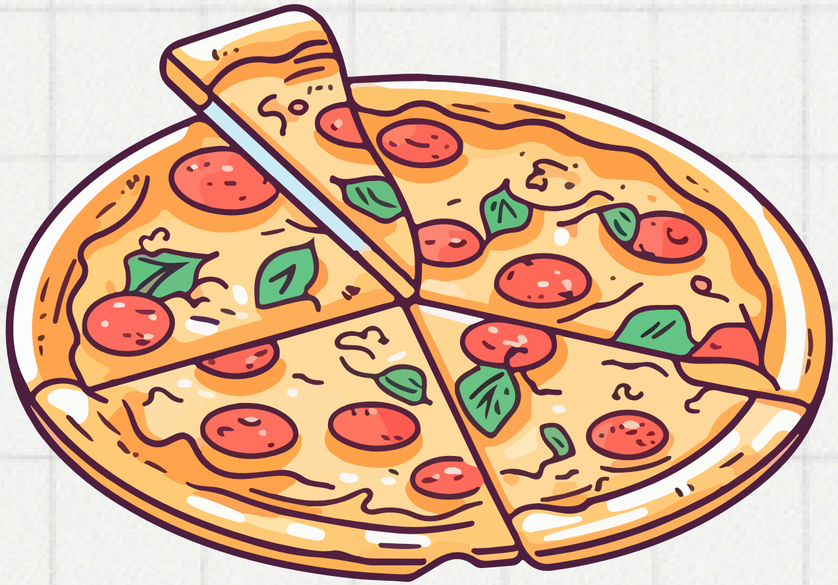
SQL PROJECT

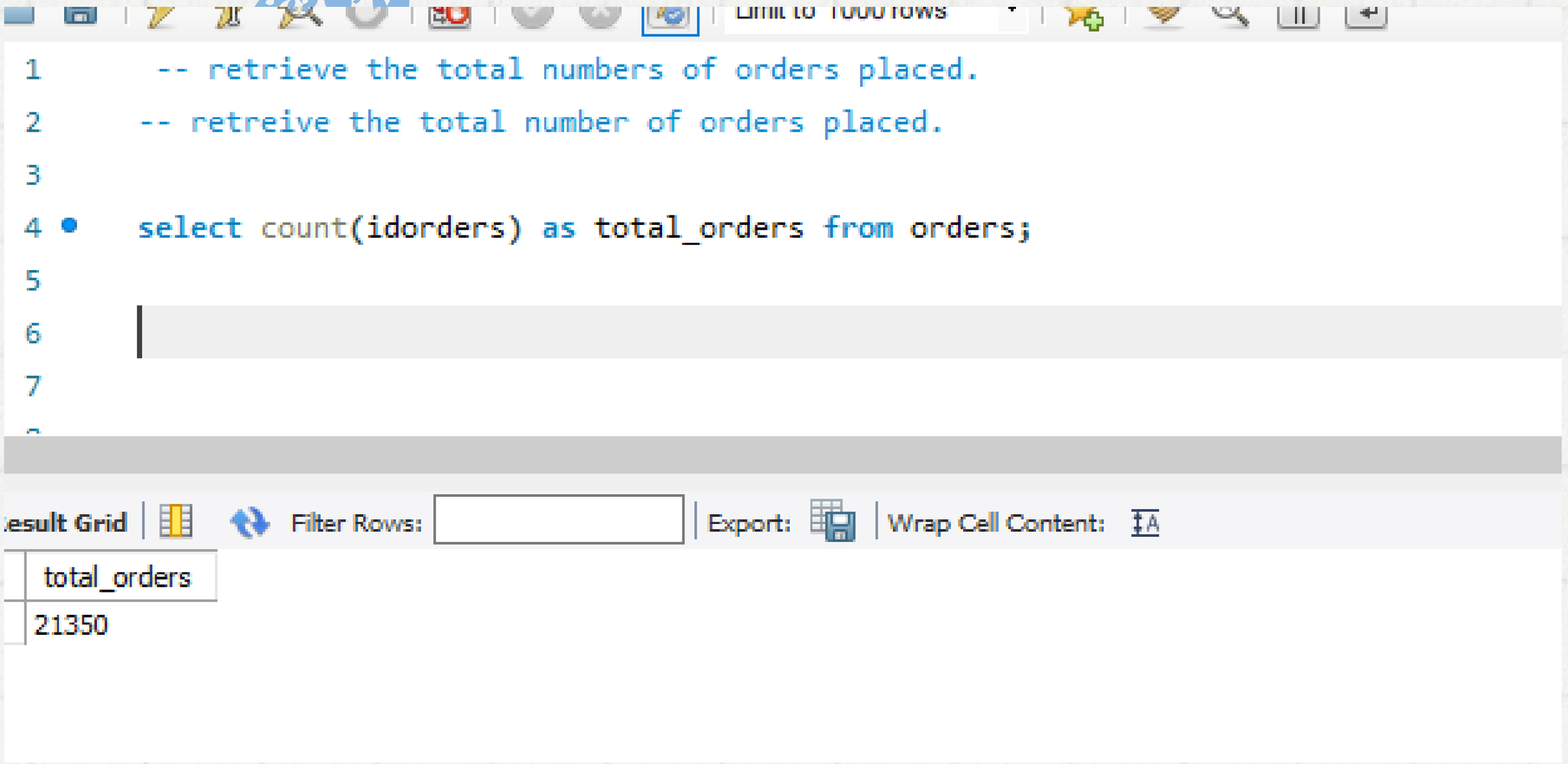
Presented by PRAGYA SINGH

PIZZA SALES



This Pizza Sales Project uses SQL to analyze sales data for a pizza restaurant. It tracks customer orders, inventory, and sales trends in a database. The project focuses on finding out which menu items are popular, managing stock levels, and improving marketing efforts. By looking at the data, the restaurant can make better decisions to increase profits .

Retrieve the total number of orders placed.



Calculate the total revenue generated from pizza sales.

```sql
-- 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;
```

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

Total_sales
817860.05

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

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

```sql
-- identify the most common pizza size ordered.
SELECT
    pizzas.size,
    COUNT(order_details.idorder_details) 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:

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

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

```sql
-- list the tp 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;
```

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.

```sql
-- 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 order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

```sql
-- determine the distribution of orders by hour of the day.

SELECT
    HOUR(orderscol1) AS hour, COUNT(idorders) AS order_count
FROM
    orders
GROUP BY HOUR(orderscol1);
```

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

our	order_count
	1231
	2520
	2455
	1472
	1468
	1920
	2336
	2399
	2009
	1642
	1198
	663
	28
	8

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

```sql
-- join relevant tables to find the
-- category-wise distribution of pizzas.
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

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

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

SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.orderscol, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.idorders = order_details.order_id
    GROUP BY orders.orderscol) AS order_quantity;
```

Result Grid | Filter Rows:

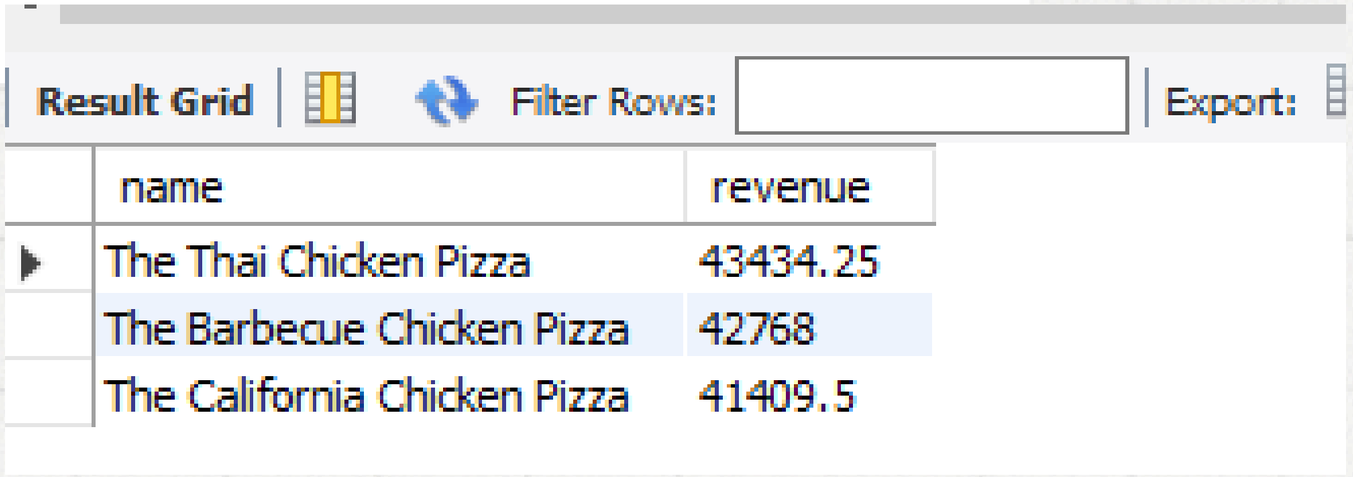
	ROUND(AVG(quantity), 0)
▶	138

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;
```



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
-- calculate the percentage contribution of each
-- pizza type to total revenue.

SELECT
    pizza_types.category,
    (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 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;
```

Result Grid | Filter Rows: | E

category	revenue
Classic	26.90596025566967
Supreme	25.45631126009862
Chicken	23.955137556847287
Veggie	23.68259092738457 7

Analyze the cumulative revenue generated over time.

```sql
-- analyse the comulative revenue generated over time.

select orderscol ,sum(revenue) over(order by orderscol) as cum_revenue
from
(select orders.orderscol,
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.idorders = order_details.order_id
group by orders.orderscol) as sales;
```

Result Grid | Filter Rows:

orderscol	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
2015-01-17	39001.75000000001

Result 1

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;
```

Result Grid | Filter Rows: | Ex

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 very much!