SQL PROJECT ON 🍕 PIZZA SALES

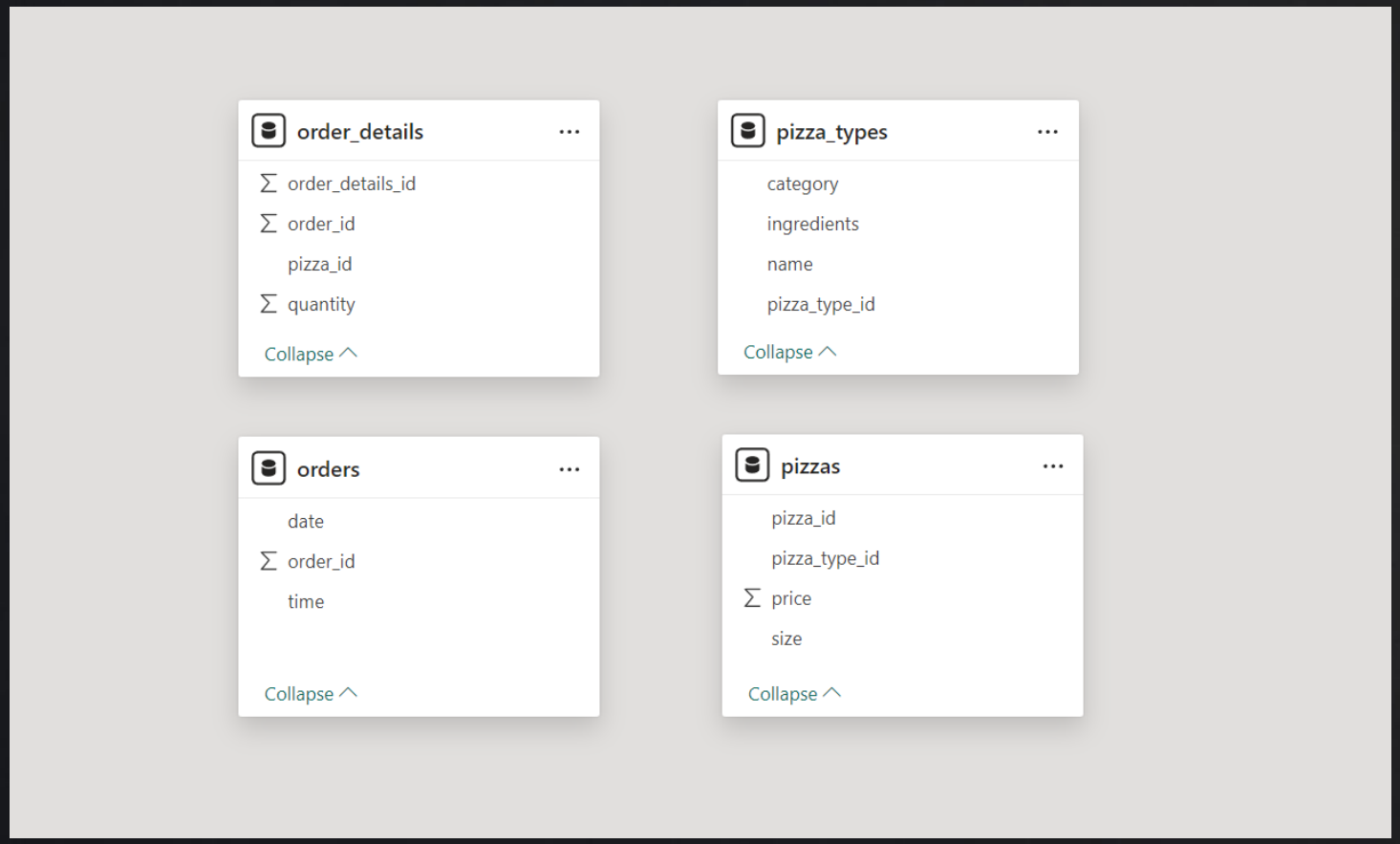
Analyzing pizza sales for top brands to uncover key insights.

INTRODUCTION

Welcome to the Pizza Sales SQL Project by Farhan Raza! This project analyzes Pizza Papa John's sales data using SQL queries to uncover insights and optimize inventory.

Thank you for exploring this project!

Below is a list of all the tables used in the SQL queries.



RETRIEVE THE TOTAL NUMBER OF PRDERS PLACED

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

Query

	total_orders
▸	21350

Output

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

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

Query

	total_revenues
▸	817860.05

Output

IDENTIFY THE HIGHEST-PRICED PIZZA, FOLLOW THESE STEPS:

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

Query

	name	price
▸	The Greek Pizza	35.95

Output

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
SELECT
    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 size
ORDER BY order_count DESC
```

Query

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

Output

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

```sql
SELECT
    pizza_types.name, SUM(order_details.quantity) AS qty
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY qty DESC
LIMIT 5;
```

Query

name	qty
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Output

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

```sql
SELECT
    category, SUM(quantity) AS qty
FROM
    pizza_types pt
        JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    order_details od ON od.pizza_id = p.pizza_id
GROUP BY category
ORDER BY qty DESC
```

Query

category	qty
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Output

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 o
GROUP BY hour
```

Query

Output

hour	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.

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

Query

category	pizza_types_count
Chicken	6
Classic	8
Supreme	9
Veggie	9

Output

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

```sql
SELECT
    ROUND(AVG(qty)) as average_per_day
FROM
    (SELECT
        (o.order_date) AS date, SUM(quantity) AS qty
    FROM
        orders o
    JOIN order_details od ON od.order_id = o.order_id
    GROUP BY date) AS order_quantity;
```

Query

average_per_day
138

Output

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

```sql
SELECT
    (pizza_types.name) AS 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_name
ORDER BY revenue DESC
LIMIT 3;
```

Query

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

Output

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL

```sql
WITH cte1 AS (
    SELECT
        pizza_types.category AS pizza_category,
        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_category
    ORDER BY revenue DESC
)
SELECT
    pizza_category,
    revenue,
    round((revenue / (SELECT SUM(revenue) FROM cte1)) * 100,2) AS percentage_contribution
FROM
    cte1;
```

Query

pizza_category	revenue	percentage_contribution
Classic	220053.1000000001	26.91
Supreme	208196.99999999822	25.46
Chicken	195919.5	23.96
Veggie	193690.45000000298	23.68

Output

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
WITH cte AS (
    SELECT
        pizza_types.category AS pizza_category,
        SUM(pizzas.price * order_details.quantity) AS revenue,
        orders.order_date
    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
    JOIN
        orders ON orders.order_id = order_details.order_id
    GROUP BY pizza_category, orders.order_date
)
SELECT
    pizza_category,
    order_date,
    ROUND(SUM(revenue) OVER (ORDER BY order_date asc)) AS cumulative_revenue
FROM
    cte
ORDER BY pizza_category, order_date;
```

Query

Output

pizza_category	order_date	cumulative_revenue
Chicken	2015-01-01	2714
Chicken	2015-01-02	5446
Chicken	2015-01-03	8108
Chicken	2015-01-04	9864
Chicken	2015-01-05	11930
Chicken	2015-01-06	14358
Chicken	2015-01-07	16561
Chicken	2015-01-08	19399
Chicken	2015-01-09	21526
Chicken	2015-01-10	23990

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

```sql
WITH cte AS (
    SELECT
        pizza_types.category AS pizza_category,
        pizza_types.name AS pizza_name,
        SUM(pizzas.price * order_details.quantity) AS revenue,
        RANK() OVER (PARTITION BY pizza_types.category ORDER BY SUM(pizzas.price * order_details.quantity) DESC) AS revenue_rank
    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
    JOIN
        orders ON orders.order_id = order_details.order_id
    GROUP BY pizza_category, pizza_name
)
SELECT
    pizza_category,
    pizza_name,
    revenue
FROM
    cte
WHERE
    revenue_rank <= 3
ORDER BY
    pizza_category, revenue_rank;
```

Query

Output

pizza_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

THANK YOU

This SQL report provides key insights into the pizza company’s sales and operations

.

Feel free to ask any questions! 👋

05-09-2024