SQL PROJECT ON

PIZZA

SALES

FREE DELIVERY

Contact Us:

HELLO

I'm subhash In this project,
I utilized SQL queries to solve the questions that where related to piza sales

CONTACT

9481612592

https://www.linkedin.com/in/subhash-babu-vn-235b6b236

Retrieve the total number of orders placed.

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13*

Limit to 1000 rows

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

Calculate the total revenue generated from pizza sales.

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13*

Limit to 1000 rows

```sql
-- 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
```

Identify the highest-priced pizza.

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13*

Limit to 1000 rows

```sql
-- 3.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;
```

Identify the most common pizza size ordered.

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13*

Limit to 1000 rows

```sql
-- 4.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
LIMIT 1;
```

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

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13* | SQL File 14*

Limit to 1000 rows

```sql
-- 5.List the top 5 most ordered pizza types along with their quantities.
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;
```

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

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13* | SQL File 14* | SQL File 15* | SQL File 16* | SQL File 17* | S

Limit to 1000 rows

```sql
-- Intermediate:
-- 6.Join the necessary tables to find the total quantity of each pizza category ordered.
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 pizzas.pizza_id=order_details.pizza_id
group by pizza_types.category order by quantity;
```

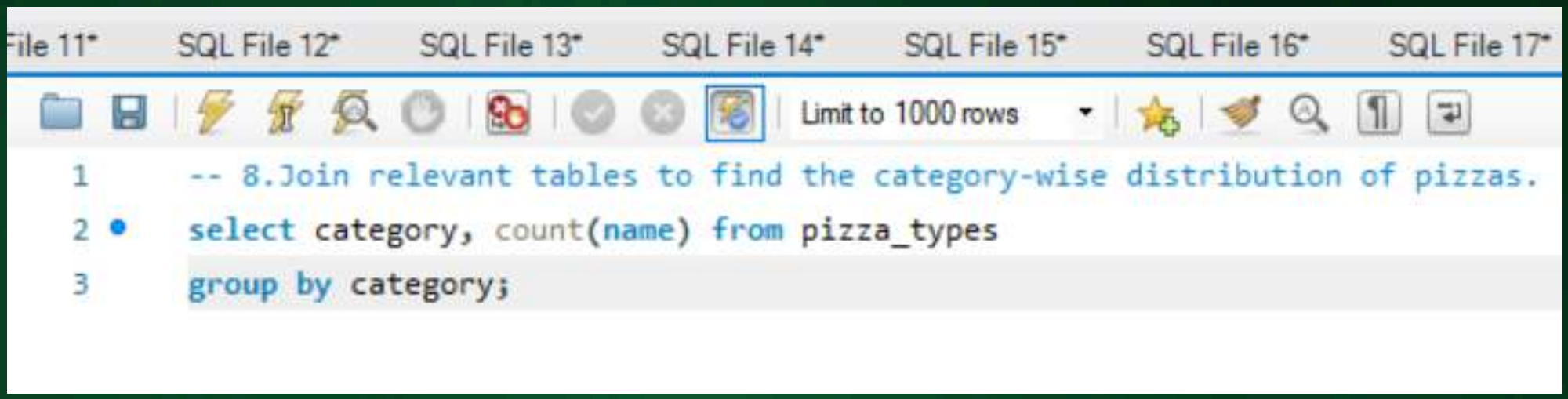
Determine the distribution of orders by hour of the day.

etails | pizza_types | pizzas | SQL File 11* | SQL File 12* | SQL File 13* | SQL File 14*

Limit to 1000 rows

```sql
-- 7.Determine the distribution of orders by hour of the day.
select hour(order_time) as hour ,count(order_id) as order_count from orders
group by hour ;
```

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




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

File 11* | SQL File 12* | SQL File 13* | SQL File 14* | SQL File 15* | SQL File 16* | SQL File 17* | SQL File 18*

Limit to 1000 rows

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

SELECT
    ROUND(AVG(order_per_day), 0) as av_pizza_order_perday
FROM
    (SELECT
        orders.order_date AS date,
            SUM(order_details.quantity) AS order_per_day
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY date) AS quantity;
```

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

SQL File 12* | SQL File 13* | SQL File 14* | SQL File 15* | SQL File 16* | SQL File 17* | SQL File 18* | SQL File 19* | SQL File 20*

Limit to 1000 rows

```sql
-- 10.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 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 revenue desc limit 3;
```

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

SQL File 15* | SQL File 16* | SQL File 17* | SQL File 18* | SQL File 19* | SQL File 20* | SQL File 21* | SQL File 22*

Limit to 1000 rows

```sql
-- Advanced:
-- 11.Calculate the percentage contribution of each pizza type to total revenue.
select pizza_types.category as pizza,
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 order by revenue desc;
```

Analyze the cumulative revenue generated over time.

SQL File 15* | SQL File 16* | SQL File 17* | SQL File 18* | SQL File 19* | SQL File 20* | SQL File 21* | SQL File 22* | SQL File 23* | SQL

Limit to 1000 rows

```sql
-- 12.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  ;
```

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

SQL File 15* SQL File 16* SQL File 17* SQL File 18* SQL File 19* SQL File 20* SQL File 21* SQL File 22*

Limit to 1000 rows

```sql
-- 13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.
select name,category,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;
```