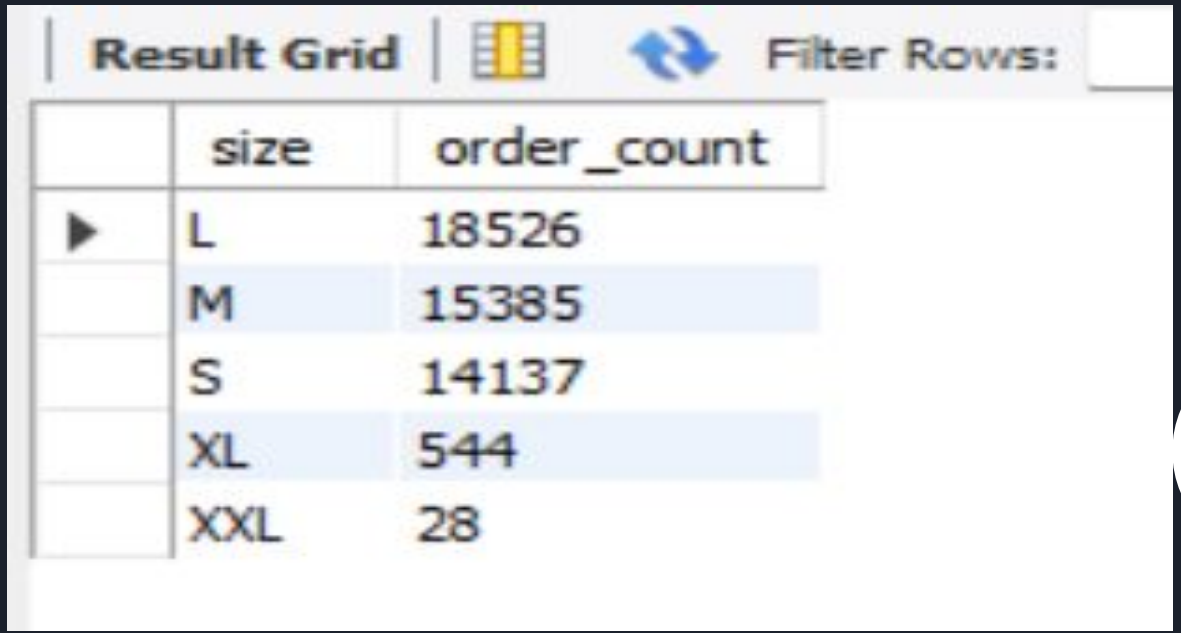
name	price
The Greek Pizza	35.95

4. identify the most common pizza size ordered.

```sql
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;
```

4.

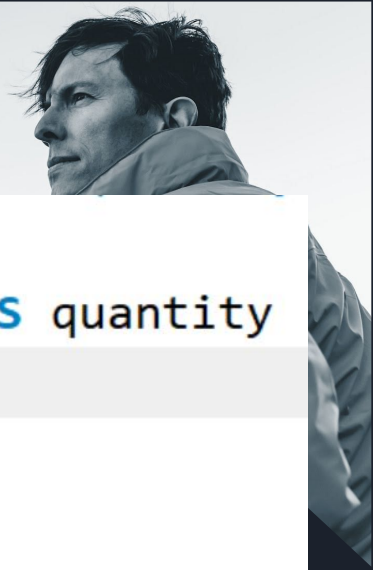
Output common order pizza size



Result Grid | Filter Rows:

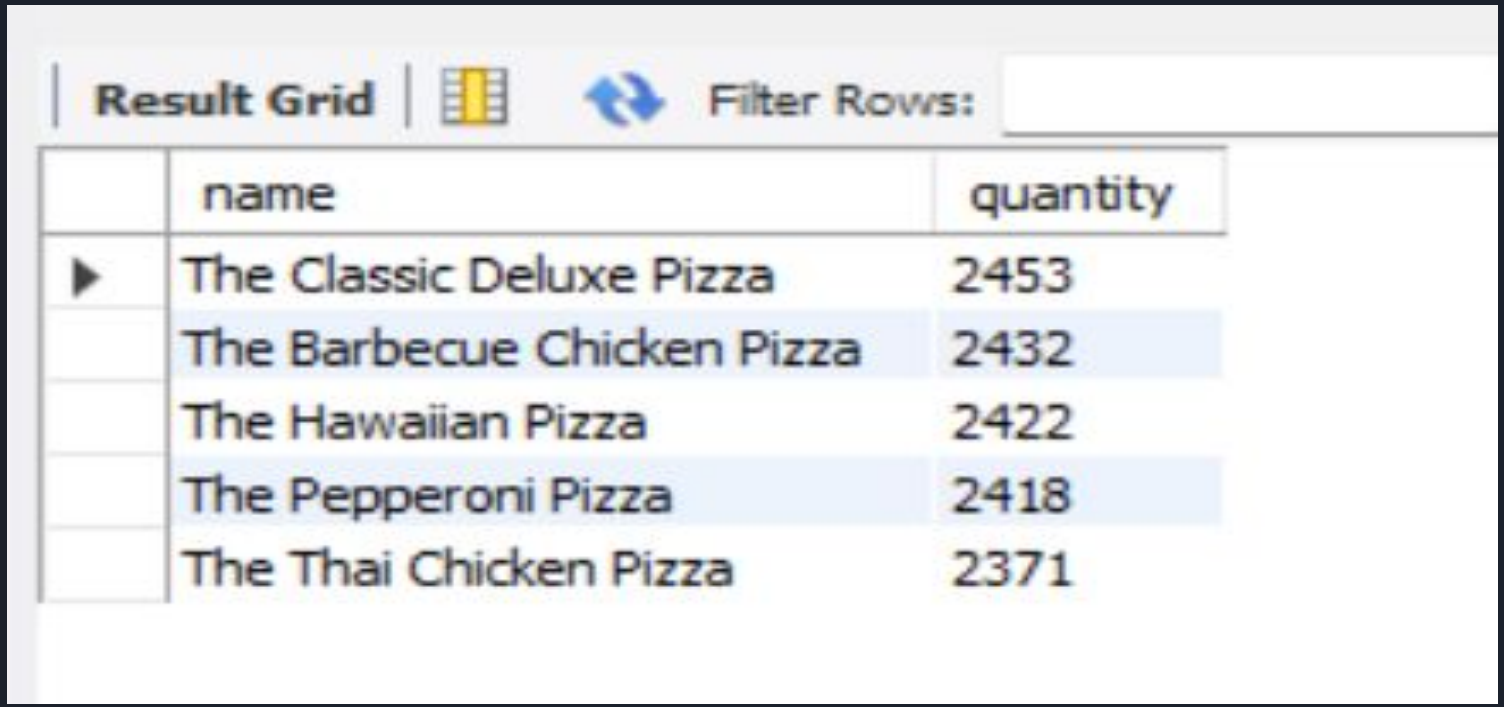
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

5. list the top 5 most ordered pizza types alon



```sql
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;
```

Output 5 most ordered pizza name along with quantity.

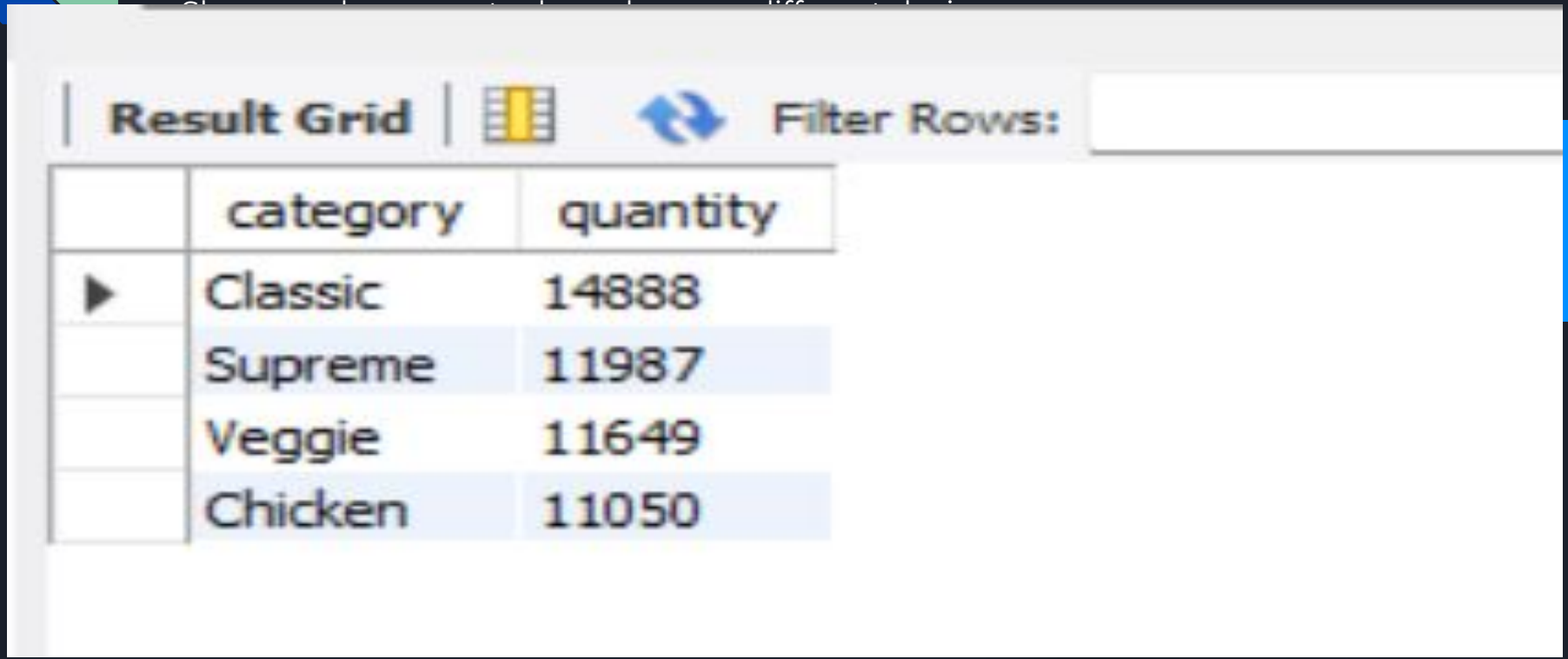


name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

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;
```

Output



	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

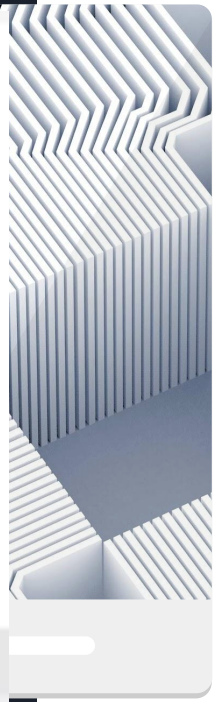
7. determine distribution of orders by hour of the day

```sql
SELECT
    HOUR(time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(time);
```

OUPUT

	hour	order_count
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009

Result 1 ×



find category-wise distribution of pizza.

```
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

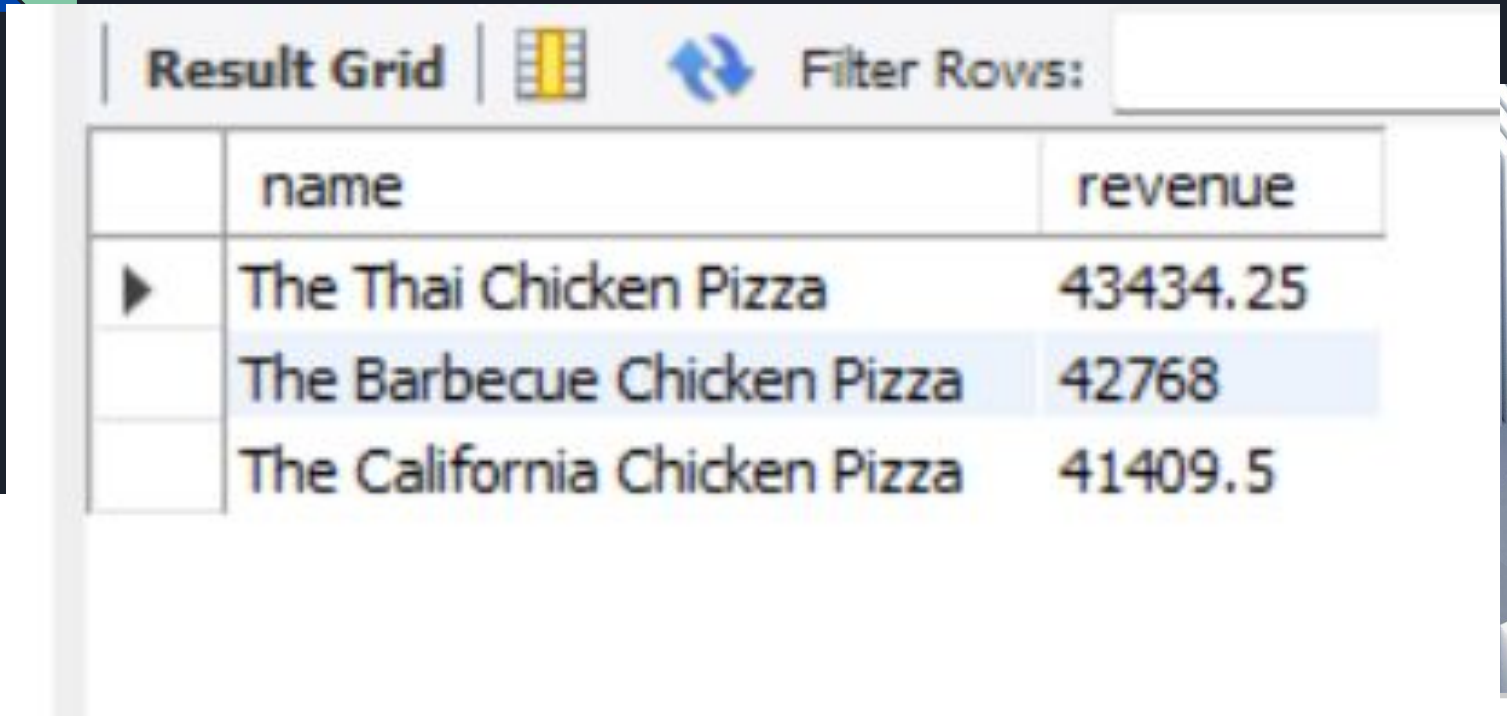
group the orders by date and calculate the average number of pizzas ordered per day

```sql
SELECT
    round(AVG(quantity), 0)
FROM
    (SELECT
        orders.date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_
    GROUP BY orders.date) AS order_quantity;
```

determine the top 3 most ordered pizza types based

```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 3;
```

OUTPUT



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

```sql
-- 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;
```

OUTPUT revenue in %

Result Grid | Filter Rows:

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

analyze the cumulative revenue generated over time

```sql
select date ,
sum(revenue) over(order by date) as cum_revenue
from
(select orders.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.date) as sales;
```

output

	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
	2015-01-10	23990.350000000002
	2015-01-11	25862.65
	2015-01-12	27781.7
	2015-01-13	29831.300000000003
	2015-01-14	32358.700000000004
	2015-01-15	34343.50000000001
	2015-01-16	36937.65000000001
	2015-01-17	39001.75000000001
	2015-01-18	40978.60000000001

Result 1