

# SQL PROJECT ON PIZZA SALES

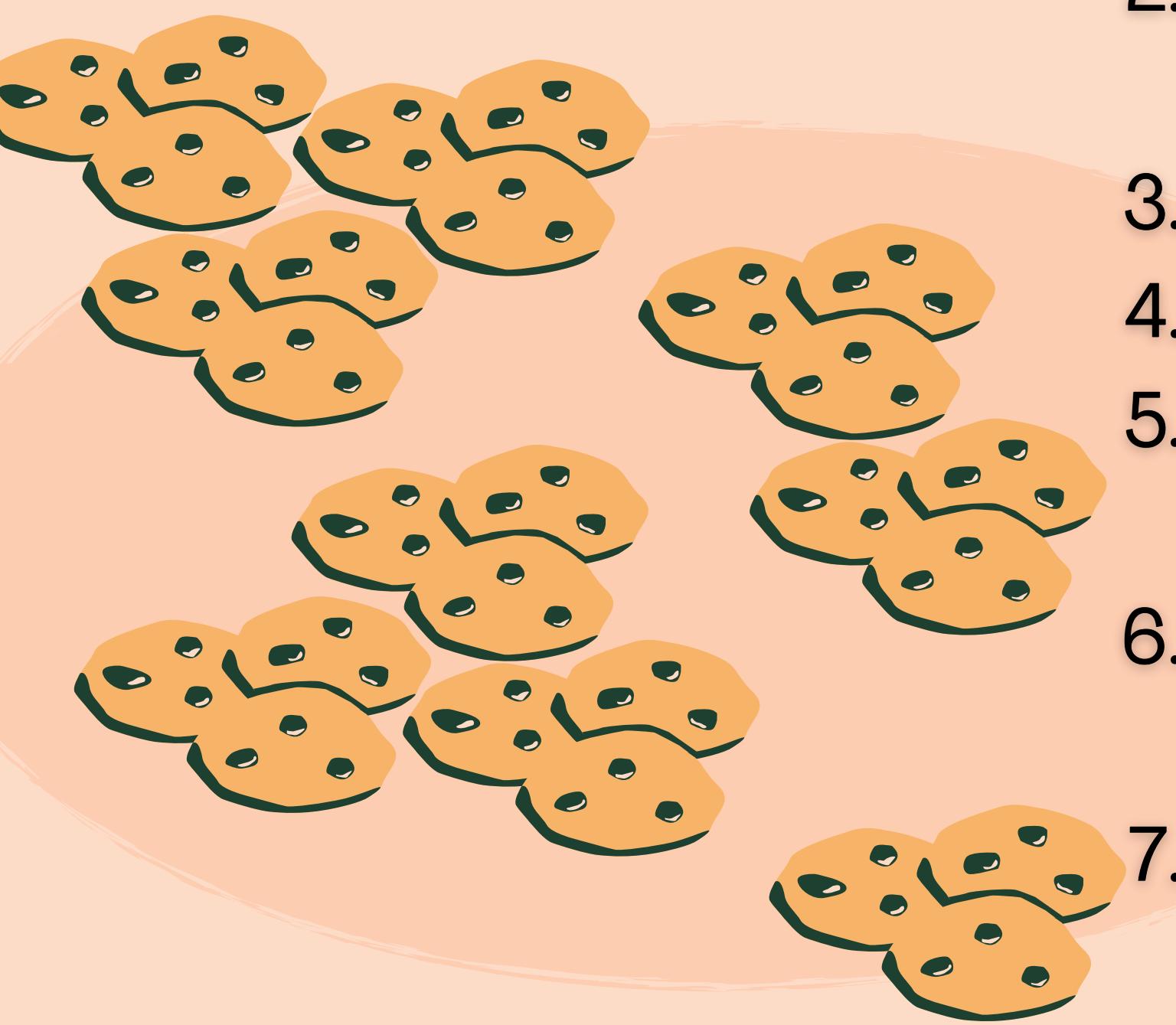
Analysis of Pizza Sales by using Sql



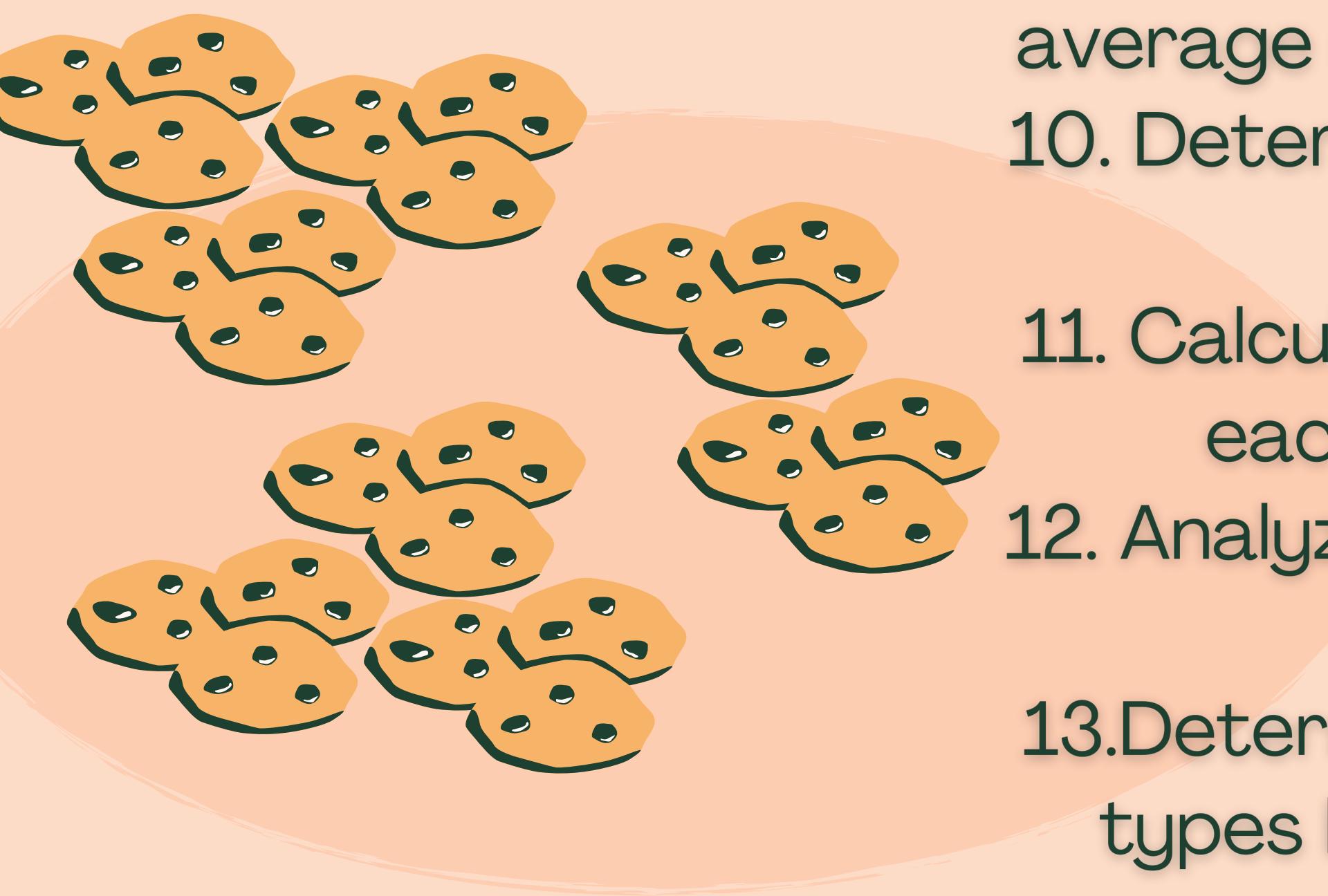
# **HELLO!**

**My name is Anshika Srivastava . In this project I have utilized my SQL queries to solve the question related to Pizza Sales Analysis.**

# QUESTIONS??

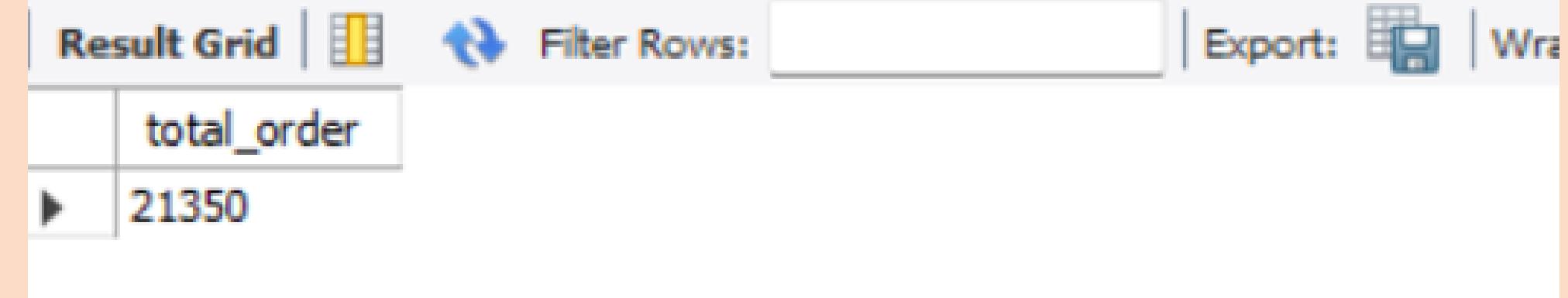
- 
1. Retrieve the total number of orders placed .
  2. Calculate the total revenue generated from pizza sales.
  3. Identify the highest-priced pizza.
  4. Identify the most common pizza size ordered.
  5. List the top 5 most ordered pizza types along with their quantities.
  6. Join the necessary tables to find the total quantity of each pizza category ordered.
  7. Determine the distribution of orders by hour of the day.

# QUESTIONS??

- 
8. Join relevant tables to find the category-wise distribution of pizzas.
  9. 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.

**RETRIEVE  
THE TOTAL  
NUMBER OF  
ORDERS  
PLACED.**

```
select count(order_id) as total_order from orders;
```



A screenshot of a MySQL query results window. The window has a light gray header bar with the text "Result Grid" and a grid icon, "Filter Rows:" with a search input field, and "Export:" with a grid icon and a "Wr" button. Below the header is a table with one row. The table has a single column labeled "total\_order" which contains the value "21350".

total_order
21350

# 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:	Print		
<table border="1"><thead><tr><th>total_sales</th></tr></thead><tbody><tr><td>817860.05</td></tr></tbody></table>				total_sales	817860.05	
total_sales						
817860.05						

# 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: |

	name	price
▶	The Greek Pizza	35.95

# 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 Co

	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.

```
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.

```
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 category  
ORDER BY quantity DESC;
```

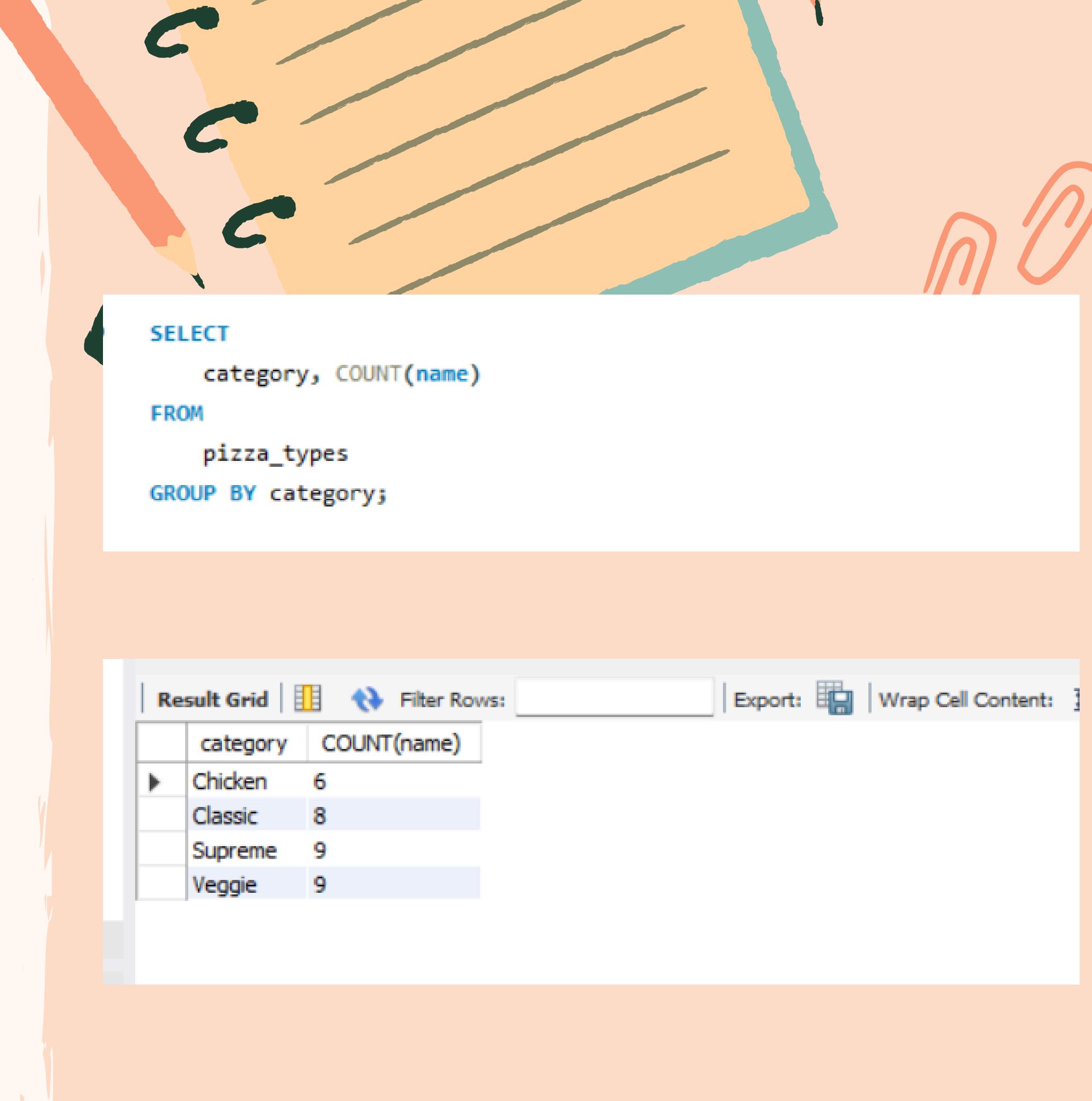
Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	category	quantity		
▶	Classic	14888		
	Supreme	11987		
	Veggie	11649		
	Chicken	11050		

# 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(order_time);
```

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.**



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

	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.

```
SELECT  
    ROUND(AVG(quantity), 2) AS pizzas_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;
```

pizzas_ordered_per_day
138.47

# 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 pizzas.pizza_id = order_details.pizza_id  
 GROUP BY pizza_types.name  
 ORDER BY revenue DESC  
 LIMIT 3;
```

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

	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.

```
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 order_details.pizza_id = pizzas.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 category
ORDER BY revenue DESC;
```

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

# 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;
```

order_date	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	23990.35000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.30000000003
2015-01-14	32358.70000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.60000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001
2015-01-23	52724.60000000006

# 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)
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 pizzas.pizza_id=order_details.pizza_id
group by pizza_types.category,pizza_types.name) as a) as b
where rn <=3;
```

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!!**