PIZZAHUT SALES ANALYSIS

A SQL PROJECT

Hey there! I'm Sukanya Das, and I'm excited to share my project on PizzaHut Sales Analysis, where I used MySQL to explore sales data. The goal of this project was to analyze PizzaHut's sales trends, uncover top-performing pizzas, and understand the revenue flow across different categories. Using advanced SQL queries, I examined the sales data in-depth, ranking pizzas based on their popularity and revenue. This project helped me identify valuable insights into customer preferences and overall sales performance. Let's dive into the data and see what it reveals!

For this project, I sourced pizza sales data from Kaggle which contains four csv files: orders_details.csv, orders.csv, pizza_types.csv, and pizzas.csv, which I subsequently imported into MySQL Workbench.

- **orders.csv has columns :** *order_id, date, time*
- **orders_details.csv has columns :** *order_details_id, order_id, pizza_id, quantity*
- **pizza_types.csv has columns :** *pizza_type_id, name, category, ingredients*
- **pizzas.csv has columns :** *pizza_id, pizza_type_id, size, price*

RETRIVE THE TOTAL NUMBER OF ORDERS PLACED

QUERY

```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

OUTPUT

Result Grid

	total_orders
▶	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

QUERY

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

OUTPUT

Result Grid

	total_sales
▶	817860.05

IDENTIFY THE HIGHEST-PRICED PIZZA

QUERY

```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;
```

OUTPUT

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

QUERY

```
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;
```

OUTPUT

Result Grid

size	order_count
L	18526

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

QUERY

```
SELECT
    pizza_types.name, SUM(orders_details.quantity) 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;
```

OUTPUT

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

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

QUERY

```
SELECT
    pizza_types.category,
    SUM(orders_details.quantity) 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.category
ORDER BY quantity DESC;
```

OUTPUT

Result Grid

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

QUERY

```sql
SELECT
    HOUR(order_time), COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```

OUTPUT

Result Grid | Filter Rows:

HOUR(order_time)	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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

QUERY

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

OUTPUT

Result Grid | Filter Rows

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

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

QUERY

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

OUTPUT

Result Grid | Filter Rows:

	avg_pizza_orders_perday
▶	138

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE

QUERY

```sql
SELECT
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

OUTPUT

Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE

QUERY

```
SELECT
    pizza_types.category,
    ROUND(SUM(orders_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(orders_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    orders_details
                        JOIN
                    pizzas ON pizzas.pizza_id = orders_details.pizza_id) * 100,
            2) 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
ORDER BY revenue DESC;
```

OUTPUT

Result Grid

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

QUERY

```sql
SELECT
    order_date,
    SUM(revenue) OVER (ORDER BY order_date) AS cum_revenue
FROM (
    SELECT
        orders.order_date,
        SUM(orders_details.quantity * 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;
```

OUTPUT

Result Grid | Filter Rows:

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	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	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001

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

QUERY

```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.quantity * 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;
```

OUTPUT

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
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5

THANK YOU!