Where Every Slice is a Taste of Perfection

WELCOME TO PIZZA RESTO

ORDER NOW

Start Your Slide

HELLO !

This report demonstrates my hold in basic to advance SQL techniques, including aggregation, joins, window functions, and subqueries, applied to analyzing pizza sales data. The analysis identifies trends in pizza types, order volumes by time, and revenue generation, providing actionable insights for business decision-making.

Q1 - CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```sql
select round(sum(orders_details.qunatity * pizzas.price), 2) as Total_Price
from orders_details join pizzas on pizzas.pizza_id=orders_details.pizza_id;
```

Result Grid

Total_Price
817860.05

Q 2 - IDENTIFY THE HIGHEST-PRICED PIZZA

```sql
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 Row

name	price
The Greek Pizza	35.95

Q 3 - IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

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

Result Grid | Filter

size	order_count
L	18526

Q 4 - LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES

```sql
SELECT pizza_types.name, SUM(orders_details.qunatity) AS quantity
FROM pizza_types
JOIN
pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN
orders_details ON orders_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

Q 5 - 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);
```

Result Grid

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468

Q 6 - GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY

```sql
SELECT ROUND(AVG(quantity), 0) AS avg_pizzas_ordered_per_day
FROM
(SELECT orders.order_date, SUM(orders_details.qunatity) AS quantity
FROM
orders
JOIN orders_details ON orders.order_id = orders_details.order_id
GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filter Rows:

avg_pizzas_ordered_per_day
138

Q 7 - ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

```
select order_date, sum(revenue) over (order by order_date) as cum_rev from
(select orders.order_date, SUM(orders_details.qunatity * pizzas.price) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders on orders.order_id = orders_details.order_id
group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

order_date	cum_rev
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

Q 8 - DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY

```sql
select name, revenue from
(select category, name, revenue, rank() over(partition by category order by revenue desc ) as rn from
(select pizza_types.category, pizza_types.name,
sum((orders_details.qunatity) * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category, pizza_types.name) as a) as b where rn <= 3;
```

Result Grid | Filter Rows:

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

Q 9 -FIND THE TOTAL QUANTITY OF PIZZAS SOLD PER ORDER AND THE ORDER WITH THE HIGHEST TOTAL QUANTITY SOLD.

```sql
WITH TotalQuantity AS (
    SELECT orders.order_id,
    SUM(orders_details.qunatity) AS total_quantity
    FROM orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_id
)
SELECT order_id, total_quantity
FROM TotalQuantity
WHERE total_quantity = (SELECT MAX(total_quantity) FROM TotalQuantity);
```

Result Grid | Filter Rows:

order_id	total_quantity
18845	28