Q7. Determine the distribution of orders by hour of the day.

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

Result Grid

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336

Q8. Join relevant tables to find the category-wise distribution of pizzas.

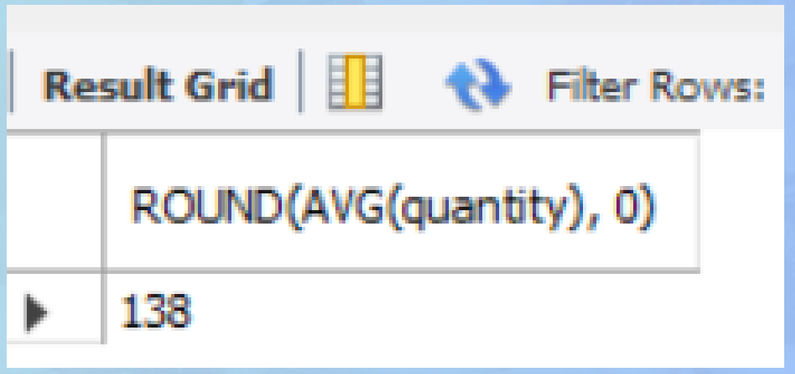
```sql
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Rows:

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

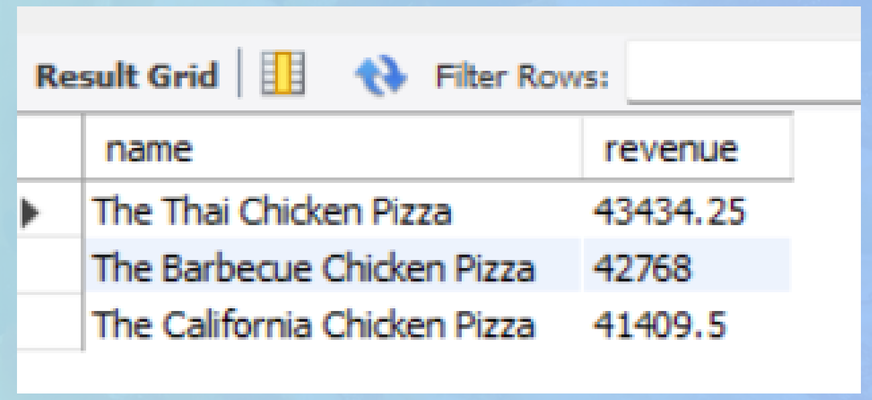
Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

```sql
SELECT
    ROUND(AVG(quantity), 0)
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;
```



Q10. Determine the top 3 most ordered pizza types based on revenue.

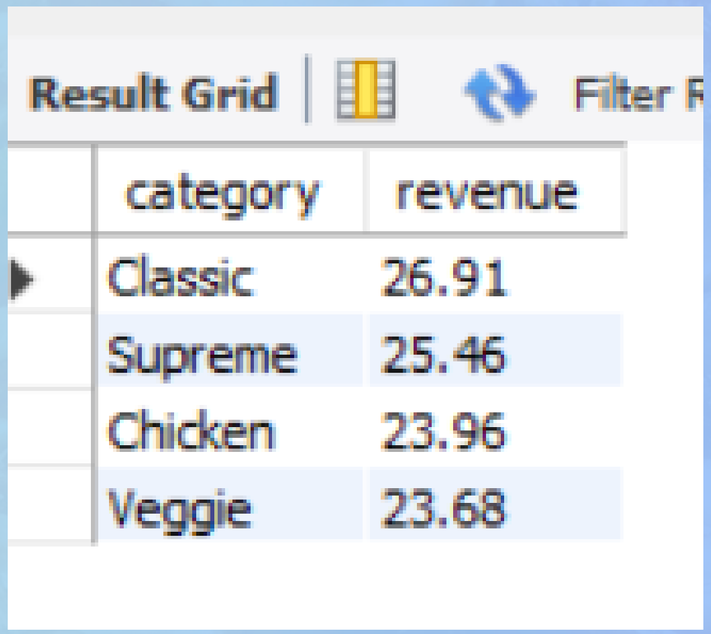
```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```



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

Q11. 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 R

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Q12. 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:

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

Q13. 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) 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
The Italian Supreme Pizza	33476.75

THANK YOU !