SQL
Project

PIZZA HUT

- WHERE EVERY SLICE TELLS A STORY

SQL
Project

ABOUT PROJECT

📌 OVERALL SUMMARY

- The project is a Pizza Hut sales analysis using SQL.
- It progresses from simple counts and totals → grouping & joins → advanced revenue insights.
- Business insights you'd get from this:
- How many orders and total revenue.
- Best-selling pizza sizes and types.
- Which hours are busiest.
- Which categories perform best.
- Revenue contribution by product (good for pricing/marketing).
- Long-term growth trend via cumulative revenue.
-

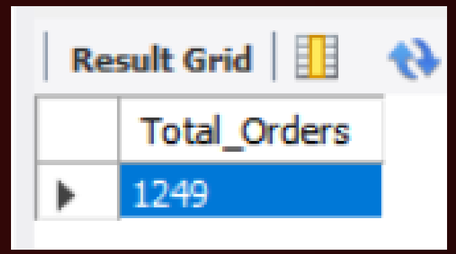
SQL
Project

1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

QUERY :

```
SELECT
    COUNT(Order_id) AS Total_Orders
FROM
    orders;
```

RESULT :



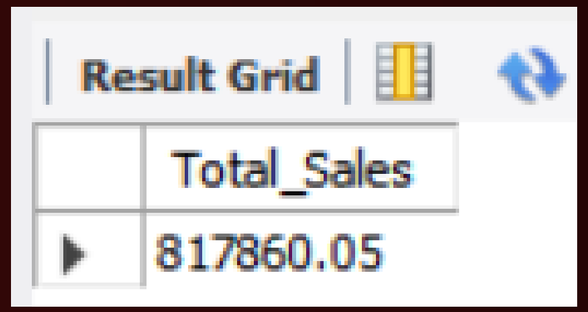
SQL
Project

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

QUERY :

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

RESULT :



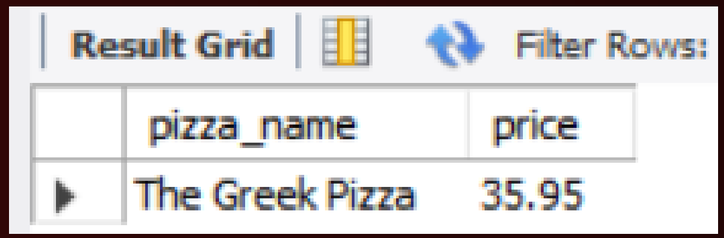
SQL
Project

3. IDENTIFY THE HIGHEST-PRICED PIZZA

QUERY :

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

RESULT :


Result Grid | Filter Rows:

	pizza_name	price
▶	The Greek Pizza	35.95

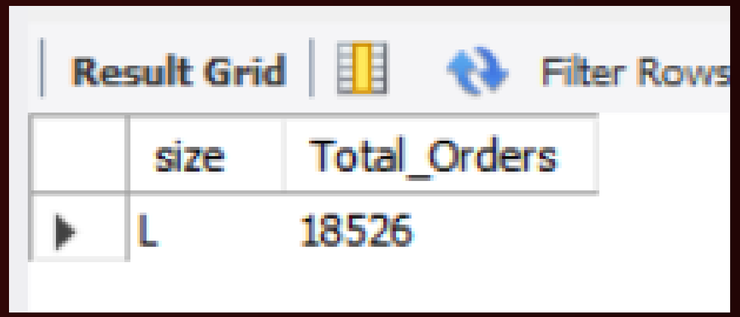
SQL
Project

4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

QUERY :

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

RESULT :



	size	Total_Orders
▶	L	18526

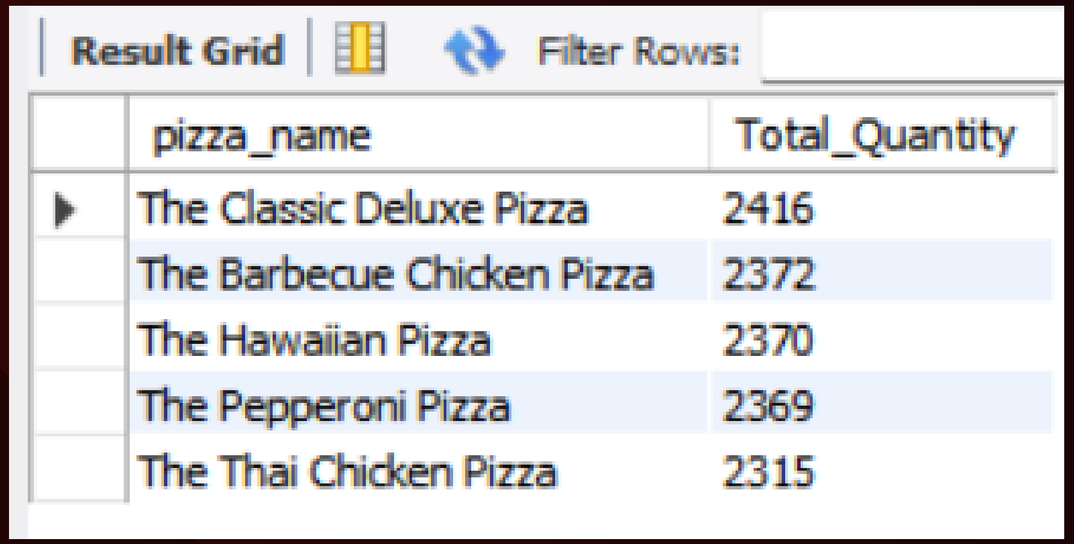
SQL
Project

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

QUERY :

```
SELECT
    pizza_types.pizza_name,
    COUNT(order_details.quantity) AS Total_Quantity
FROM
    pizza_types
        JOIN
    pizzas 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.pizza_name
ORDER BY Total_Quantity DESC
LIMIT 5;
```

RESULT :


Result Grid | Filter Rows:

pizza_name	Total_Quantity
The Classic Deluxe Pizza	2416
The Barbecue Chicken Pizza	2372
The Hawaiian Pizza	2370
The Pepperoni Pizza	2369
The Thai Chicken Pizza	2315

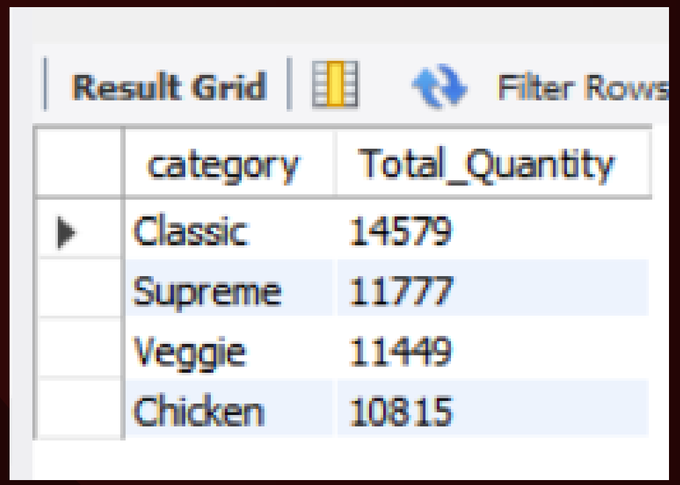
SQL
Project

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

QUERY :

```
SELECT
    pizza_types.category,
    COUNT(order_details.quantity) AS Total_Quantity
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.category
ORDER BY Total_Quantity DESC;
```

RESULT :



	category	Total_Quantity
▶	Classic	14579
	Supreme	11777
	Veggie	11449
	Chicken	10815

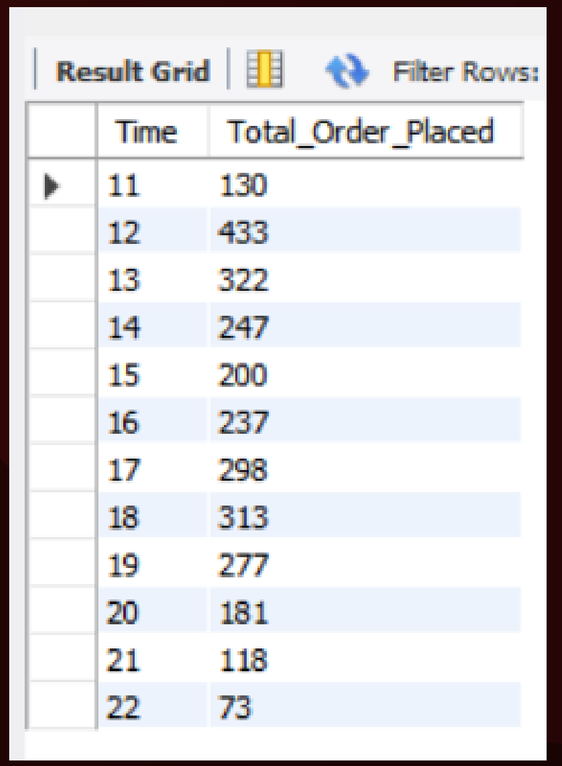
SQL
Project

7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

QUERY :

```
SELECT
    HOUR(orders.order_time) as Time,
    COUNT(order_details.order_details_id) AS
Total_Order_Placed
FROM
    Orders
        JOIN
    order_details ON orders.order_id =
order_details.order_id
GROUP BY Time;
```

RESULT :


Result Grid | Filter Rows:

Time	Total_Order_Placed
11	130
12	433
13	322
14	247
15	200
16	237
17	298
18	313
19	277
20	181
21	118
22	73

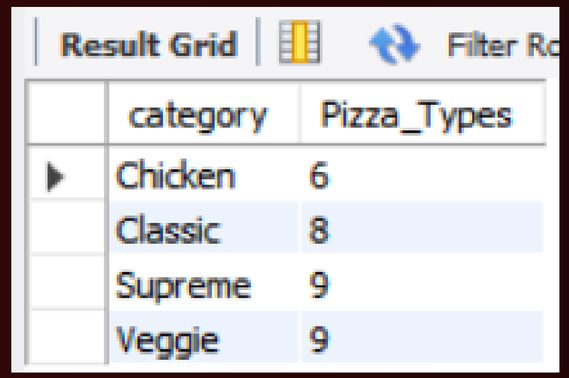
SQL
Project

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

QUERY :

```
SELECT
    category, COUNT(pizza_name) AS Pizza_Types
FROM
    pizza_types
GROUP BY category;
```

RESULT :


Result Grid | Filter Rc

	category	Pizza_Types
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

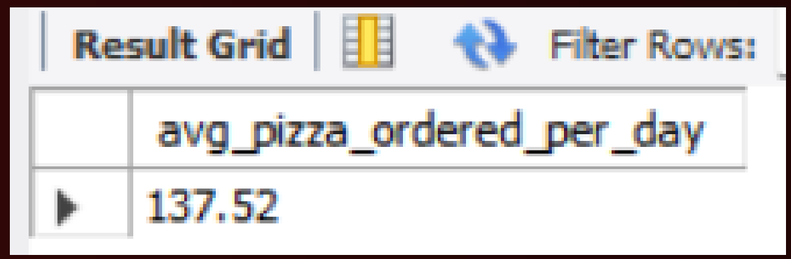
SQL
Project

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

QUERY :

```
SELECT
    ROUND(AVG(quantity), 2) AS
avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS
quantity
    FROM
        order_details
    JOIN orders ON orders.order_id =
order_details.order_id
    GROUP BY orders.order_date) AS
Total_Pizza_Quantity;
```

RESULT :



SQL
Project

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

QUERY :

```
SELECT
    pizza_types.pizza_name,
    SUM(pizzas.price * order_details.quantity) 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.pizza_name
ORDER BY Revenue DESC
LIMIT 3;
```

RESULT :

Result Grid | Filter Rows:

pizza_name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

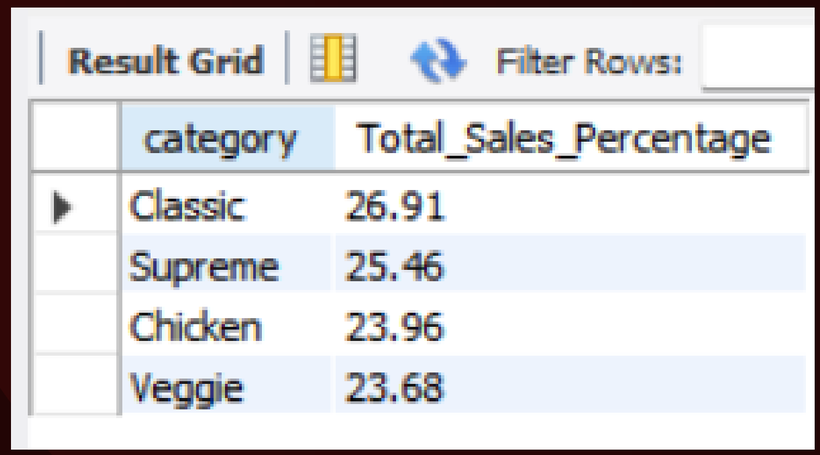
SQL
Project

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

QUERY :

```
SELECT
    pizza_types.category,
    ROUND(SUM(pizzas.price * order_details.quantity) / (SELECT
                ROUND(SUM(pizzas.price * order_details.quantity),
                        2) AS Total_Sales
            FROM
                pizzas
                    JOIN
                order_details ON order_details.pizza_id = pizzas.pizza_id) * 100,
            2) AS Total_Sales_Percentage
FROM
    pizzas
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_types.category
order by Total_Sales_Percentage desc;
```

RESULT :



category	Total_Sales_Percentage
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

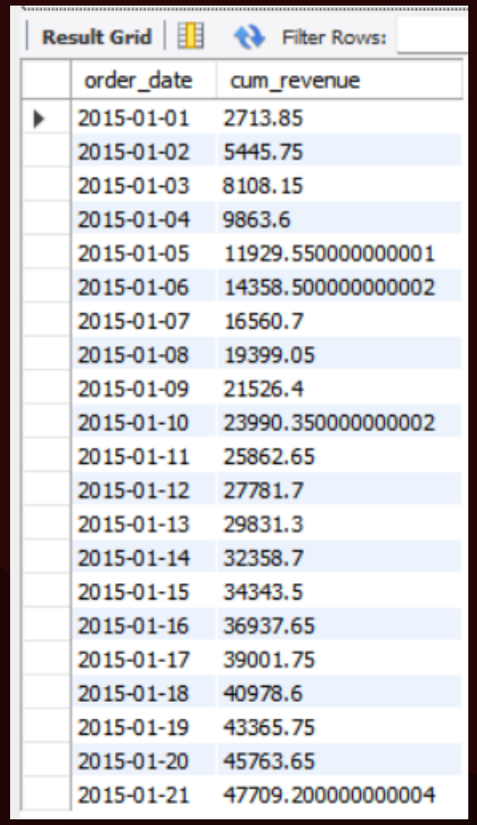
SQL
Project

12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

QUERY :

```
SELECT order_date, sum(Total_sales) OVER (order by
order_date) AS cum_revenue
FROM
 (SELECT orders.order_date, sum(order_details.quantity *
pizzas.price) AS Total_sales
 FROM orders
 JOIN order_details ON orders.order_id =
order_details.order_id
 JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id
GROUP BY orders.order_date) AS sales;
```

RESULT :



Result Grid | Filter Rows:

order_date	cum_revenue
2015-01-01	2713.85
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.550000000001
2015-01-06	14358.500000000002
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
2015-01-12	27781.7
2015-01-13	29831.3
2015-01-14	32358.7
2015-01-15	34343.5
2015-01-16	36937.65
2015-01-17	39001.75
2015-01-18	40978.6
2015-01-19	43365.75
2015-01-20	45763.65
2015-01-21	47709.200000000004

SQL
Project

13. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY

QUERY :

```
SELECT category, pizza_name , Revenue FROM
(SELECT category, pizza_name , Revenue,
RANK() over (PARTITION BY category ORDER BY Revenue DESC) AS
rn
FROM
(SELECT pizza_types.category, pizza_types.pizza_name,
   SUM(pizzas.price * order_details.quantity) 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.category, pizza_types.pizza_name) AS a)
AS b
WHERE rn <= 3;
```

RESULT :

Result Grid | Filter Rows: | Export:

category	pizza_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

SQL
Project

THANK YOU

FOR ATTENTION