PIZZA SALES ANALYSIS USING MYSQL

Agenda Style

01 BASIC ANALYSIS

- TOTAL NUMBER OF ORDERS
- TOTAL GENERATED REVENUE
- HIGHEST-PRICED PIZZA
- MOST COMMON PIZZA ORDERED SIZE
- TOP 5 MOST ORDERED PIZZA TYPES

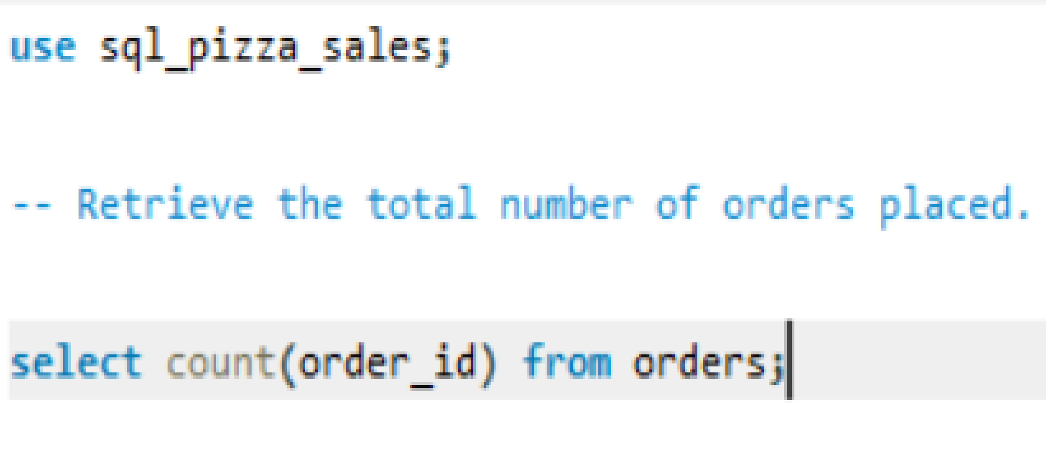
02 INTERMEDIATE ANALYSIS

- Total Quantity of Each Pizza Category Ordered
- Distribution of Orders by Hour of the Day
- Category-Wise Distribution of Pizzas
- Average Number of Pizzas Ordered Per Day
- Top 3 Most Ordered Pizza Types Based on Revenue

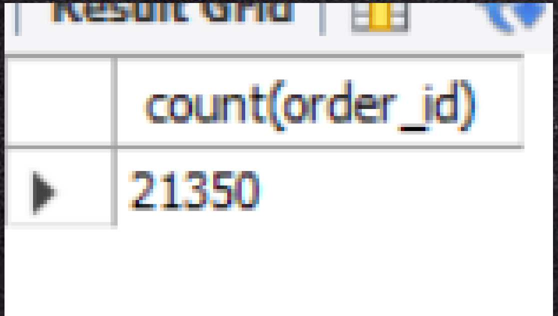
03 ADVANCE ANALYSIS

- Percentage Contribution of Each Pizza Type to Total Revenue
- Cumulative Revenue Generated Over Time
- Top 3 Most Ordered Pizza Types Based on Revenue for Each Category

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

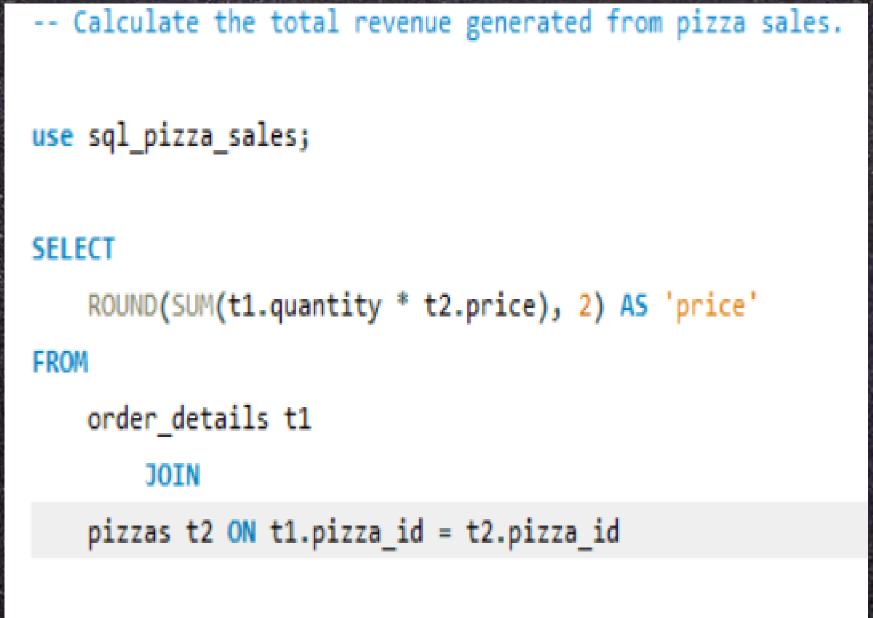


```sql
use sql_pizza_sales;


-- Retrieve the total number of orders placed.


select count(order_id) from orders;
```

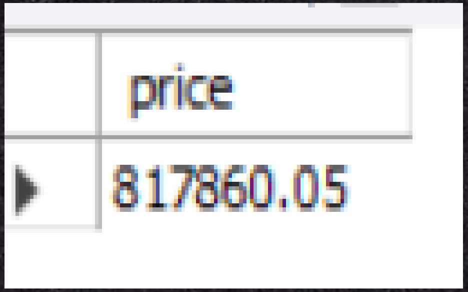


```
count(order_id)
21350
```

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

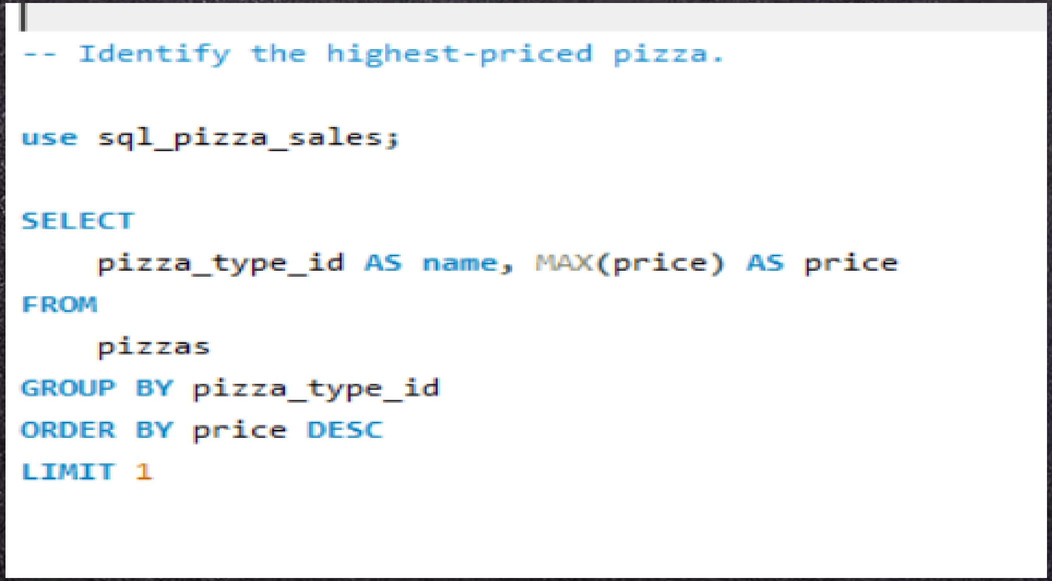


```
-- Calculate the total revenue generated from pizza sales.


use sql_pizza_sales;


SELECT
    ROUND(SUM(t1.quantity * t2.price), 2) AS 'price'
FROM
    order_details t1
        JOIN
    pizzas t2 ON t1.pizza_id = t2.pizza_id
```

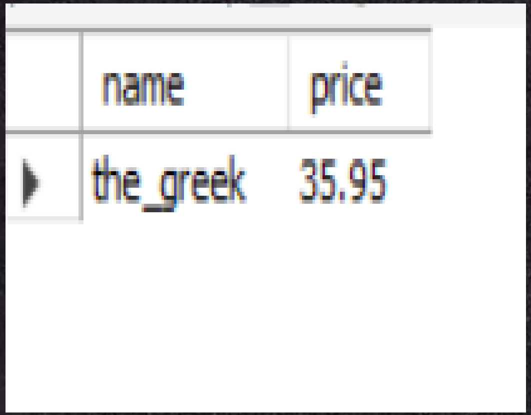


price
817860.05

IDENTIFY THE HIGHEST-PRICED PIZZA.

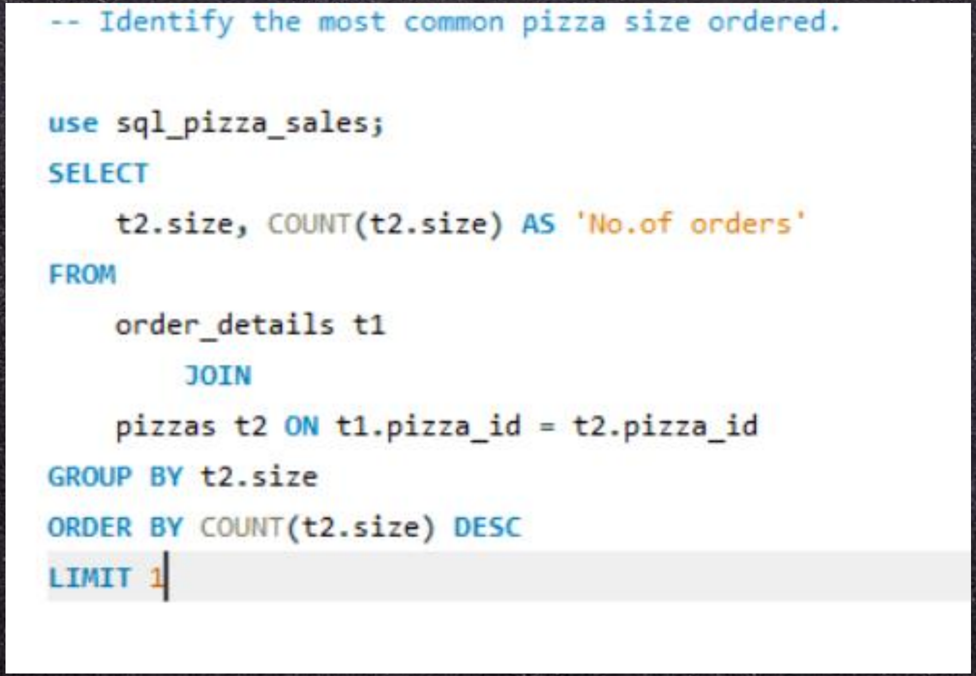


```
-- Identify the highest-priced pizza.

use sql_pizza_sales;

SELECT
    pizza_type_id AS name, MAX(price) AS price
FROM
    pizzas
GROUP BY pizza_type_id
ORDER BY price DESC
LIMIT 1
```

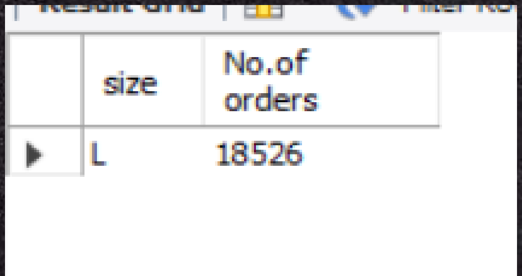


	name	price
▶	the_greek	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

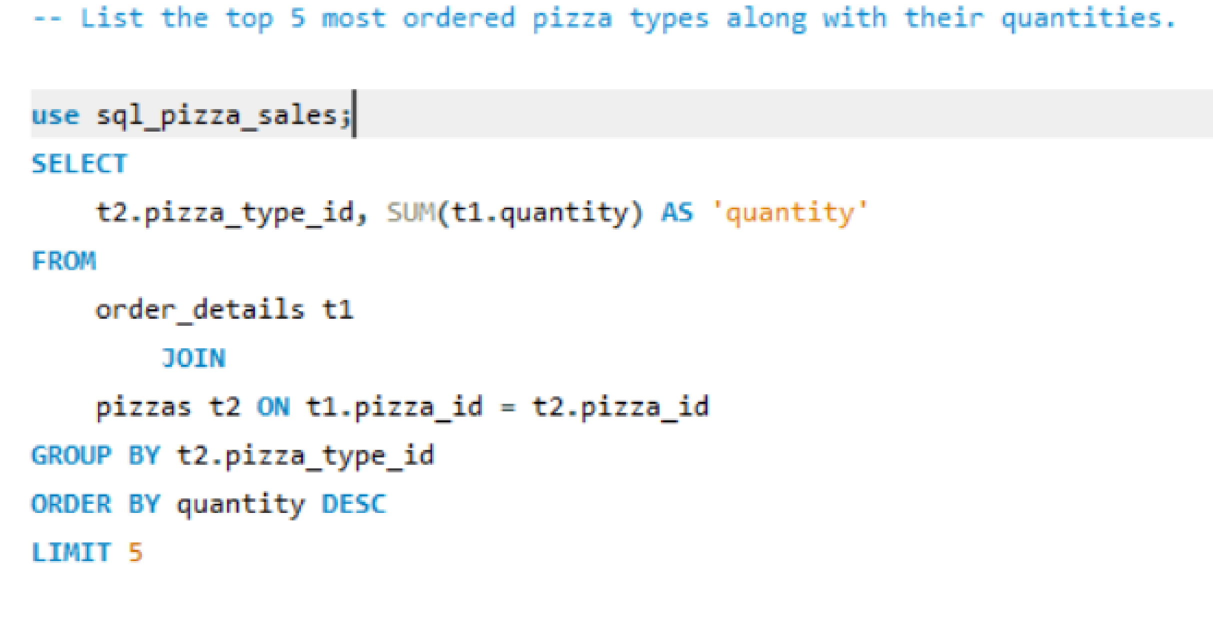


```
-- Identify the most common pizza size ordered.

use sql_pizza_sales;
SELECT
    t2.size, COUNT(t2.size) AS 'No.of orders'
FROM
    order_details t1
        JOIN
    pizzas t2 ON t1.pizza_id = t2.pizza_id
GROUP BY t2.size
ORDER BY COUNT(t2.size) DESC
LIMIT 1
```

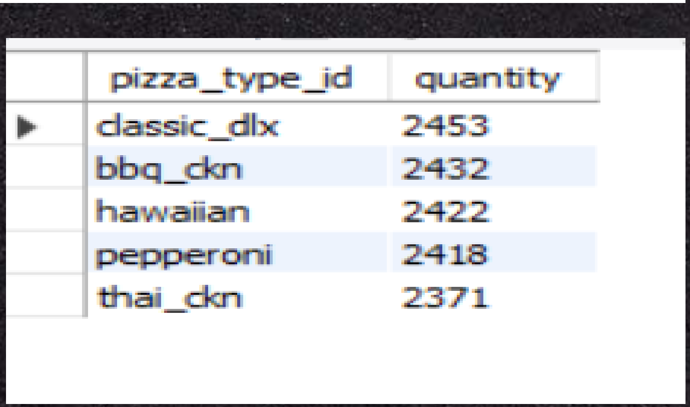


size	No.of orders
L	18526

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

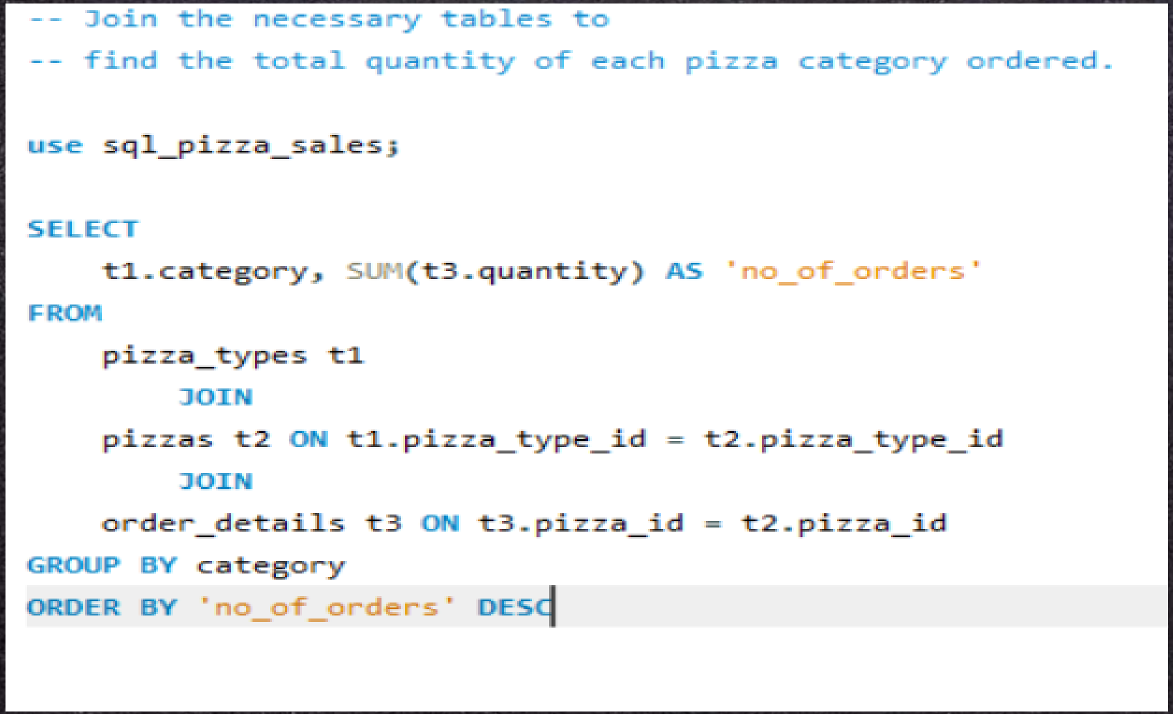


```sql
-- List the top 5 most ordered pizza types along with their quantities.

use sql_pizza_sales;
SELECT
    t2.pizza_type_id, SUM(t1.quantity) AS 'quantity'
FROM
    order_details t1
        JOIN
    pizzas t2 ON t1.pizza_id = t2.pizza_id
GROUP BY t2.pizza_type_id
ORDER BY quantity DESC
LIMIT 5
```

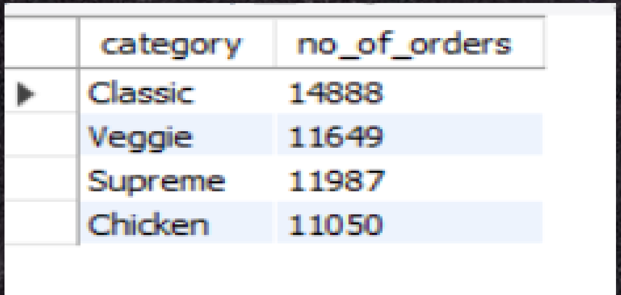


pizza_type_id	quantity
classic_dlx	2453
bbq_ckn	2432
hawaiian	2422
pepperoni	2418
thai_ckn	2371

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

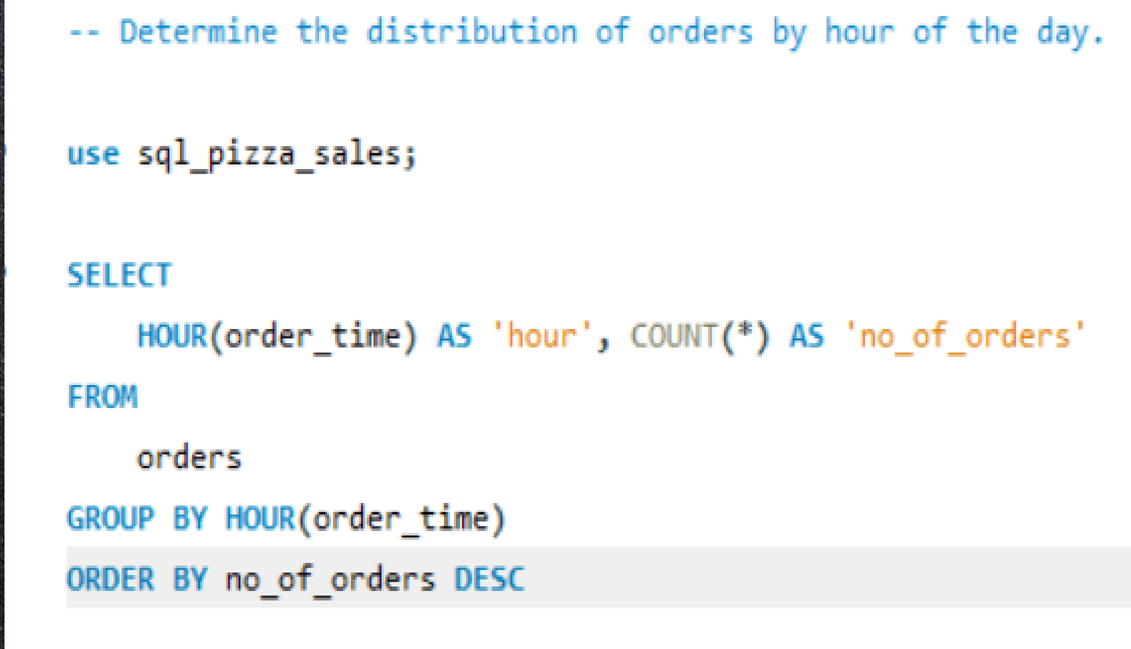


```sql
-- Join the necessary tables to
-- find the total quantity of each pizza category ordered.

use sql_pizza_sales;

SELECT
    t1.category, SUM(t3.quantity) AS 'no_of_orders'
FROM
    pizza_types t1
        JOIN
    pizzas t2 ON t1.pizza_type_id = t2.pizza_type_id
        JOIN
    order_details t3 ON t3.pizza_id = t2.pizza_id
GROUP BY category
ORDER BY 'no_of_orders' DESC
```

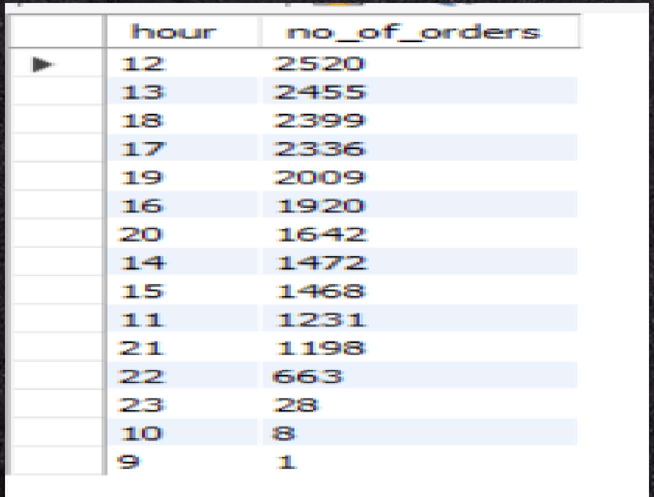


category	no_of_orders
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

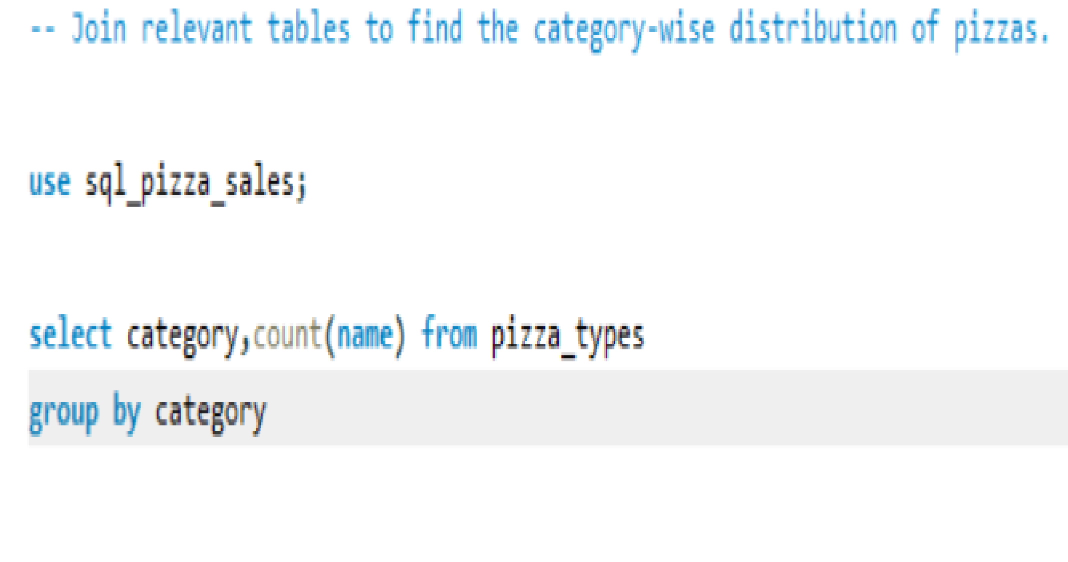


```
-- Determine the distribution of orders by hour of the day.


use sql_pizza_sales;


SELECT
    HOUR(order_time) AS 'hour', COUNT(*) AS 'no_of_orders'
FROM
    orders
GROUP BY HOUR(order_time)
ORDER BY no_of_orders DESC
```

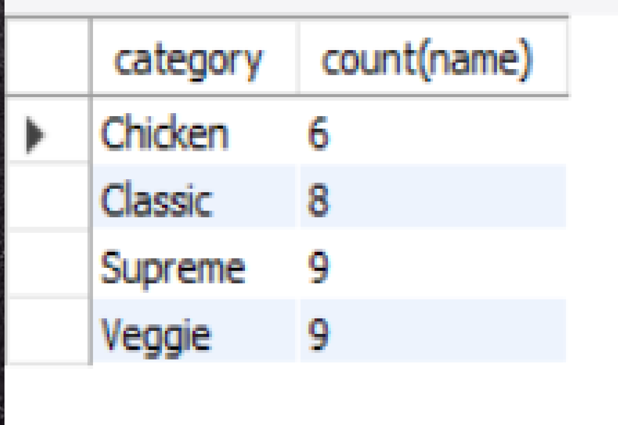


hour	no_of_orders
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231
21	1198
22	663
23	28
10	8
9	1

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

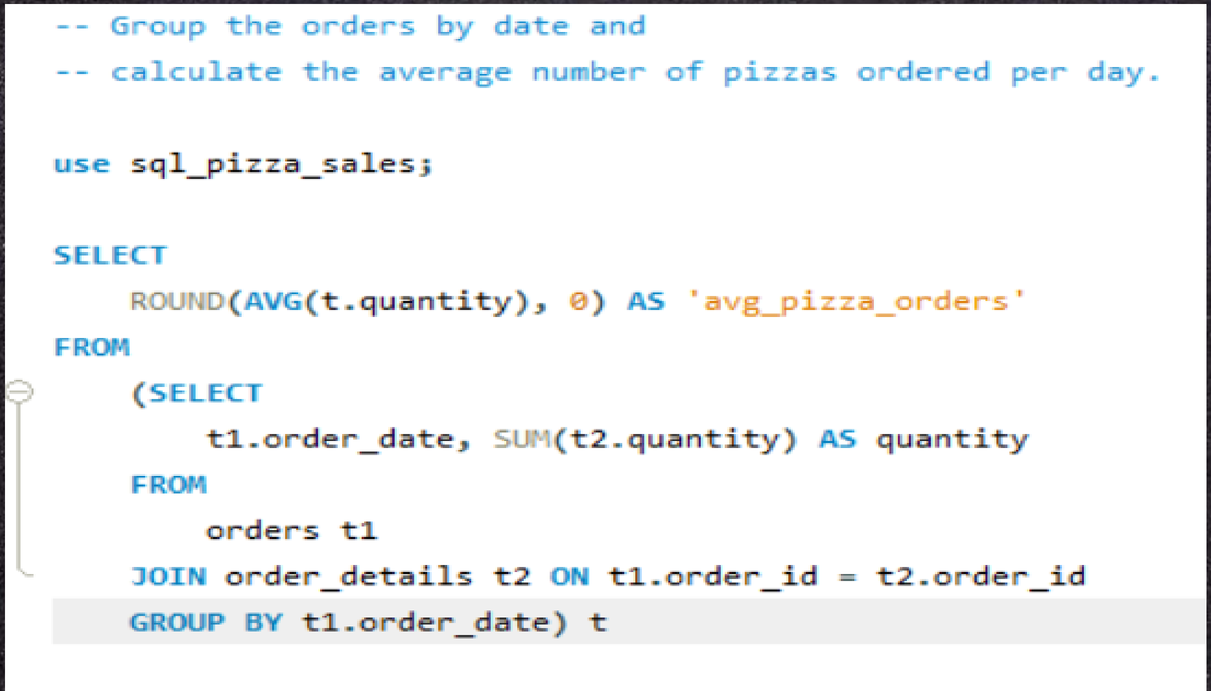


```
-- Join relevant tables to find the category-wise distribution of pizzas.


use sql_pizza_sales;


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

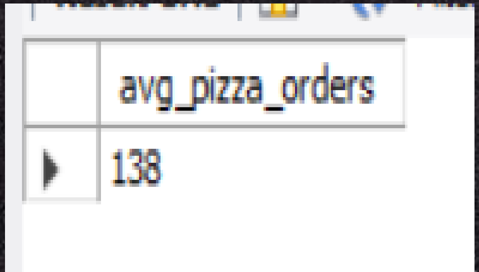


category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

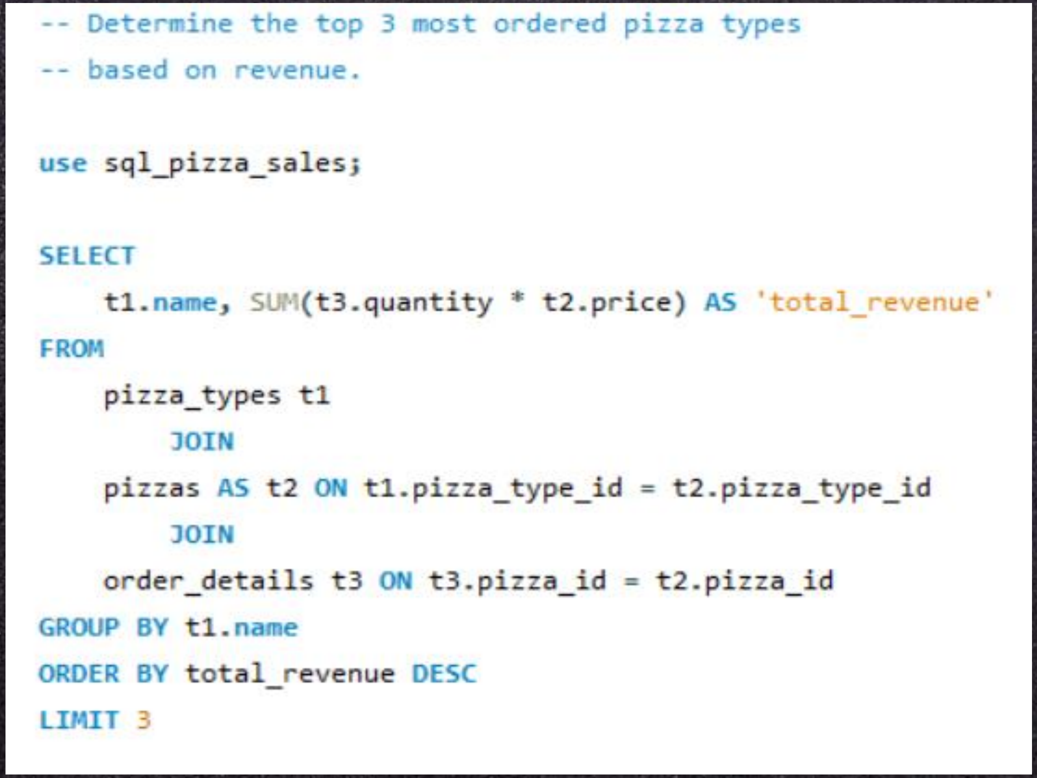


```
-- Group the orders by date and
-- calculate the average number of pizzas ordered per day.

use sql_pizza_sales;

SELECT
    ROUND(AVG(t.quantity), 0) AS 'avg_pizza_orders'
FROM
    (SELECT
        t1.order_date, SUM(t2.quantity) AS quantity
    FROM
        orders t1
    JOIN order_details t2 ON t1.order_id = t2.order_id
    GROUP BY t1.order_date) t
```

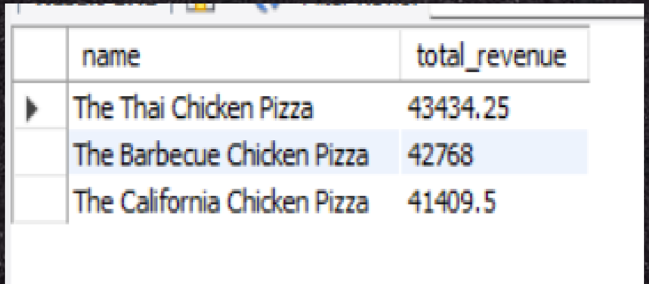


avg_pizza_orders
138

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

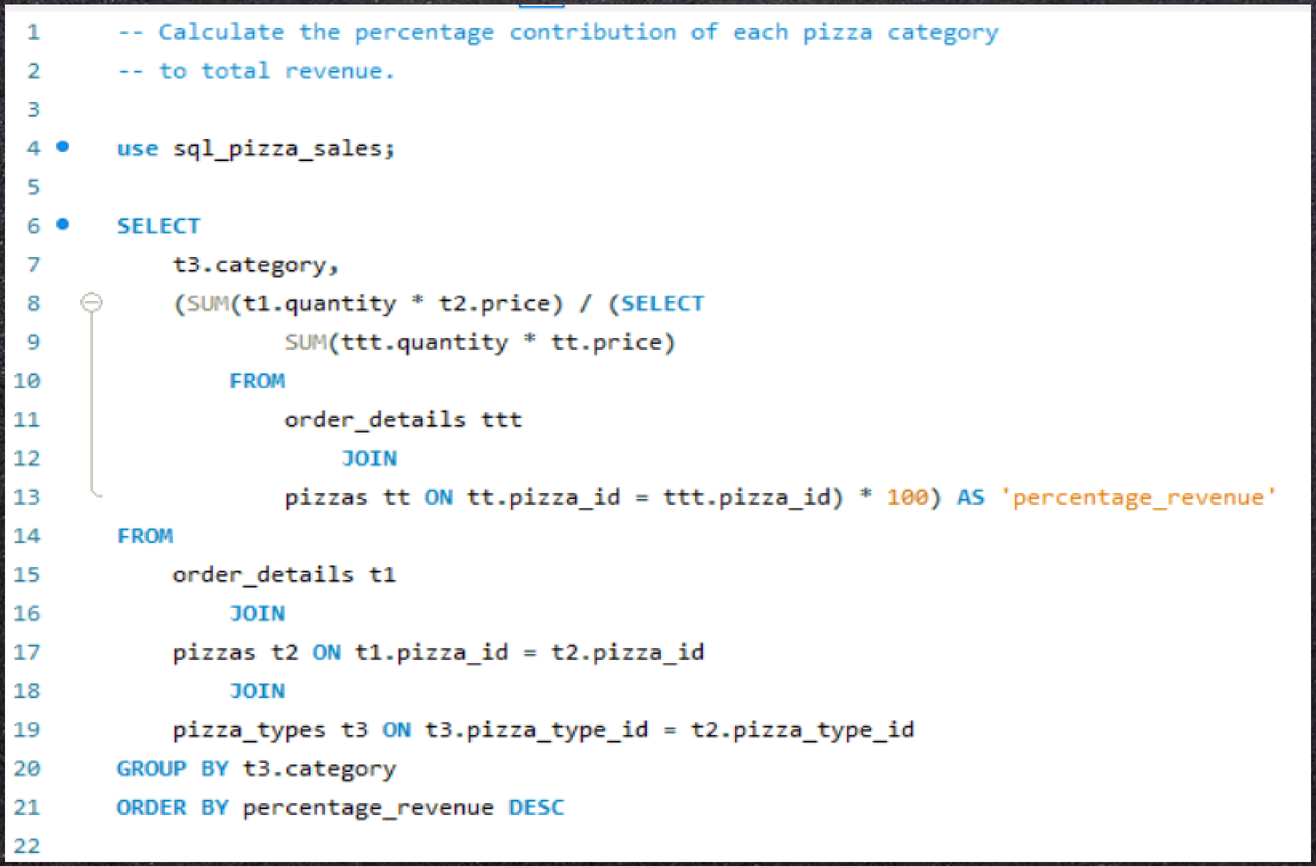


```
-- Determine the top 3 most ordered pizza types
-- based on revenue.

use sql_pizza_sales;

SELECT
    t1.name, SUM(t3.quantity * t2.price) AS 'total_revenue'
FROM
    pizza_types t1
        JOIN
    pizzas AS t2 ON t1.pizza_type_id = t2.pizza_type_id
        JOIN
    order_details t3 ON t3.pizza_id = t2.pizza_id
GROUP BY t1.name
ORDER BY total_revenue DESC
LIMIT 3
```

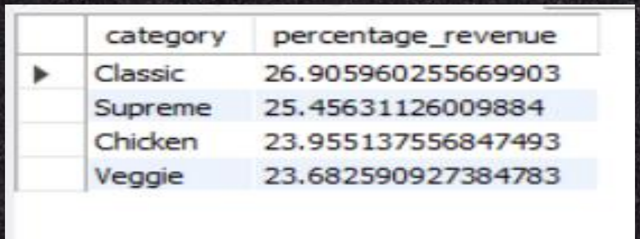


name	total_revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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

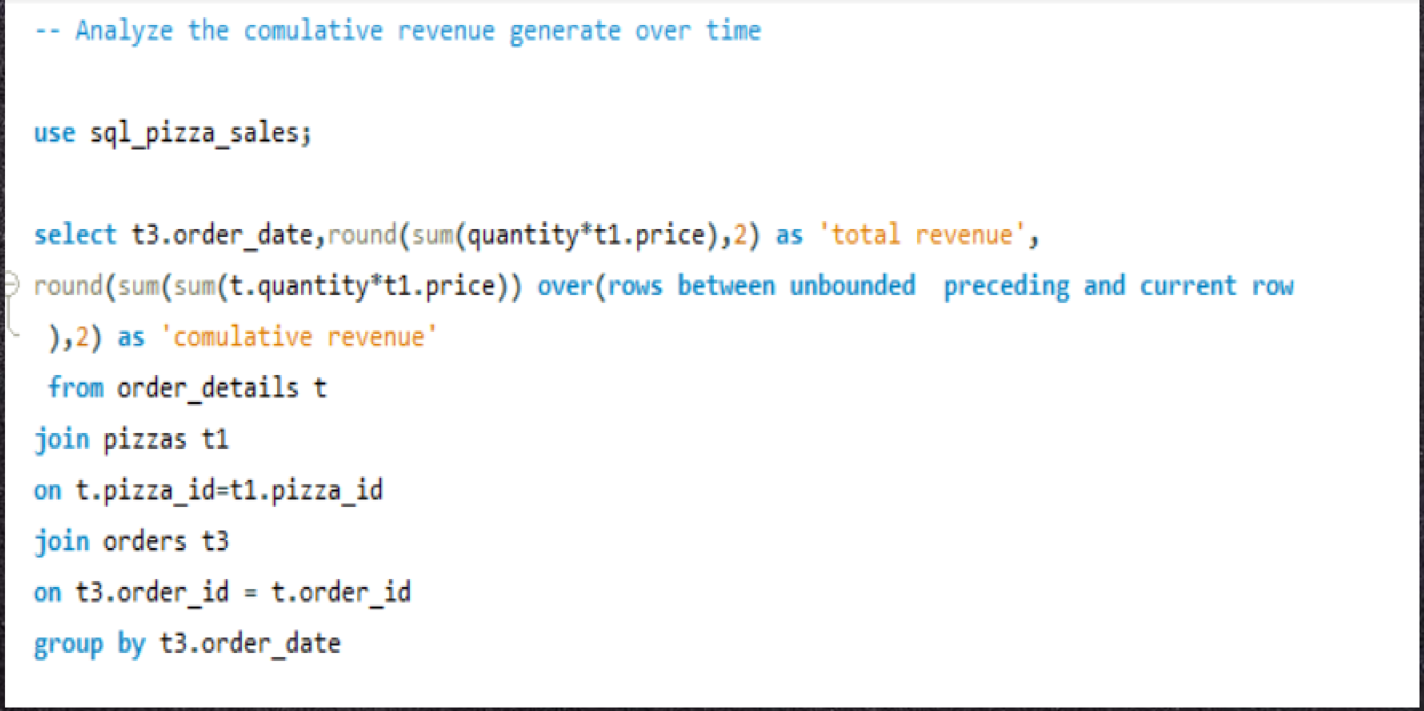


```sql
-- Calculate the percentage contribution of each pizza category
-- to total revenue.

use sql_pizza_sales;

SELECT
    t3.category,
    (SUM(t1.quantity * t2.price) / (SELECT
            SUM(ttt.quantity * tt.price)
        FROM
            order_details ttt
                JOIN
            pizzas tt ON tt.pizza_id = ttt.pizza_id) * 100) AS 'percentage_revenue'
FROM
    order_details t1
        JOIN
    pizzas t2 ON t1.pizza_id = t2.pizza_id
        JOIN
    pizza_types t3 ON t3.pizza_type_id = t2.pizza_type_id
GROUP BY t3.category
ORDER BY percentage_revenue DESC

```

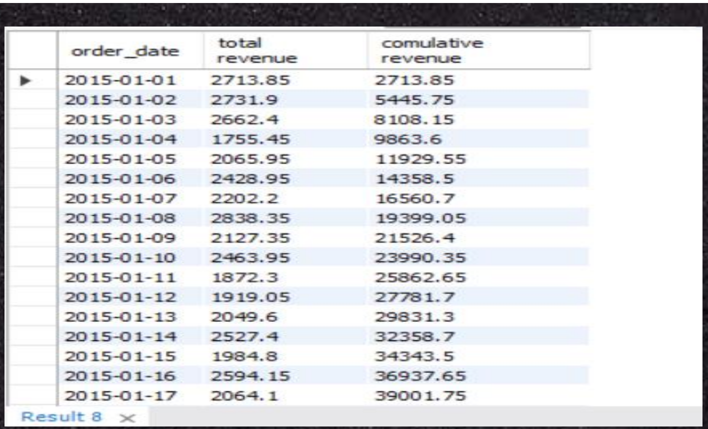


category	percentage_revenue
Classic	26.905960255669903
Supreme	25.45631126009884
Chicken	23.955137556847493
Veggie	23.682590927384783

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.



```
-- Analyze the comulative revenue generate over time


use sql_pizza_sales;


select t3.order_date,round(sum(quantity*t1.price),2) as 'total revenue',
round(sum(sum(t.quantity*t1.price)) over(rows between unbounded  preceding and current row
 ),2) as 'comulative revenue'
 from order_details t
join pizzas t1
on t.pizza_id=t1.pizza_id
join orders t3
on t3.order_id = t.order_id
group by t3.order_date
```



order_date	total revenue	comulative revenue
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15
2015-01-04	1755.45	9863.6
2015-01-05	2065.95	11929.55
2015-01-06	2428.95	14358.5
2015-01-07	2202.2	16560.7
2015-01-08	2838.35	19399.05
2015-01-09	2127.35	21526.4
2015-01-10	2463.95	23990.35
2015-01-11	1872.3	25862.65
2015-01-12	1919.05	27781.7
2015-01-13	2049.6	29831.3
2015-01-14	2527.4	32358.7
2015-01-15	1984.8	34343.5
2015-01-16	2594.15	36937.65
2015-01-17	2064.1	39001.75

THANK YOU

yogeshchouhan263@gmail.com