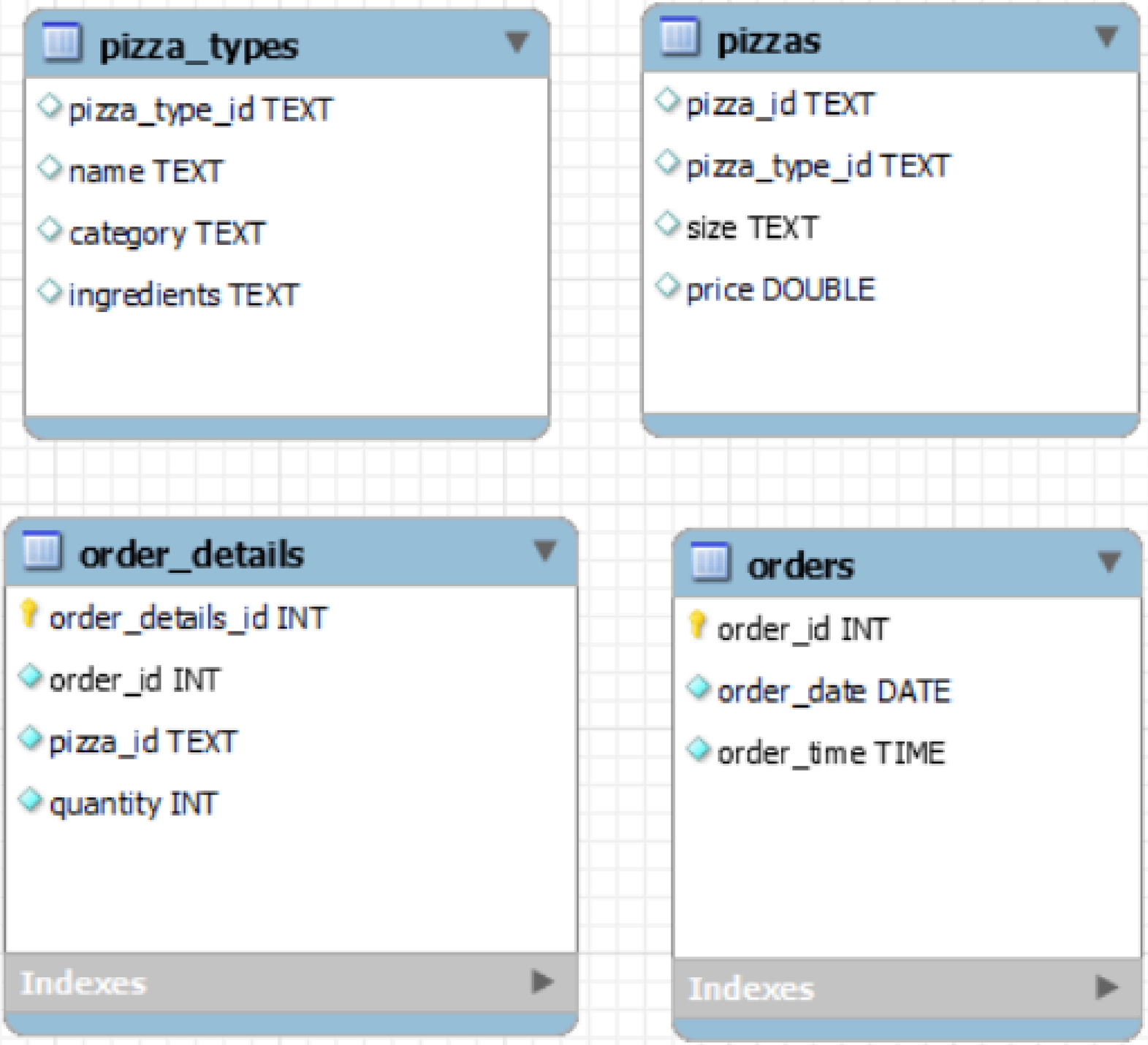
Pizzahut sales using sql

26 May, 2024

Access to organic, healthy food

Hello! My name is Sumit Prasad. In this project, I will utilize SQL queries to solve questions related to pizza. This will involve working with a database that likely contains information about pizza orders, ingredients, customers, and other related data. By writing and executing SQL queries, I aim to extract meaningful insights and answer specific questions about the pizza data. Let's dive into the specifics of the project and explore the queries needed to achieve our objectives!


pizza_types
pizza_type_id TEXT
name TEXT
category TEXT
ingredients TEXT
pizzas
pizza_id TEXT
pizza_type_id TEXT
size TEXT
price DOUBLE
order_details
order_details_id INT
order_id INT
pizza_id TEXT
quantity INT
Indexes
orders
order_id INT
order_date DATE
order_time TIME
Indexes

01

Retrieve the total number of orders placed

02

Calculate the total revenue generated from pizza sales.

03

Identify the highest-priced pizza.

04

Identify the most common pizza size ordered.

05

List the top 5 most ordered pizza types along with their quantities

06

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

07

Determine the distribution of orders by hour of the day.

08

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

09

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

10

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

11

Calculate the percentage contribution of each pizza type to total revenue.

12

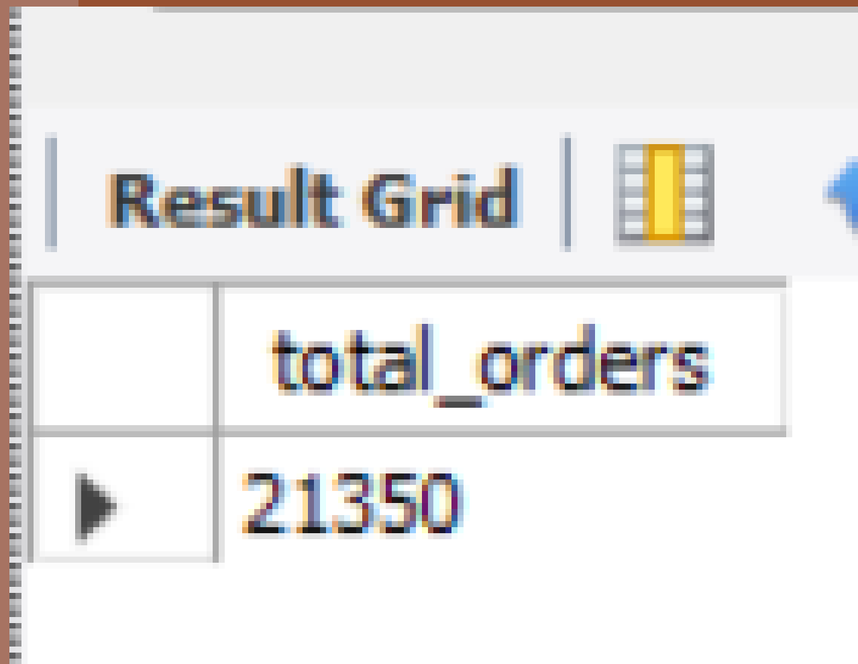
Analyze the cumulative revenue generated over time.

13

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

RETRIVE THE TOTAL NUMBER OF ORDER PLACES

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



CALCULATE THE TOTAL RENVENUE 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;
```

Result Grid

	total_sales
▶	817860.05

IDENTIFY 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 Rows: | Export: | Wrap

name	price
The Greek Pizza	35.95

IDENTITY MOST COMMON PIZZA SIZE ORDERED

```sql
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;
```

Result Grid | Filter R

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

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

```sql
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantitiy
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 quantitiy DESC
LIMIT 5;
```

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

name	quantitiy
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

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

Result Grid | Filter Rows

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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 | Filter R

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.

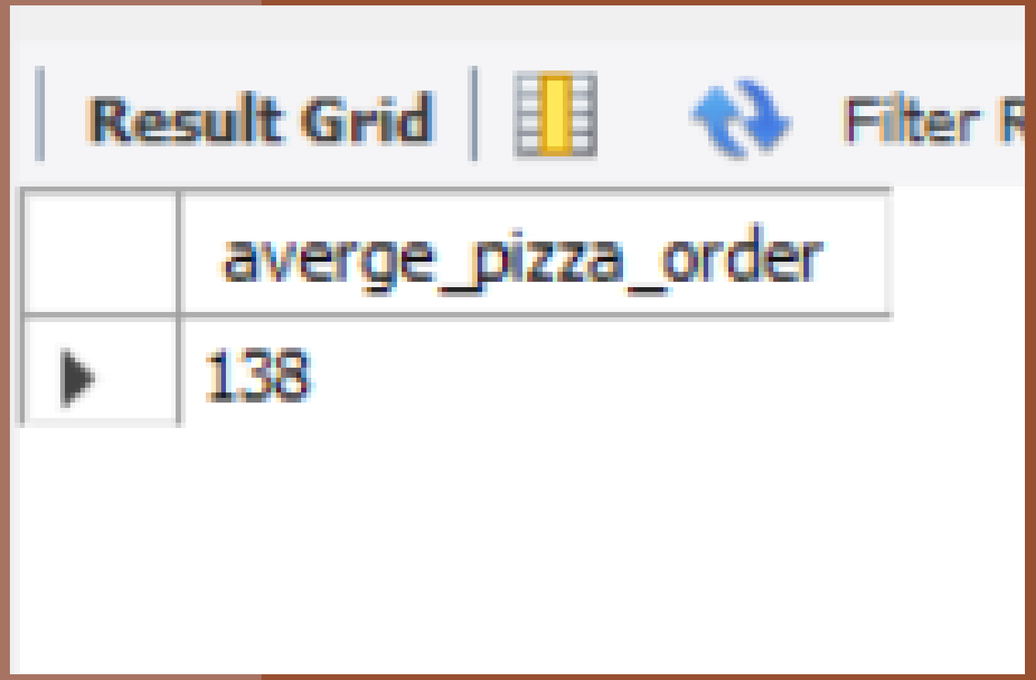
```sql
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Rows:

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

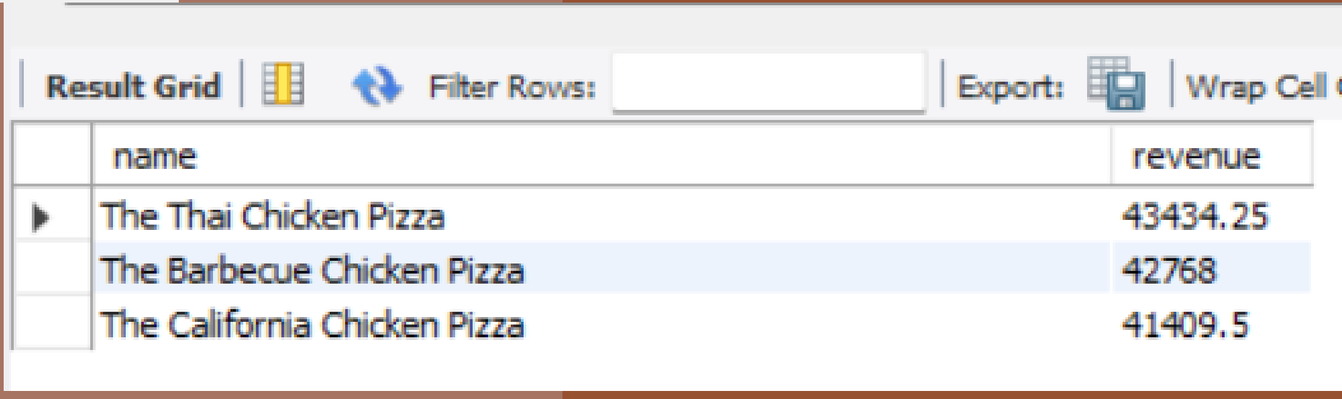
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

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



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

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizzas
        JOIN
    pizza_types 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;
```



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.

```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
    pizzas
        JOIN
    pizza_types 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 revenue DESC;
```

Result Grid | Filter R

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select order_date,
sum(renvenue) over(order by order_date) as cumm_revenue
 from
 (select orders.order_date,sum(order_details.quantity*pizzas.price) as renvenue
 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;
```

Result Grid | Filter Rows:

order_date	cumm_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
2015-01-17	39001.75000000001
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001

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((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;
```

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

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
LIKE AND COMMENT

https://github.com/Sumit123sm

sumitprasad102@gmail.com