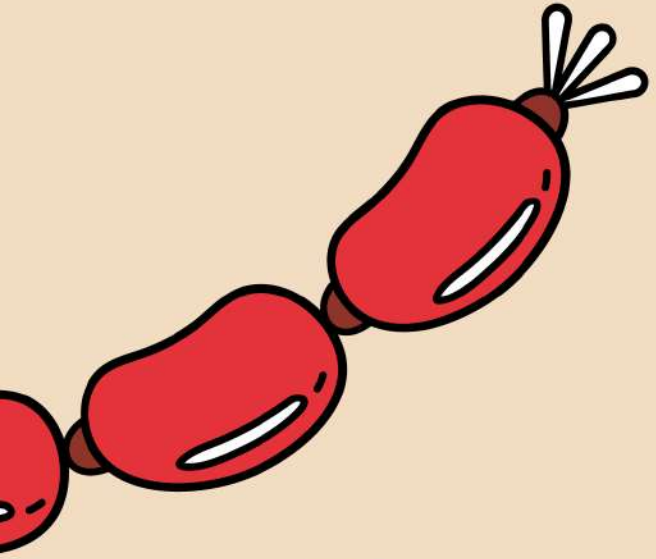


# PIZZAS SALES USING SQL





# Over view

**These queries help answer key business questions regarding:**

- **Sales performance (total orders, revenue, top pizzas)**
- **Customer preferences (most ordered pizza sizes, types)**
- **Operational insights (orders by time, pizza distribution by category)**





# Schemas

Table: **pizza\_types**

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

Table: **pizzas**

Columns:

pizza_id	text
pizza_type_id	text
size	text
price	double

Table: **orders**

Columns:

<u>order_id</u>	int PK
order_date	date
order_time	time

Table: **order\_details**

Columns:

<u>order_details_id</u>	int PK
order_id	int
pizza_id	text
quantity	int







# Retrieve the total number of orders placed.

```
-- Retrieve the total number of orders placed.  
SELECT  
    COUNT(order_id) AS total_order  
FROM  
    orders;
```

Result Grid	
	total_order
▶	21350



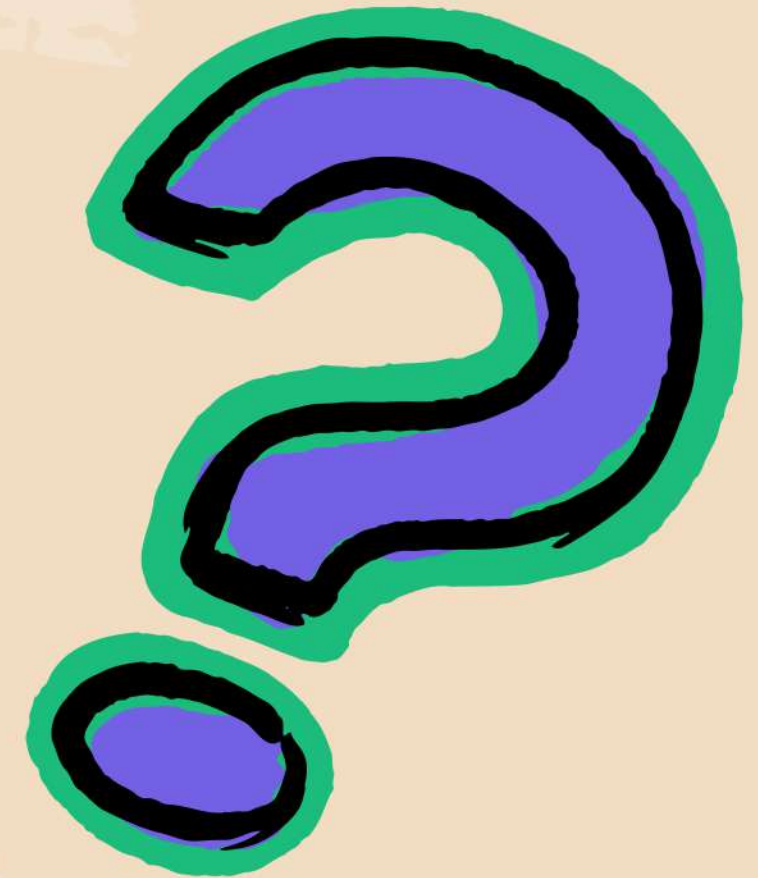


# Calculate the total revenue generated from pizza sales.



```
-- Calculate the total revenue generated from pizza sales.  
SELECT  
    ROUND(SUM(order_details.quantity * pizzas.price),  
          2) AS Total_revenue  
FROM  
    order_details  
    JOIN  
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

Result Grid	
	Total_revenue
▶	817860.05





# Identify the highest-priced pizza.

```
1  -- Identify the highest-priced pizza.
2
3  • SELECT
4      pizza_types.name, pizzas.price
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9  ORDER BY price DESC
10 LIMIT 1;
```



Result Grid			Filter Rows
	name	price	
▶	The Greek Pizza	35.95	





# Identify the most common pizza size ordered.

```
1  -- Identify the most common pizza size ordered
2
3  •  SELECT
4      pizzas.size,
5      COUNT(order_details.order_details_id) AS total_count
6  FROM
7      pizzas
8      JOIN
9      order_details ON order_details.pizza_id = pizzas.pizza_id
10 GROUP BY size
11 ORDER BY total_count DESC
12 LIMIT 1;
```

Result Grid			Filter
	size	total_count	
▶	L	18526	







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


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

Result Grid			Filter Rows:
	name	total_quantities	
▶	The Classic Deluxe Pizza	2416	
	The Barbecue Chicken Pizza	2372	
	The Hawaiian Pizza	2370	
	The Pepperoni Pizza	2369	
	The Thai Chicken Pizza	2315	





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



```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS total_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 category;
```

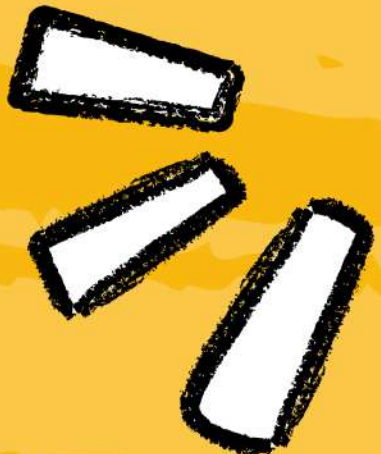
Result Grid			Filter Rows
	category	total_quantity	
▶	Classic	14888	
	Veggie	11649	
	Supreme	11987	
	Chicken	11050	



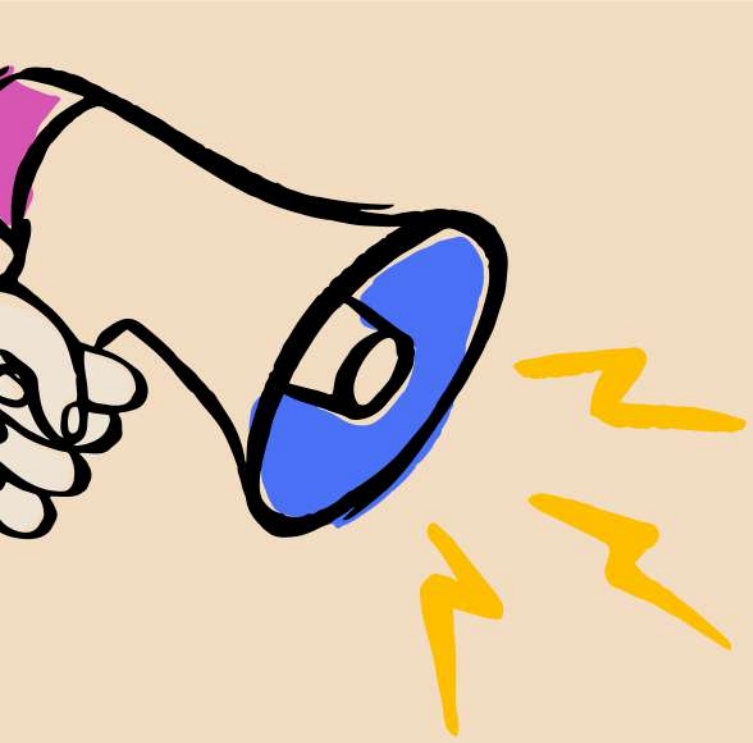
# Determine the distribution of orders by hour of the day.

```
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS orders
FROM
    orders
GROUP BY hour;
```

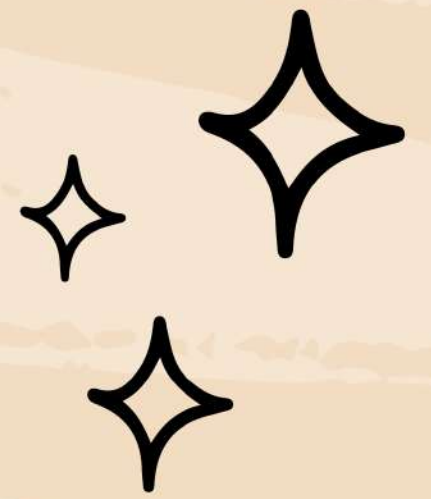
Result Grid		
	hour	orders
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009
	20	1642
	21	1198
	22	663
	23	28







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



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

Result Grid			Filter Rows
	category	distribution	
▶	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(total_sale)) AS average_sale_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS total_sale
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY order_date) AS order_quantity
```

Result Grid		Filter R
	average_sale_per_day	
▶	138	



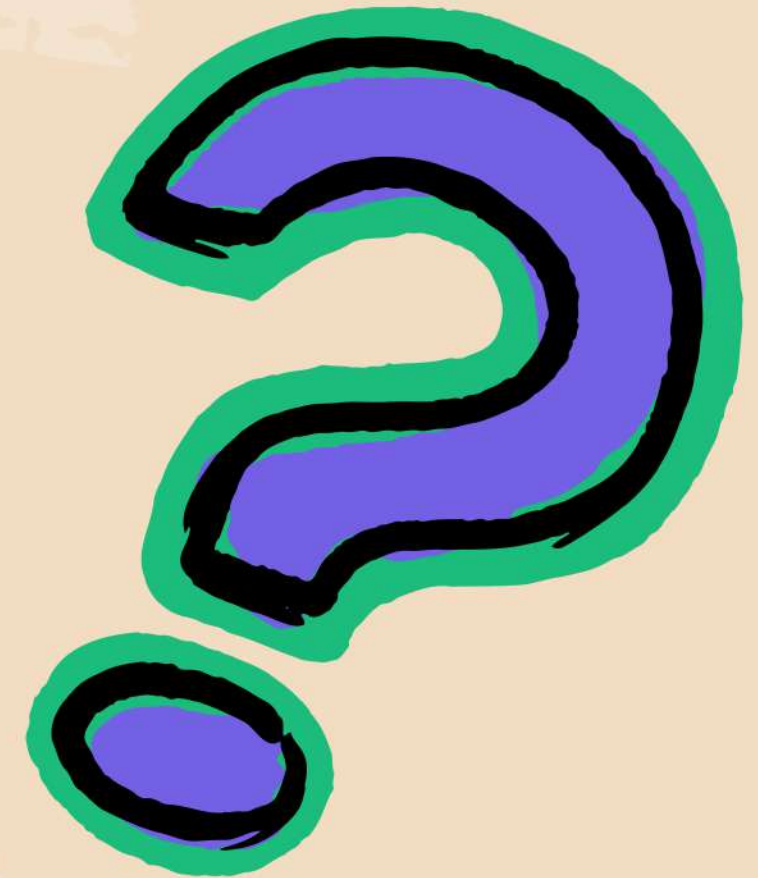


# Determine the top 3 most ordered pizza types based on revenue.



```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS total_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 name
ORDER BY total_Revenue DESC
LIMIT 3;
```

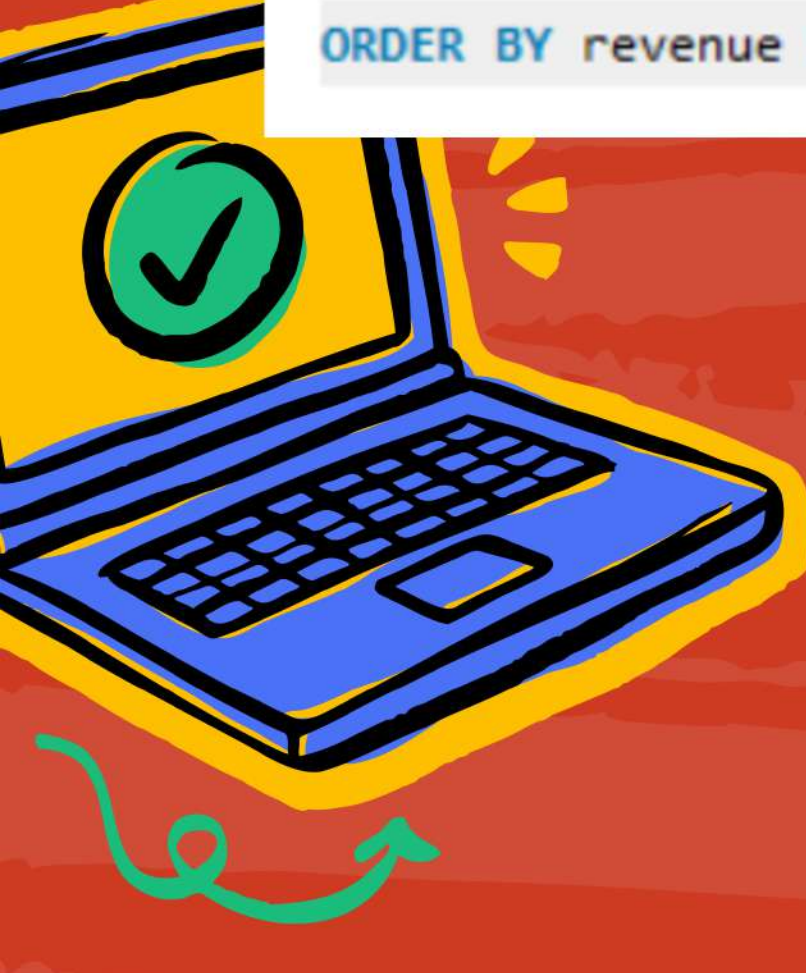
	name	total_Revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5





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)
    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) * 100,2) 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 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, round(revenue, 2) as revenue, round(sum(revenue) over(order by order_date), 2) as cum_revenue
from (select orders.order_date, sum(order_details.quantity*pizzas.price) as revenue
from orders join order_details
on orders.order_id = order_details.order_id join
pizzas on
pizzas.pizza_id = order_details.pizza_id
group by order_date) as sales
```



	order_date	revenue	cum_revenue
▶	2015-01-01	2713.85	2713.85
	2015-01-02	2731.9	5445.75
	2015-01-03	2662.4	8108.15
	2015-01-04	1755.45	9863.6
	2015-01-05	2065.95	11929.55
	2015-01-06	2428.95	14358.5
	2015-01-07	2202.2	16560.7
	2015-01-08	2838.35	19399.05
	2015-01-09	2127.35	21526.4
	2015-01-10	2463.95	23990.35
	2015-01-11	1872.3	25862.65
	2015-01-12	1919.05	27781.7
	2015-01-13	2049.6	29831.3
	2015-01-14	2527.4	32358.7
	2015-01-15	1984.8	34343.5




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

```
select category, name ,ranks, revenue from
(select category , name,revenue ,rank() over(partition by category order by revenue desc) as ranks 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 category, name) as sales) as ranking where ranks <=3;
```

Result Grid



Filter Rows:

Export:


Wrap Cells

	category	name	ranks	revenue
▶	Chicken	The Thai Chicken Pizza	1	43434.25
	Chicken	The Barbecue Chicken Pizza	2	42768
	Chicken	The California Chicken Pizza	3	41409.5
	Classic	The Classic Deluxe Pizza	1	38180.5
	Classic	The Hawaiian Pizza	2	32273.25
	Classic	The Pepperoni Pizza	3	30161.75
	Supreme	The Spicy Italian Pizza	1	34831.25
	Supreme	The Italian Supreme Pizza	2	33476.75
	Supreme	The Sicilian Pizza	3	30940.5
	Veggie	The Four Cheese Pizza	1	32265.700000000065
	Veggie	The Mexicana Pizza	2	26780.75
	Veggie	The Five Cheese Pizza	3	26066.5





# Thank You!



**Email**

athuljohnson5817h@gmail.com



**github**

<https://github.com/Athul5817h>

