# PIZZA

## PIZZA SALES REPORT

# WELCOME TO OUR PIZZA SALES REPORT

Hello i am sagar sankadal in this  Pizza sales report project i have utilized SQL Queries to solve problems that were related to  pizza sales of the year

# In this project i solved these problems

1. retrieve the total number of orders placed
2. calculate the total revenue generatedfrom th pizzas sales
3. identify the highest-priced pizza
4. identify the most common pizza size ordered
5. list the top 5 most ordered pizzas type along with their quantities
6. join the necessary tables to find the total quantity of each pizza category order
7. deteremine the distribution of the order by hour of the day
8. join the relevant tables to find the category wise distribution of pizzas.
9. group the orders by date and calculate the average number of pizzas ordered per day
10. determine the top 3 most ordered pizza types based on the revenue
11. calculate the percentage contributiobn of each pizza type of total revenue
12. analyze the cumulative revenue generated over time
13. determine the top 3 most ordered pizzas based on the revewnue of each pizza category

# 01.RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

```sql
-- retrieve the total number of  orders placed

select count(order_id)from orders;
```

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

| count(order_id) |
| --- |
| 21350 |

# 02.CALCULATE THE TOTAL REVENUE GENERATEDFROM THE PIZZAS SALE

```sql
-- calculate the total revenue generatedfrom th pizzas sales
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
```

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

| total_sales |
| --- |
| 817860.05 |

# 03.IDENTIFY THE HIGHEST-PRICED PIZZA

```sql
-- 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;
```

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

| name | price |
|---|---|
| The Greek Pizza | 35.95 |

# 04.IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
-- identify the most common pizza size ordered
SELECT
    quantity, COUNT(order_details_id)
FROM
    order_details
GROUP BY quantity
```

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

| quantity | COUNT(order_details_id) |
|---|---|
| 1 | 47693 |
| 2 | 903 |
| 3 | 21 |
| 4 | 3 |

# 05. LIST THE TOP 5 MOST ORDERED PIZZAS TYPE ALONG WITH THEIR QUANTITIES

```sql
-- list the top 5 most ordered pizzas type 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 |

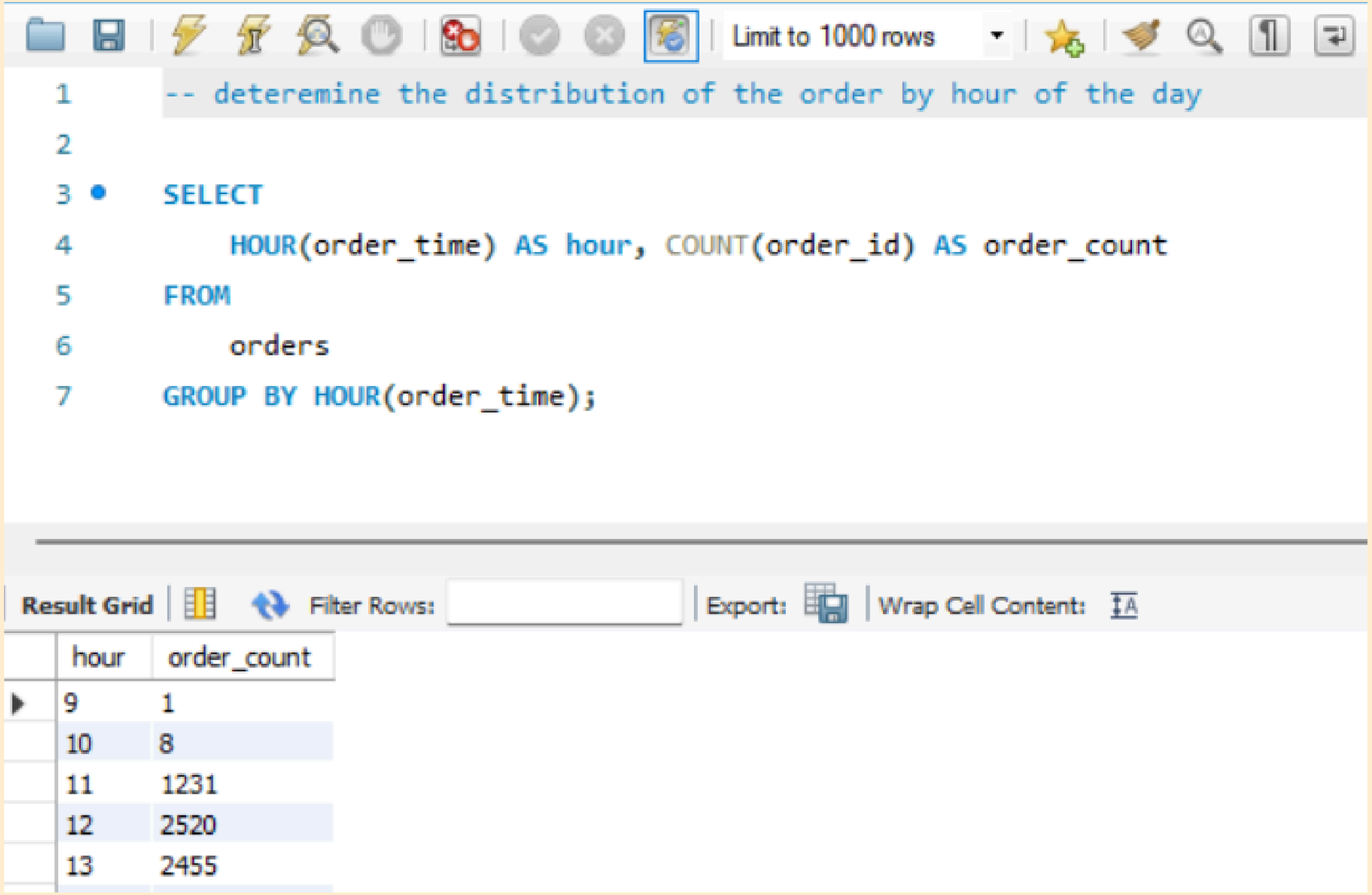
# 06.JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDER

```sql
-- join  the necessary tables to find the total quantity of each pizza category o
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS total_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 total_quantity DESC;
```

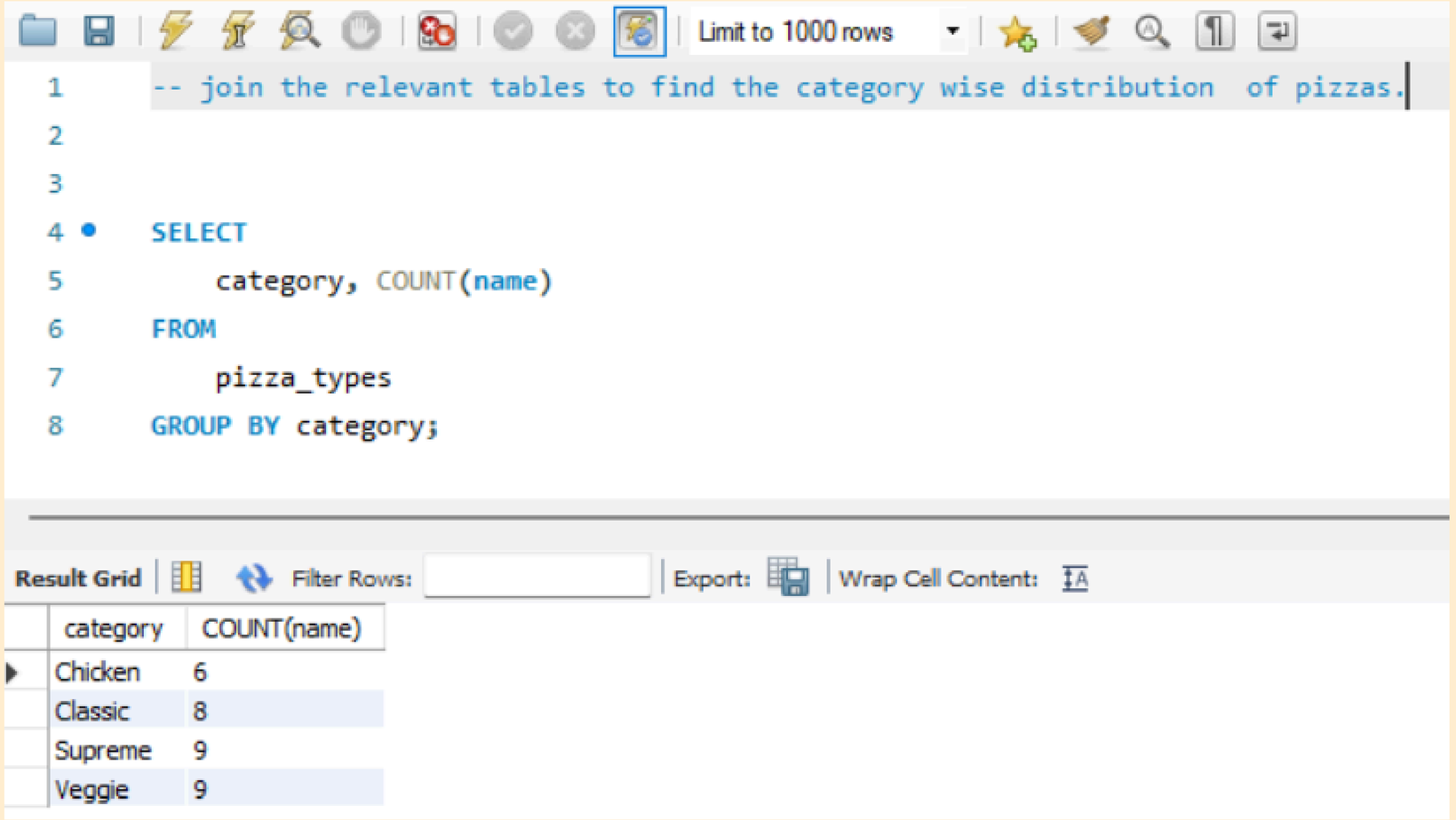
Result Grid | Filter Rows:

| category | total_quantity |
|---|---|
| Classic | 14888 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

# 07. DETEREMINE THE DISTRIBUTION OF THE ORDER BY HOUR OF THE DAY



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



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

```sql
-- group the orders by date and calculate the average  number of pizzas ordered per

SELECT
    ROUND(AVG(quantity), 0) as avg_pizza_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) AS order_quantity;
```

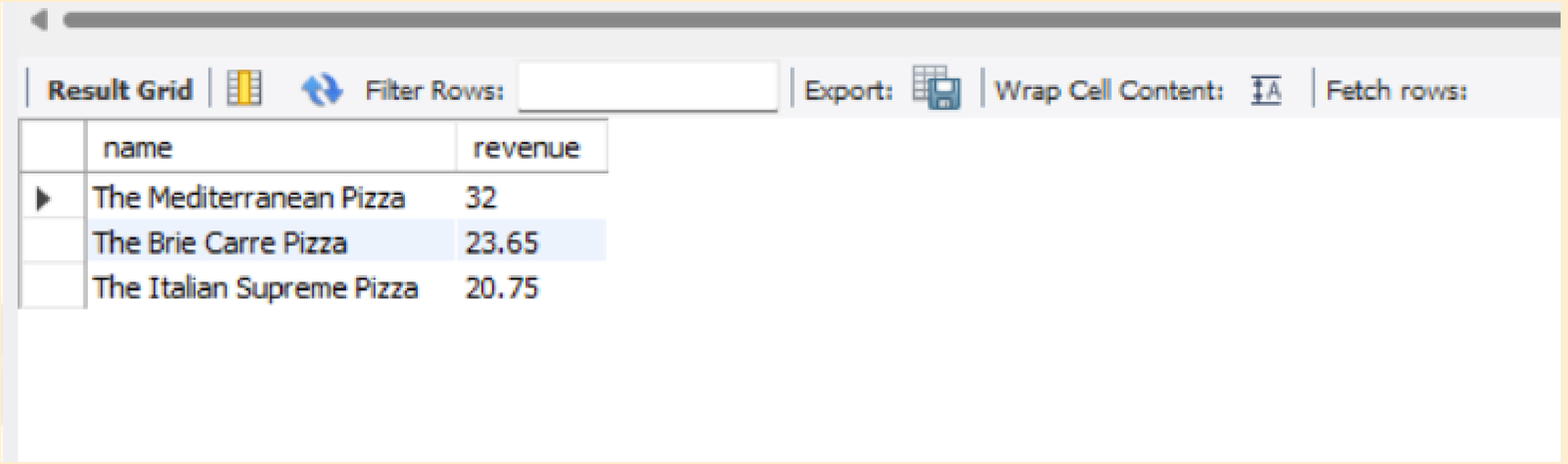
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| avg_pizza_per_day |
|---|
| 138 |

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

```sql
-- determine the top 3 most ordered  pizza types based on the revenue

SELECT
    pizza_types.name,
    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;
```



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

| name | revenue |
|---|---|
| The Mediterranean Pizza | 32 |
| The Brie Carre Pizza | 23.65 |
| The Italian Supreme Pizza | 20.75 |

# 11.CALCULATE THE PERCENTAGE CONTRIBUTIOBN OF EACH PIZZA TYPE OF TOTAL REVENUE

```sql
-- calculate the percentage contributiobn of each pizza type of 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 Rows: | Export: | Wrap Cell Content:

| category | revenue |
|---|---|
| Classic | 26.91 |
| Supreme | 25.46 |
| Chicken | 23.96 |
| Veggie | 23.68 |

# 12.ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

```sql
-- analyze the cumulative revenue generated over time
select order_date,sum(revenue) over(order by sales.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: | Export: | Wrap Cell Content:

| 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 |

# 13.DETERMINE THE TOP 3 MOST ORDERED PIZZAS BASED ON THE REVENUE OF EACH PIZZA CATEGORY

```sql
select name,revenue from
(select category,name,revenue,
rank() over(partition by category order by revenue desc) as ranks
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 ranks<=3;
```

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

| 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 |