PIZZAHUT SALE

SQL PROJECT

ABOUT

This project provides SQL-based analysis of pizza sales data to uncover sales patterns, popular items, and customer preferences. Key analysis includes:

- Sales Performance: Querying total sales by day, week, and month to identify peak periods.
- Item Popularity: Ranking pizzas by popularity based on order frequency and revenue.
- Customer Insights: Analyzing repeat customer behavior and purchase trends.
- Profitability: Calculating profit margins per pizza type and optimizing pricing strategies.

The repository includes SQL scripts and insights drawn from the analysis, useful for anyone interested in SQL data exploration in the context of retail sales.

DATASET STRUCTURE

Tables in Dataset:

- **Order Details:** Contains information on items within each order (pizza type, quantity, price).
- **Orders:** Order-specific details such as order ID, date, and potentially total amount.
- **Pizza Types:** Defines different pizza categories and descriptions.
- **Pizzas:** Lists pizzas, linking each to type and size.

BASIC . LEVEL . ANALYSIS

1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```sql
-- Retrieve the total number of orders placed.
select count(order_id) as total_orders from orders;
```

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```sql
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;
```

BASIC . LEVEL . ANALYSIS

3. 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;
```

4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```sql
SELECT
    quantity, COUNT(order_details_id)
FROM
    order_details
GROUP BY quantity;

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

BASIC . LEVEL . ANALYSIS

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

```sql
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;
```

INTERMEDIATE . LEVEL . ANALYSIS

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

```sql
SELECT
    pizza_types.category,
    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.category
ORDER BY quantity DESC;
```

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);
```

INTERMEDIATE . LEVEL . ANALYSIS

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

```sql
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS averagr_quantity
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;
```

INTERMEDIATE . LEVEL . ANALYSIS

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

```
    SELECT
     pizza_types.name,
     SUM(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;
```

ADVANCE . LEVEL . ANALYSIS

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

```sql
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;
```

2. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select order_date,
sum(revenue) over(order by 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;
```

BASIC . LEVEL . ANALYSIS

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

```
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((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 rn<=3;
```

SCHEMAS

```
1 •  SELECT * FROM pizzahut.pizza_types;
```

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

pizza_type_id	name	category	ingredients
bbq_ckn	The Barbecue Chicken Pizza	Chicken	Barbecued Chicken, Red Peppers, G
cali_ckn	The California Chicken Pizza	Chicken	Chicken, Artichoke, Spinach, Garlic,
ckn_alfredo	The Chicken Alfredo Pizza	Chicken	Chicken, Red Onions, Red Peppers,
ckn_pesto	The Chicken Pesto Pizza	Chicken	Chicken, Tomatoes, Red Peppers, S
southw_ckn	The Southwest Chicken Pizza	Chicken	Chicken, Tomatoes, Red Peppers, R
thai_ckn	The Thai Chicken Pizza	Chicken	Chicken, Pineapple, Tomatoes, Red
big_meat	The Big Meat Pizza	Classic	Bacon, Pepperoni, Italian Sausage,
classic_dlx	The Classic Deluxe Pizza	Classic	Pepperoni, Mushrooms, Red Onions,

```
1 •  SELECT * FROM pizzahut.pizzas;
```

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

pizza_id	pizza_type_id	size	price
bbq_ckn_s	bbq_ckn	S	12.75
bbq_ckn_m	bbq_ckn	M	16.75
bbq_ckn_l	bbq_ckn	L	20.75
cali_ckn_s	cali_ckn	S	12.75
cali_ckn_m	cali_ckn	M	16.75
cali_ckn_l	cali_ckn	L	20.75

```
1 •  SELECT * FROM pizzahut.order_details;
```

Result Grid | Filter Rows: | Edit: | Export/I

order_details_id	order_id	pizza_id	quantity
1	1	hawaiian_m	1
2	2	classic_dlx_m	1
3	2	five_cheese_l	1
4	2	ital_supr_l	1
5	2	mexicana_m	1
6	2	thai_ckn_l	1
7	3	ital_supr_m	1
8	3	prsc_argla_l	1

Query 1 | SQL File 10* | SQL File 2* | SQL File 4* | SQL File 9*

Limit to 1000 rows

```
1 •  SELECT * FROM pizzahut.orders;
```

Result Grid | Filter Rows: | Edit:

order_id	order_date	order_time
1	2015-01-01	11:38:36
2	2015-01-01	11:57:40
3	2015-01-01	12:12:28
4	2015-01-01	12:16:31

THANK YOU