

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 generated from the 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. determine 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 contribution 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 revenue of each pizza category

01.RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

```
1  -- retrieve the total number of orders placed
2
3 • select count(order_id)from orders;
```

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

count(order_id)
21350

02.CALCULATE THE TOTAL REVENUE GENERATED FROM THE PIZZAS SALE

```
1      -- calculate the total revenue generated from the pizzas sales
2 •  SELECT
3   ⌈   ROUND(SUM(order_details.quantity * pizzas.price),
4           2) AS total_sales
5   FROM
6       order_details
7   JOIN
8       pizzas ON pizzas.pizza_id = order_details.pizza_id
9
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
total_sales			
817860.05			

03.IDENTIFY THE HIGHEST-PRICED PIZZA

```
1      -- identify the highest-priced pizza
2 •  SELECT
3      pizza_types.name, pizzas.price
4  FROM
5      pizza_types
6      JOIN
7      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8  ORDER BY pizzas.price DESC
9  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

```
1  -- identify the most common pizza size ordered
2  SELECT
3      quantity, COUNT(order_details_id)
4  FROM
5      order_details
6  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

```
1      -- list the top 5 most ordered pizzas type along with their quantities
2
3 •  SELECT
4      pizza_types.name, SUM(order_details.quantity) AS quantity
5  FROM
6      pizza_types
7      JOIN
8          pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9      JOIN
10         order_details ON order_details.pizza_id = pizzas.pizza_id
11     GROUP BY pizza_types.name
12     ORDER BY quantity DESC
13     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

```
1 -- join the necessary tables to find the total quantity of each pizza category o
2 • SELECT
3     pizza_types.category,
4     SUM(order_details.quantity) AS total_quantity
5 FROM
6     pizza_types
7     JOIN
8     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9     JOIN
10    order_details ON order_details.pizza_id = pizzas.pizza_id
11 GROUP BY pizza_types.category
12 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

The screenshot shows a MySQL Workbench interface. At the top, there's a toolbar with various icons for database management. Below the toolbar, a query editor window contains the following SQL code:

```
1 -- deteremine the distribution of the order by hour of the day
2
3 • SELECT
4     HOUR(order_time) AS hour, COUNT(order_id) AS order_count
5 FROM
6     orders
7 GROUP BY HOUR(order_time);
```

Below the query editor is a results pane. It has tabs for "Result Grid" and "Text". The "Result Grid" tab is selected, showing a table with two columns: "hour" and "order_count". The data is as follows:

	hour	order_count
▶	9	1
	10	8
	11	1231
	12	2520
	13	2455

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

The screenshot shows a MySQL Workbench interface. The top bar includes standard icons for file operations, search, and refresh, followed by a toolbar with specific functions like 'Limit to 1000 rows'. Below the toolbar is a text area containing a SQL query:

```
1 -- join the relevant tables to find the category wise distribution of pizzas.
2
3
4 • SELECT
5     category, COUNT(name)
6 FROM
7     pizza_types
8 GROUP BY category;
```

Below the query is a 'Result Grid' section with the following data:

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

09. 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
```

- **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

```
1      -- determine the top 3 most ordered pizza types based on the revenue
2
3 •  SELECT
4      pizza_types.name,
5      order_details.quantity * pizzas.price AS revenue
6  FROM
7      pizza_types
8          JOIN
9      pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
10         JOIN
11     order_details ON order_details.pizza_id = pizzas.pizza_id
12  GROUP BY pizza_types.name
13  ORDER BY revenue DESC
14  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 CONTRIBUTION OF EACH PIZZA TYPE OF TOTAL REVENUE

```
1  -- calculate the percentage contribution of each pizza type of total revenue
2
3 • SELECT
4     pizza_types.category,
5     ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
6             ROUND(SUM(order_details.quantity * pizzas.price),
7                 2) AS total_sales
8
9         FROM
10            order_details
11            JOIN
12                pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
13                2) AS revenue
14
15     FROM
16        pizza_types
17        JOIN
18            pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
19        JOIN
20            order_details ON order_details.pizza_id = pizzas.pizza_id
21
22    GROUP BY pizza_types.category
23
24    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

```
1  -- analyze the cumulative revenue generated over time
2 • select order_date,sum(revenue) over(order by sales.order_date) as cum_revenue
3   from
4   (select orders.order_date,
5    sum(order_details.quantity * pizzas.price) as revenue
6    from order_details join pizzas
7    on order_details.pizza_id = pizzas.pizza_id
8    join orders
9    on orders.order_id = order_details.order_id
10   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
	2015-01-07	16886.5

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

```
3      select name,revenue from
4      (select category,name,revenue,
5       rank() over(partition by category order by revenue desc) as ranks
6       from
7      (select pizza_types.category, pizza_types.name,
8       sum(order_details.quantity* pizzas.price) as revenue
9       from pizza_types join pizzas
10      on pizza_types.pizza_type_id =pizzas.pizza_type_id
11      join order_details
12      on order_details.pizza_id = pizzas.pizza_id
13      group by pizza_types.category, pizza_types.name)as a)as b
14      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