PizzaHut Sales

Analysis on pizza sales

Hello,

“ My name is Mahaveer. In this project, I have utilized the question that where related to pizza sales”

Created pizzahut database

Create database pizzahut

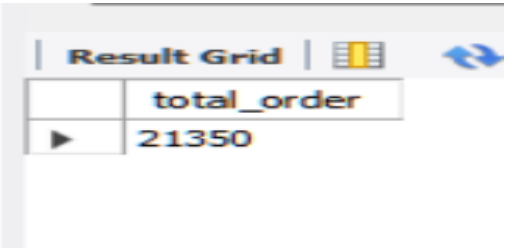
Then created tables – pizza, pizza_type
By importing data

Created table orders and order_details than imported data

```
1
2  create table orders (
3  order_id int not null,
4  order_date date not null,
5  order_time time not null,
6  primary key (order_id) );
7
8
9  create table order_details (
10 order_details_id int not null,
11 order_id int not null,
12 pizza_id text not null,
13 quantity int not null,
14 primary key (order_details_id));
```

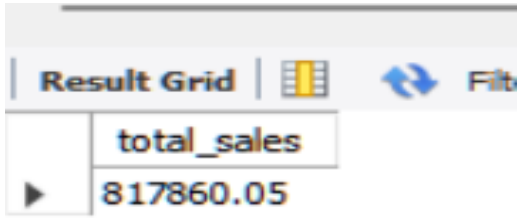
1.Retrieve the total number of orders placed.

- select count(order_id) as total_order from orders;



2. Calculate the total revenue generated from pizza sales.

```
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
```



3.Identify the highest-priced pizza

```sql
-- Identify the highest-priced pizza.

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 Cell Content: | Fetch rows:

name	price
The Greek Pizza	35.95

4.Identify the most common pizza size ordered.

```sql
-- Identify the most common pizza size ordered.

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 Rows: | Export: | Wrap Cell Content:

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

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;
```

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

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

```
3 •  SELECT
4        pizza_types.category,
5        SUM(order_details.quantity) AS quantity
6    FROM
7        pizza_types
8            JOIN
9        pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
10           JOIN
11       order_details ON order_details.pizza_id = pizzas.pizza_id
12   GROUP BY pizza_types.category
13   ORDER BY quantity DESC;
```

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

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7.Determine the distribution of orders by hour of the day.

```
3 •  SELECT
4        HOUR(order_time) AS hour, COUNT(order_id) AS order_count
5    FROM
6        orders
7    GROUP BY HOUR(order_time);
8
```

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

```
1   -- Join relevant tables to find the category-wise distribution of pizzas.
2
3 • select category , count(name) from pizza_types
4   group by category
```

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

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

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

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

SELECT
    ROUND(AVG(quantity), 0) as average_pizza_ordered_per_day
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;

```

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

average_pizza_ordered_per_day
138

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

```sql
-- 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;
```

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

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

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

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

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;
```

12.Analyze the cumulative revenue generated over time.

```sql
-- Analyze the cumulative revenue generated over time.

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;
```

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

```sql
-- 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;
```