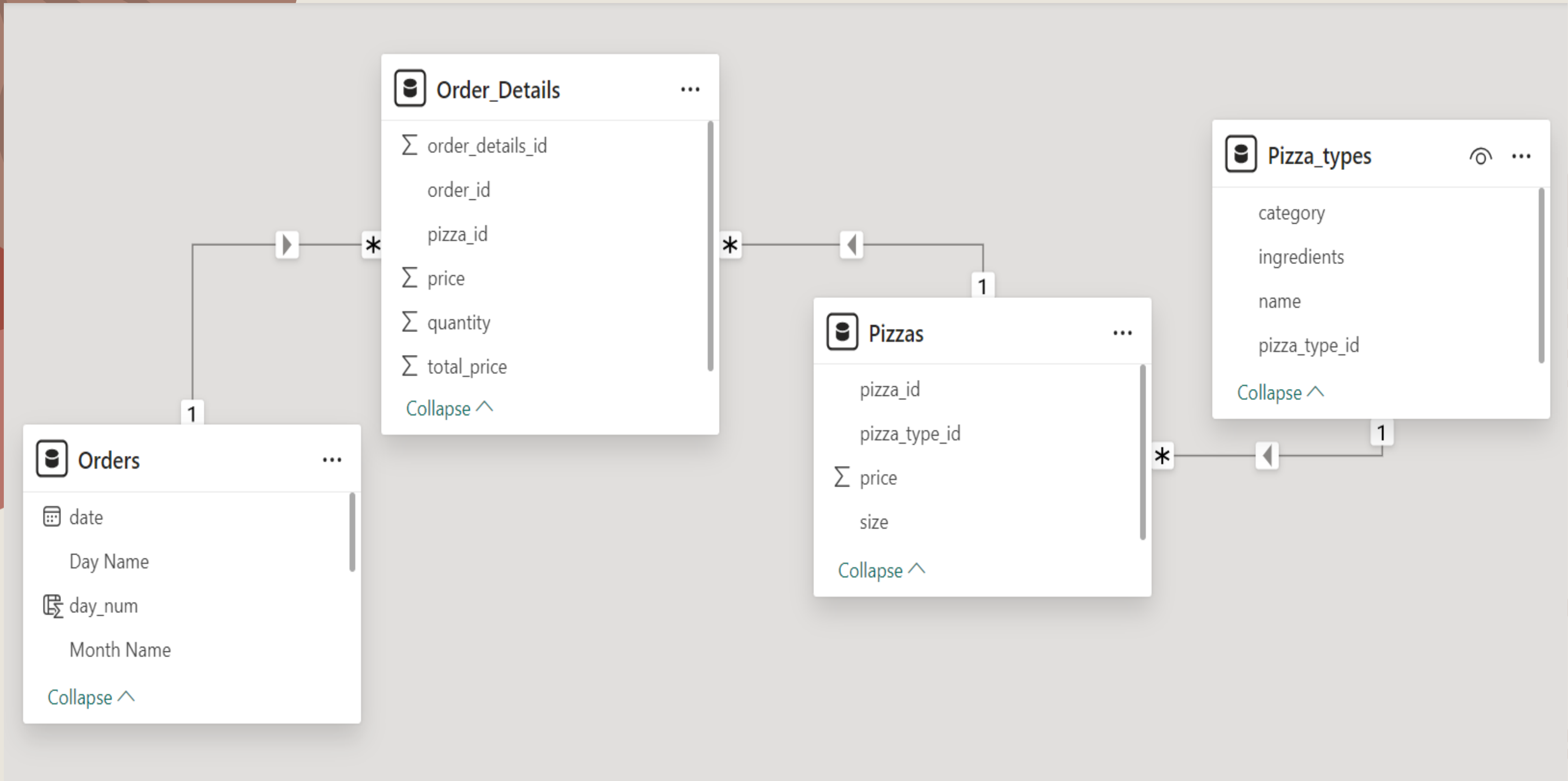
AN SQL PROJECT
ON PIZZA SHOP SALES

Hello! I am Israt Jahan Upama, currently an undergraduate student of Materials and Metallurgical Engineering at BUET, pursuing a passion on data analysis.

In this project, I have utilized sql queries to solve questions that are related to pizza sales.

Data In Model View



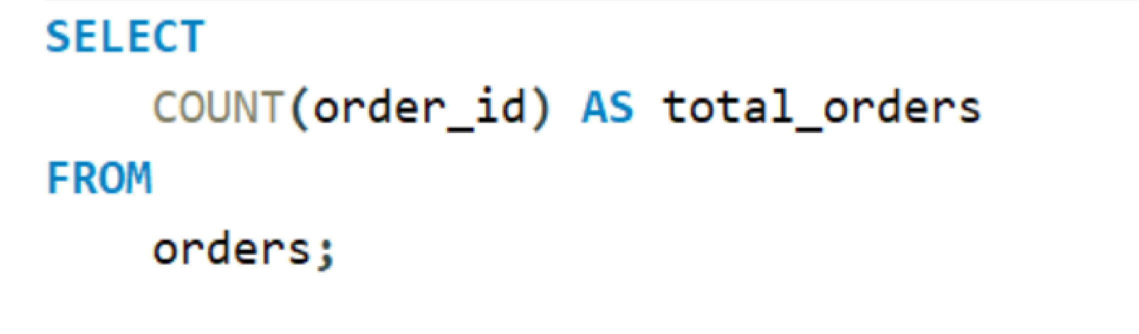
Order_Details
- ∑ order_details_id
- order_id
- pizza_id
- ∑ price
- ∑ quantity
- ∑ total_price

Orders
- date
- Day Name
- day_num
- Month Name

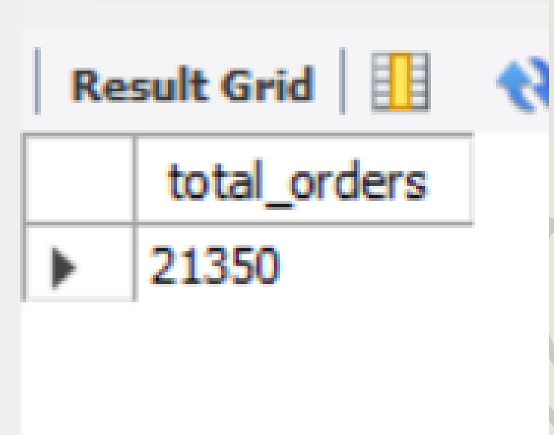
Pizzas
- pizza_id
- pizza_type_id
- ∑ price
- size

Pizza_types
- category
- ingredients
- name
- pizza_type_id

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

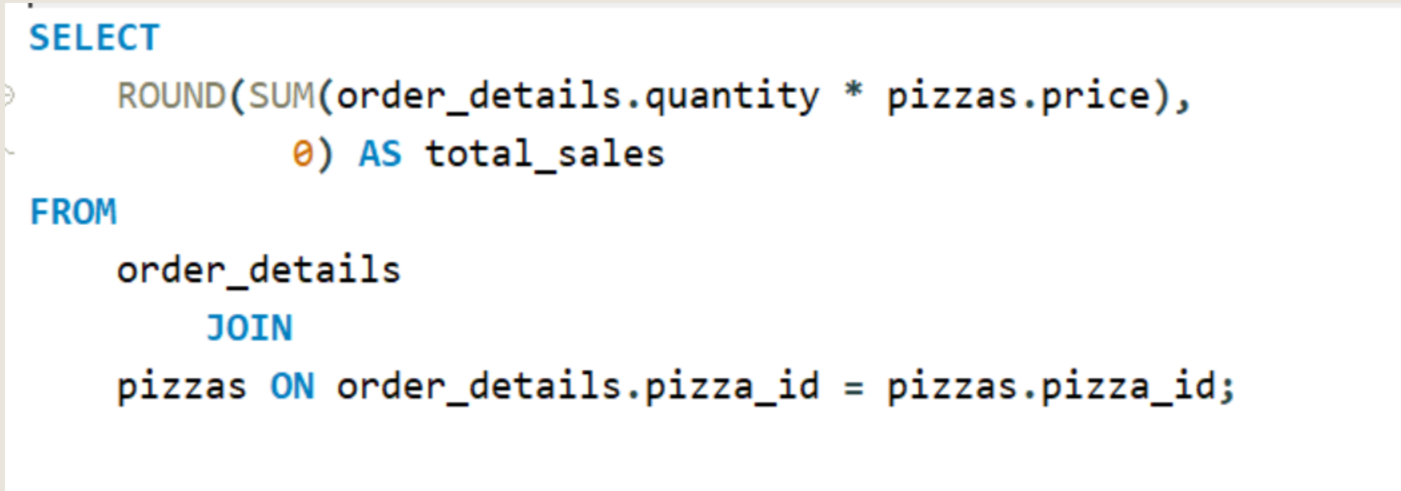


```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

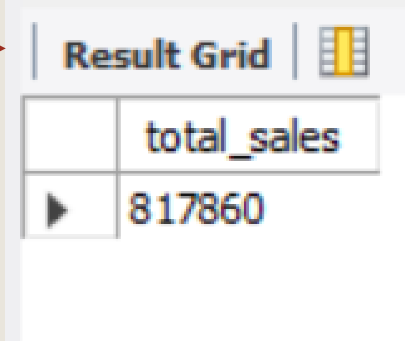


```
total_orders
21350
```

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

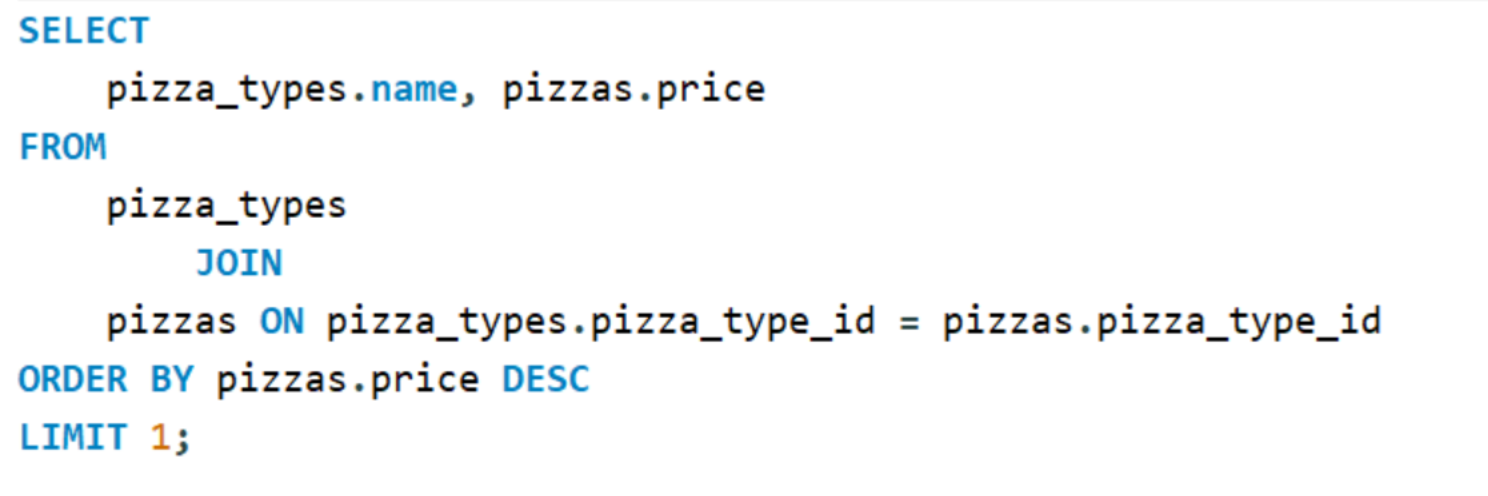


```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            0) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id;
```

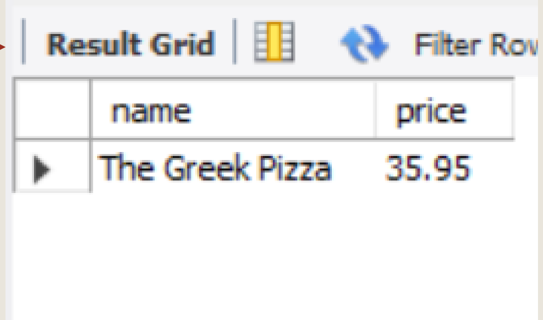


total_sales
817860

IDENTIFY THE HIGHEST-PRICED PIZZA.

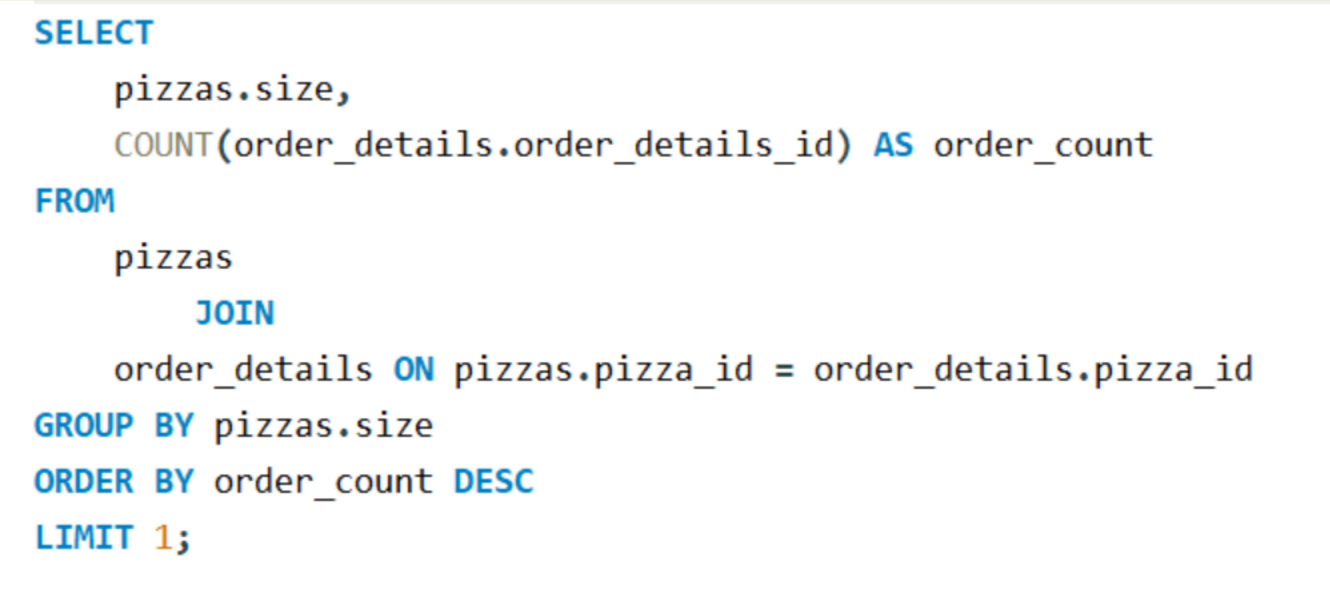


```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

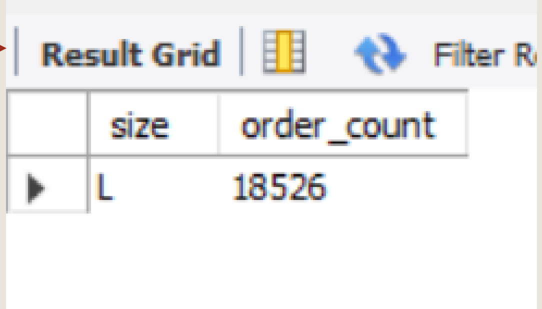


name	price
The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

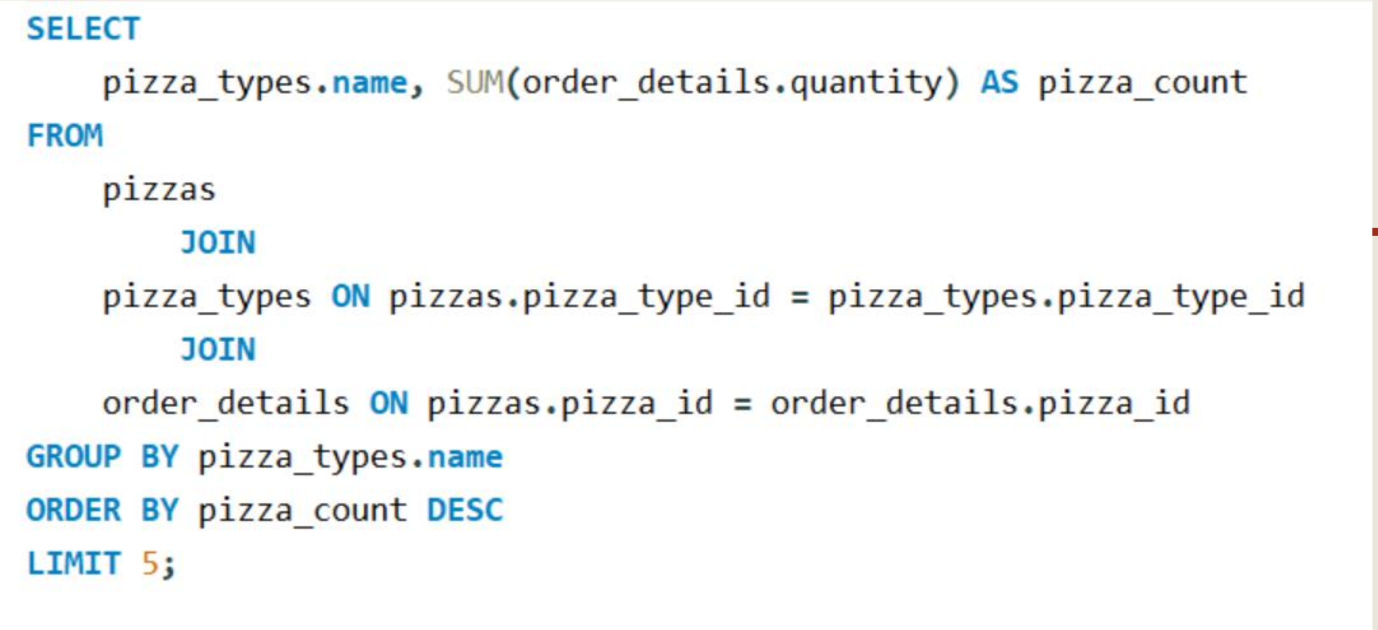


```
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```

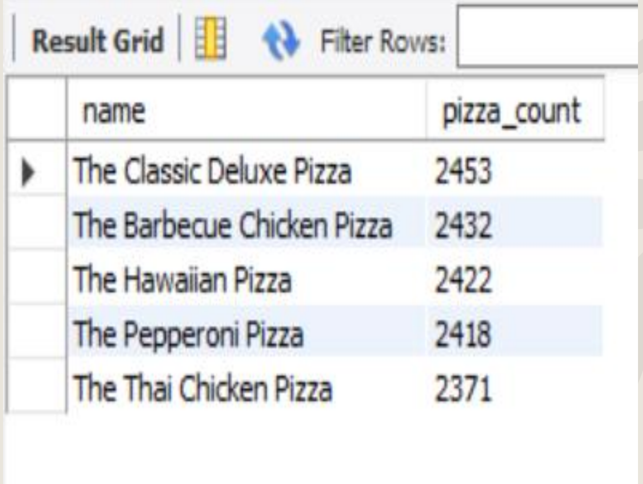


```
  size   order_count
▶ L      18526
```

LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

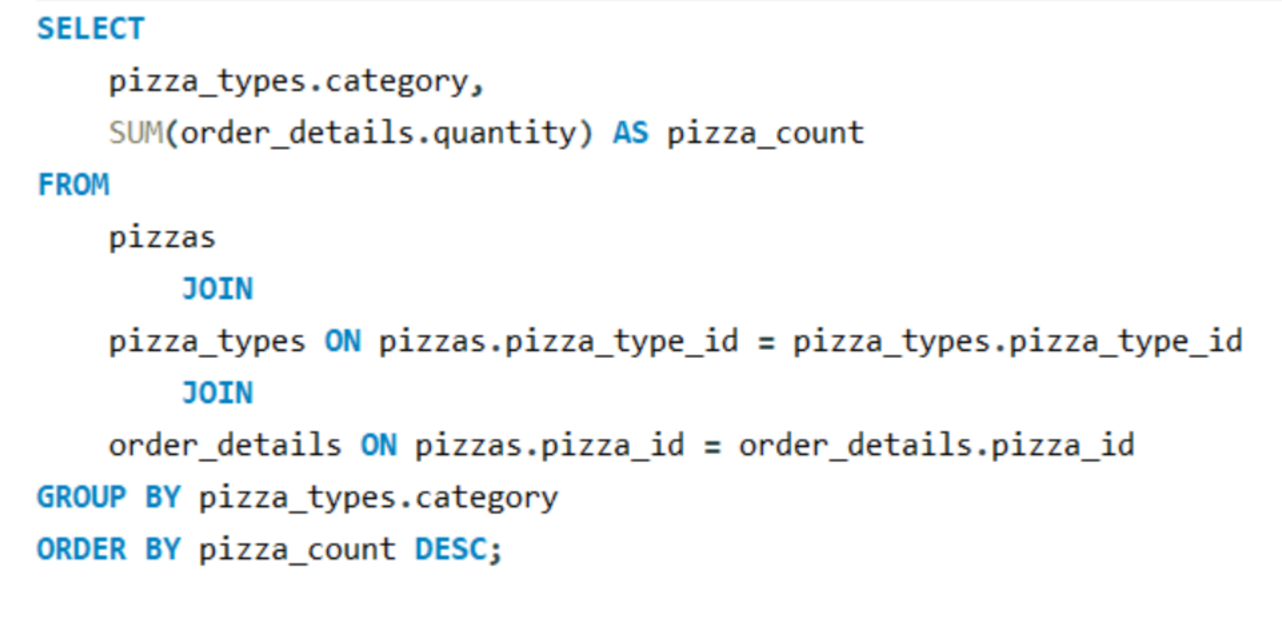


```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS pizza_count
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY pizza_count DESC
LIMIT 5;
```

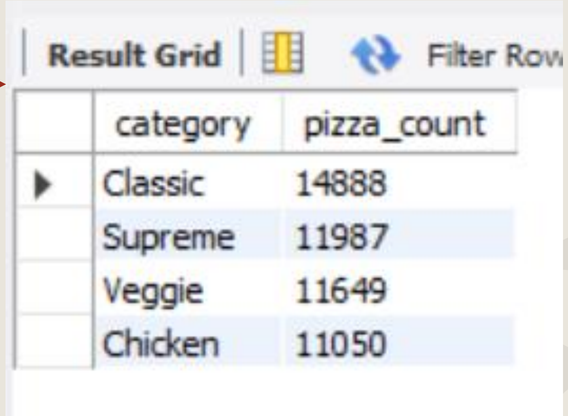


name	pizza_count
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

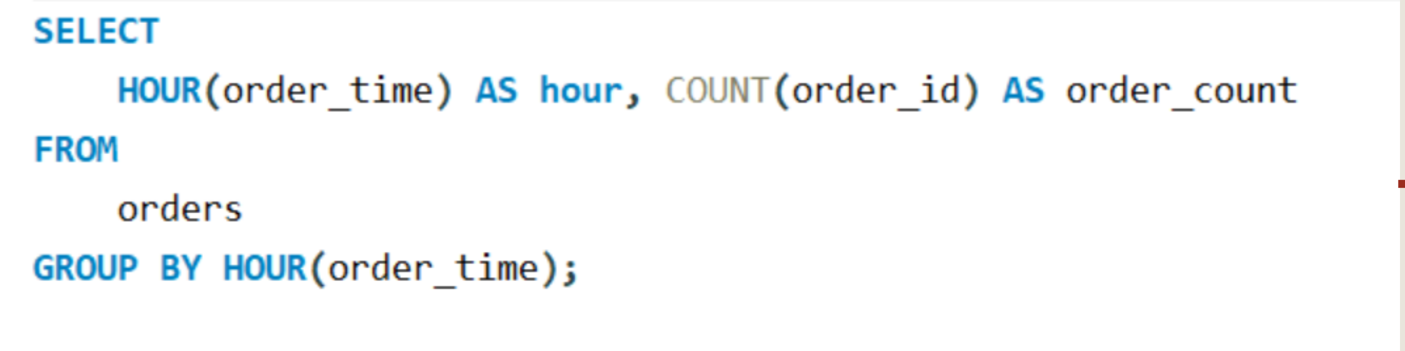


```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS pizza_count
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY pizza_count DESC;
```

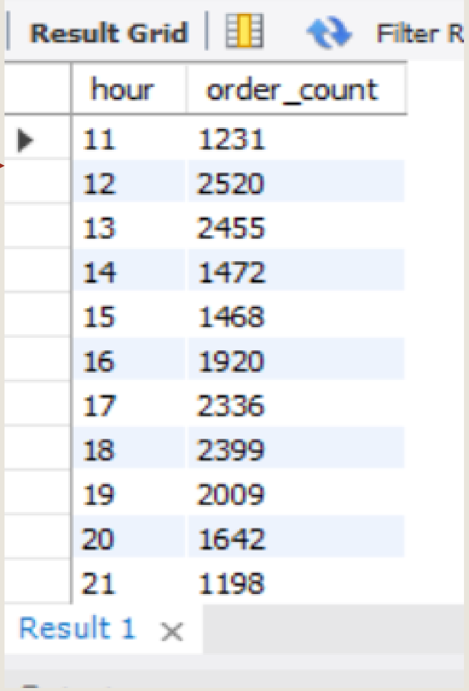


category	pizza_count
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

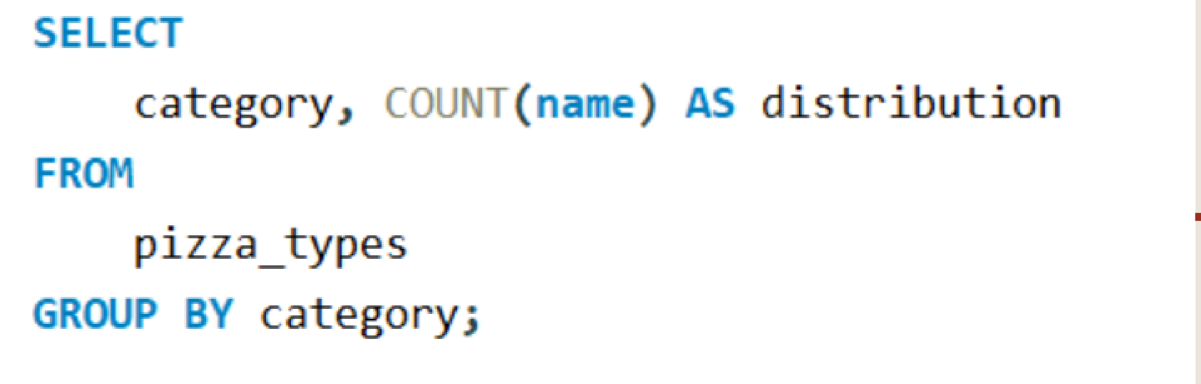


```sql
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```

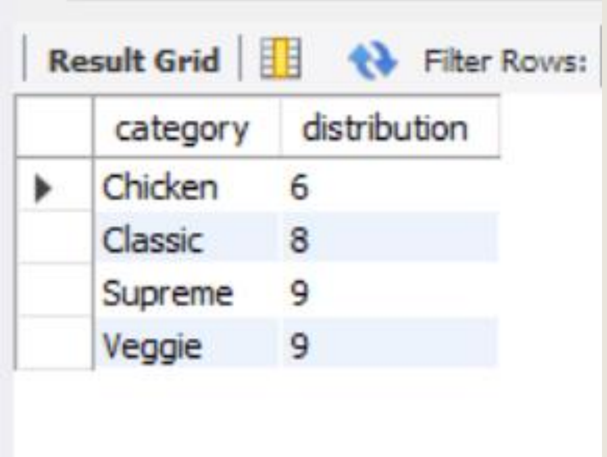


hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198

JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

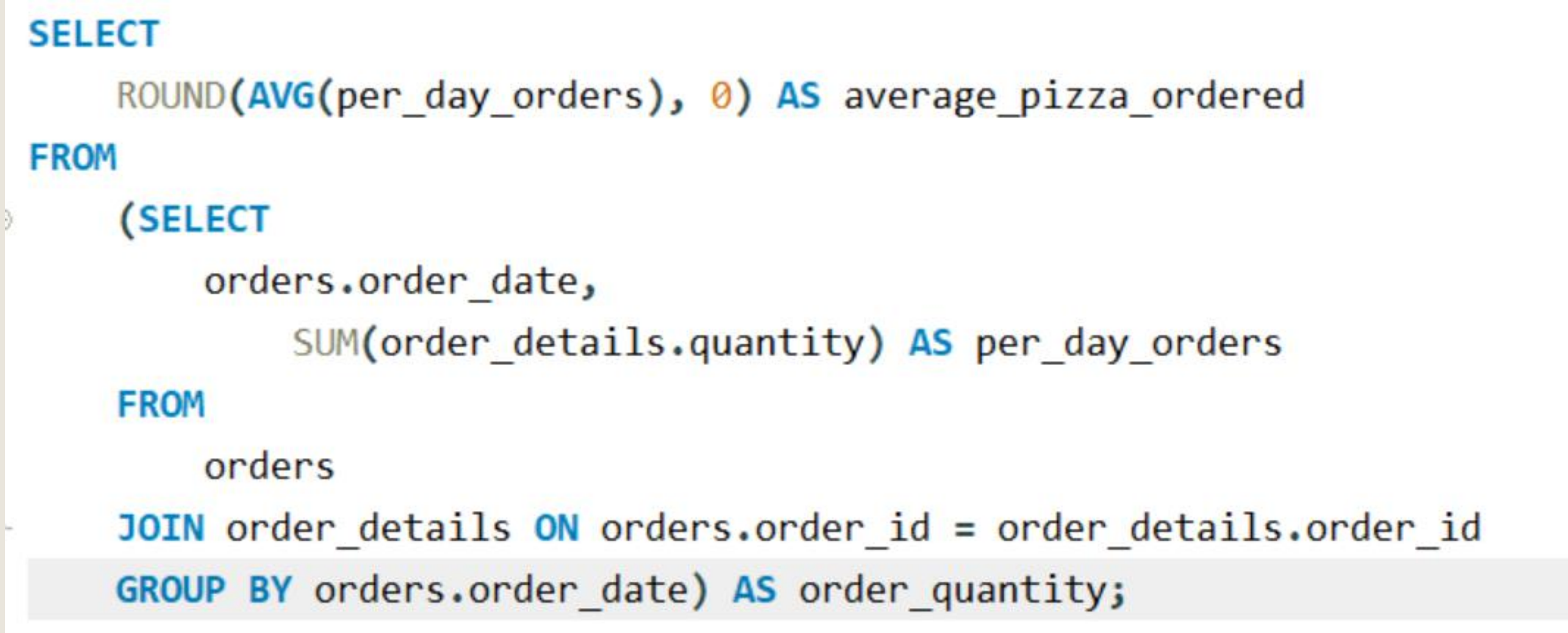


```
SELECT
    category, COUNT(name) AS distribution
FROM
    pizza_types
GROUP BY category;
```

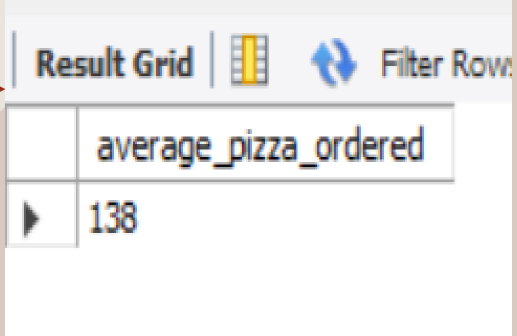


category	distribution
Chicken	6
Classic	8
Supreme	9
Veggie	9

GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

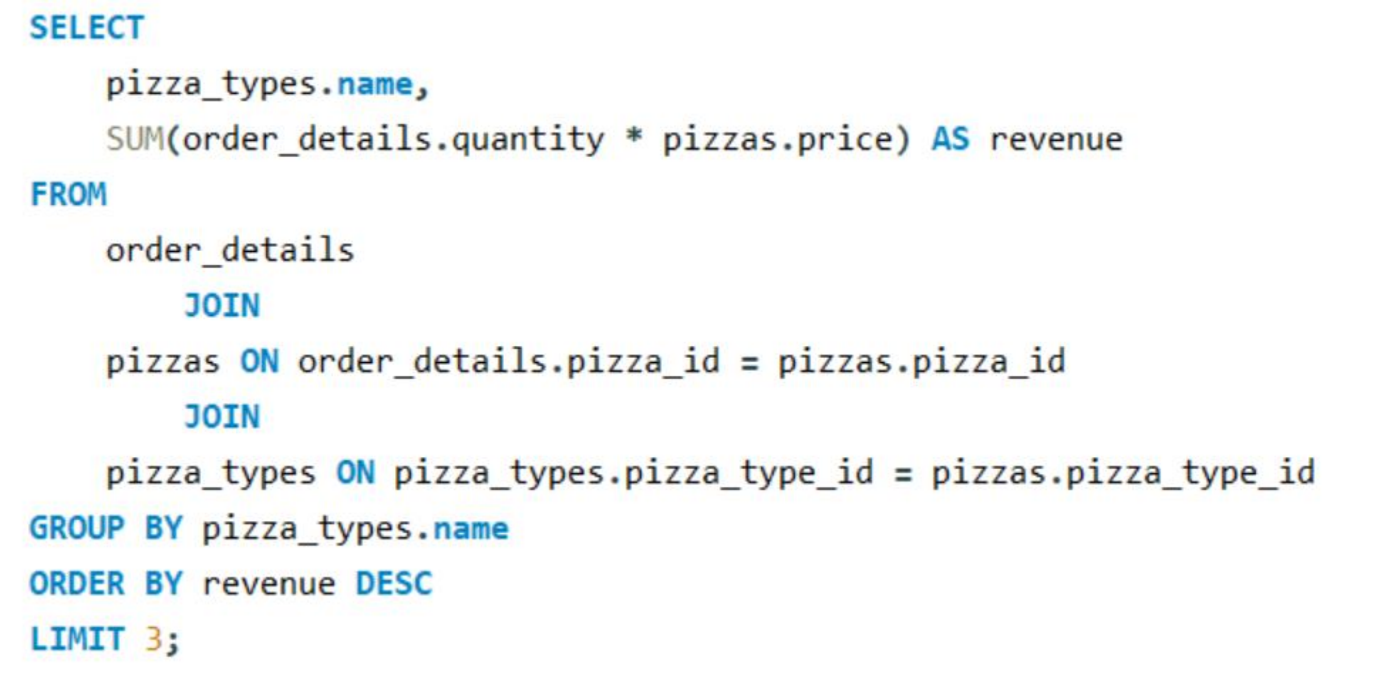


```
SELECT
    ROUND(AVG(per_day_orders), 0) AS average_pizza_ordered
FROM
    (SELECT
        orders.order_date,
            SUM(order_details.quantity) AS per_day_orders
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

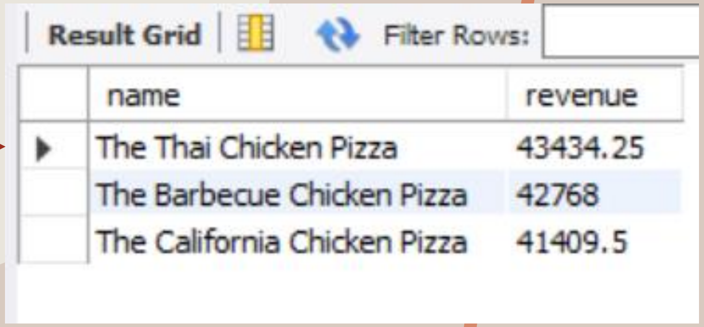


```
average_pizza_ordered
138
```

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

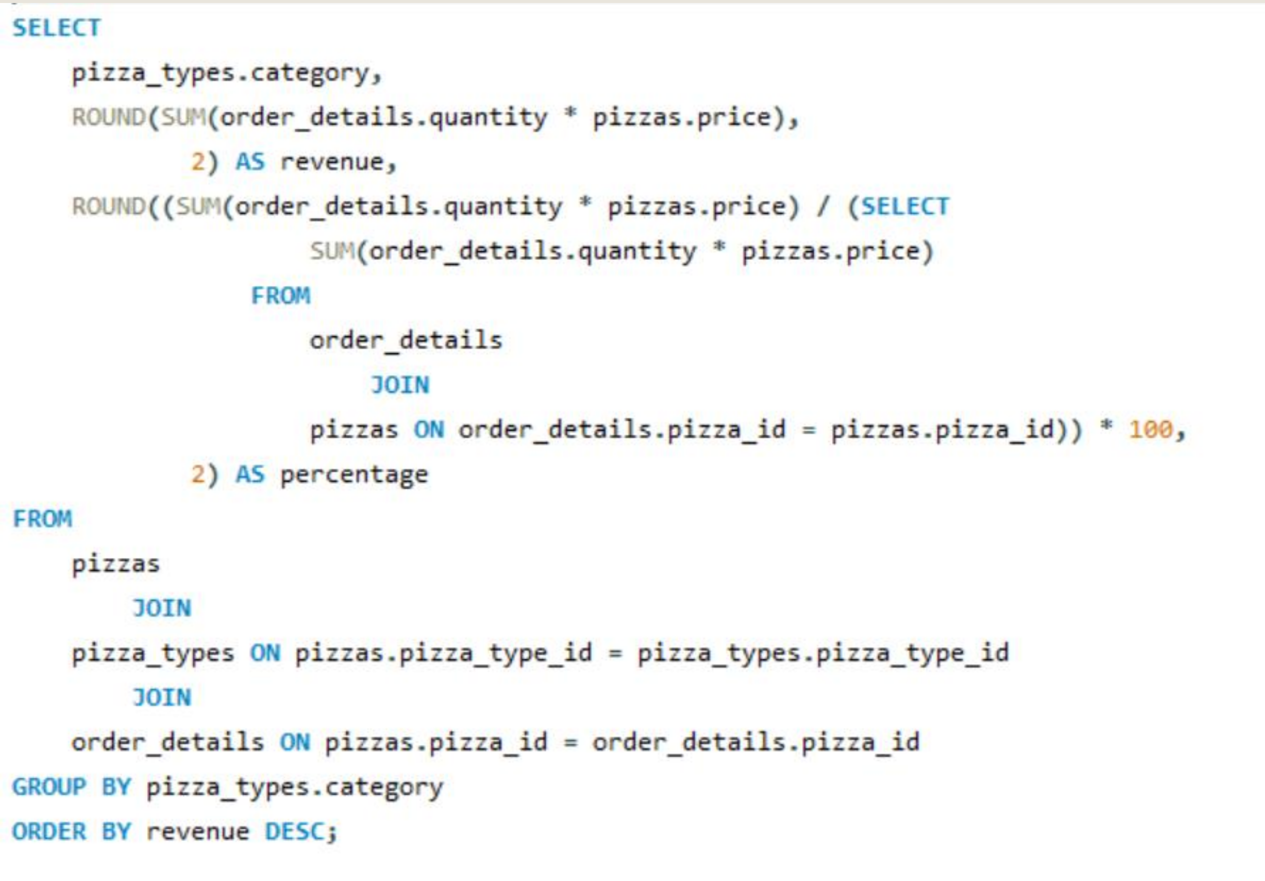


```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
        JOIN
    pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id
GROUP BY pizza_types.name
ORDER BY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

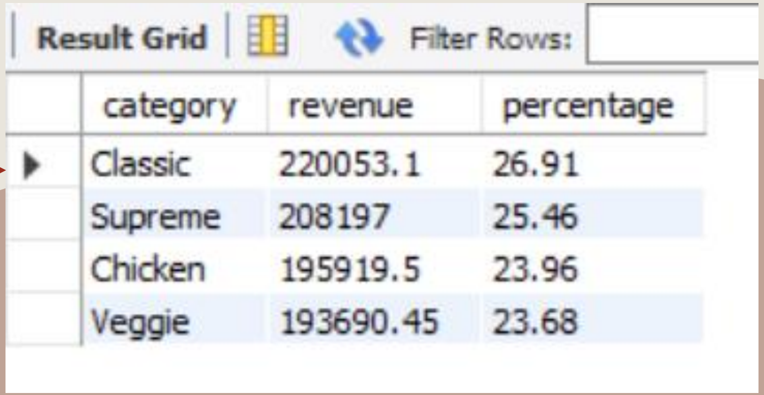


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

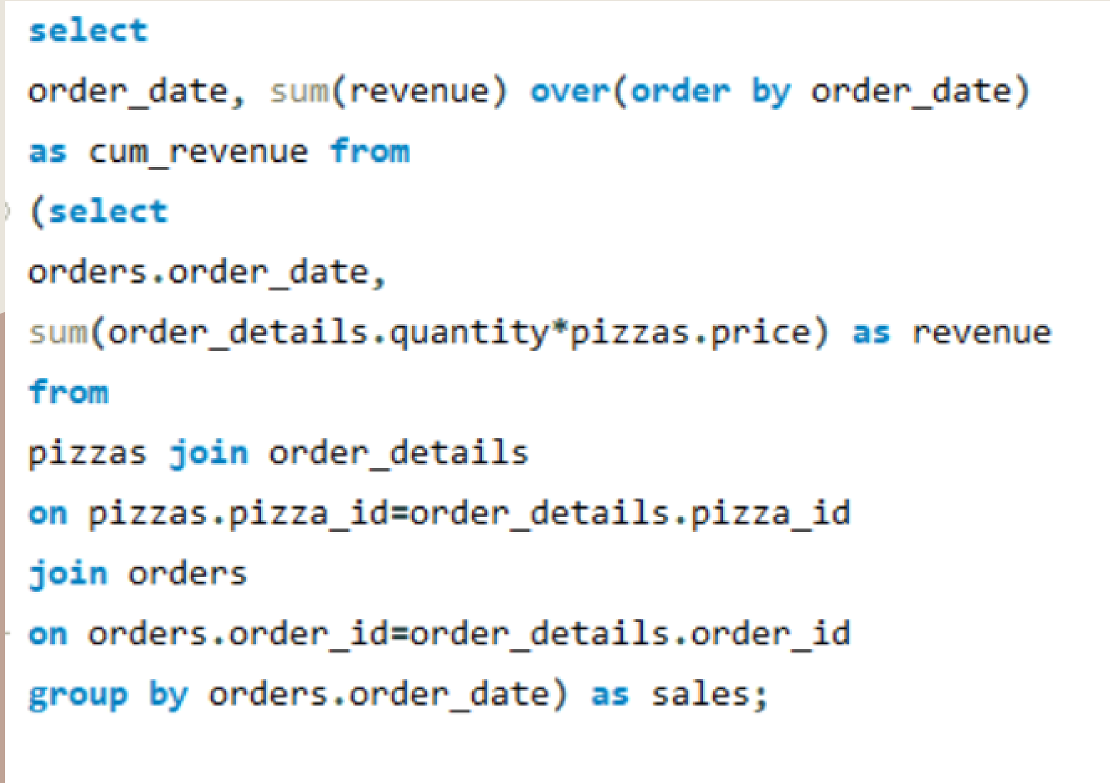


```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS revenue,
    ROUND((SUM(order_details.quantity * pizzas.price) / (SELECT
                SUM(order_details.quantity * pizzas.price)
            FROM
                order_details
                    JOIN
                pizzas ON order_details.pizza_id = pizzas.pizza_id)) * 100,
            2) AS percentage
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

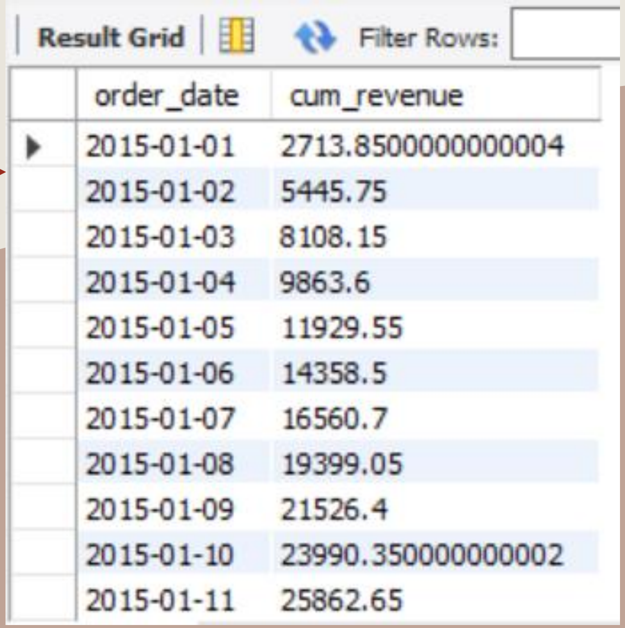


category	revenue	percentage
Classic	220053.1	26.91
Supreme	208197	25.46
Chicken	195919.5	23.96
Veggie	193690.45	23.68

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.



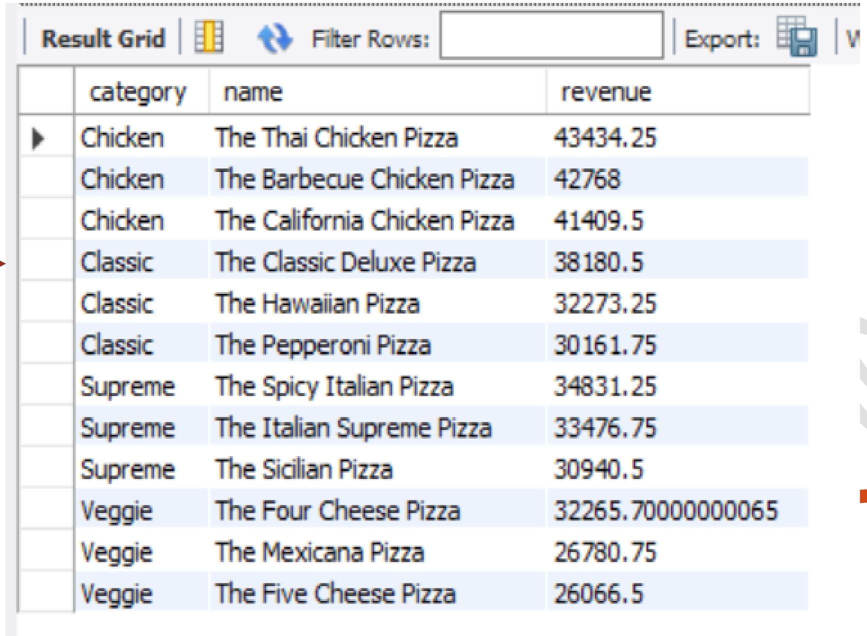
```
select
order_date, sum(revenue) over(order by order_date)
as cum_revenue from
(select
orders.order_date,
sum(order_details.quantity*pizzas.price) as revenue
from
pizzas join order_details
on pizzas.pizza_id=order_details.pizza_id
join orders
on orders.order_id=order_details.order_id
group by orders.order_date)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



```
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
2015-01-10   23990.350000000002
2015-01-11   25862.65
```

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

```
select category, name, revenue
from
(select
category, name, revenue,
rank() over(partition by category order by revenue desc)
as serialno 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 distribution)
as ranking where serialno <=3;
```



category	name	revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5
Veggie	The Four Cheese Pizza	32265.70000000065
Veggie	The Mexicana Pizza	26780.75
Veggie	The Five Cheese Pizza	26066.5

THANK YOU

ISRAT JAHAN UPAMA

+8801300919974

1911038@mme.buet.ac.bd

WWW.LINKEDIN.COM/IN/ISRATJUPAMA