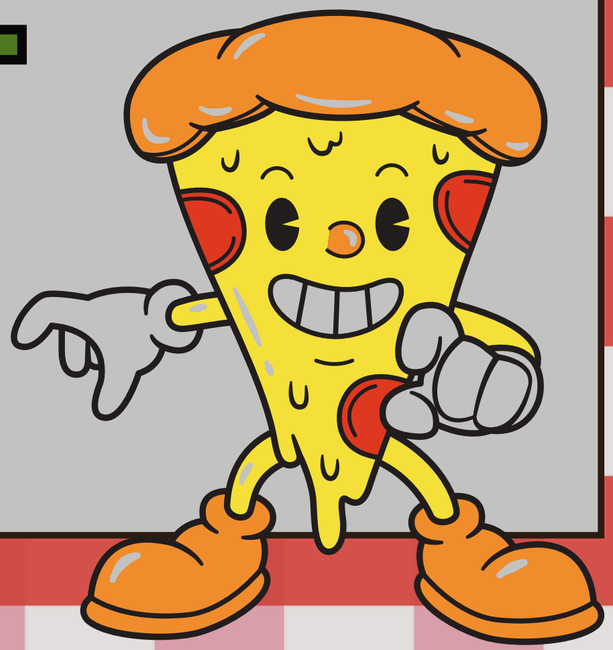
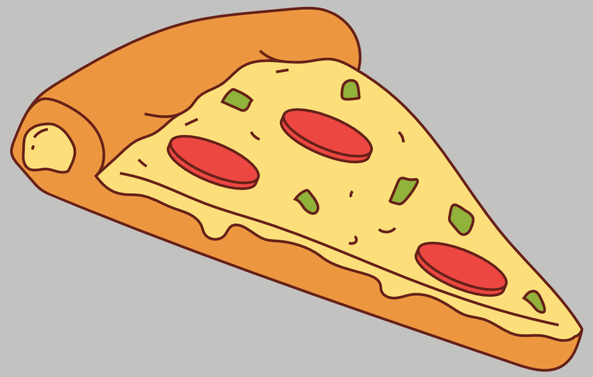
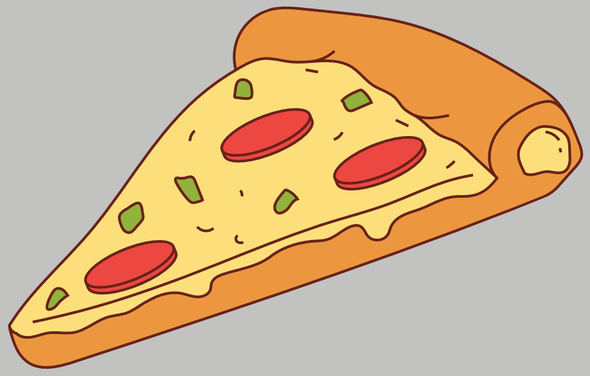
# SQL PROJECT

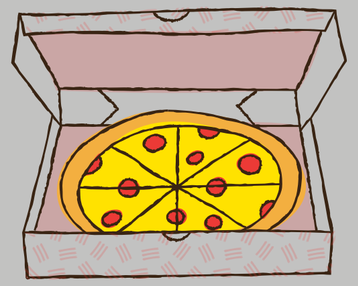




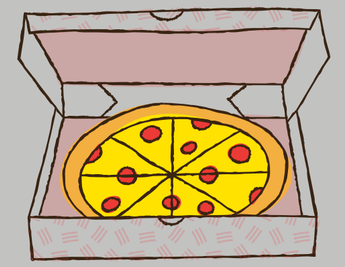
# PROJECT INTRODUCTION

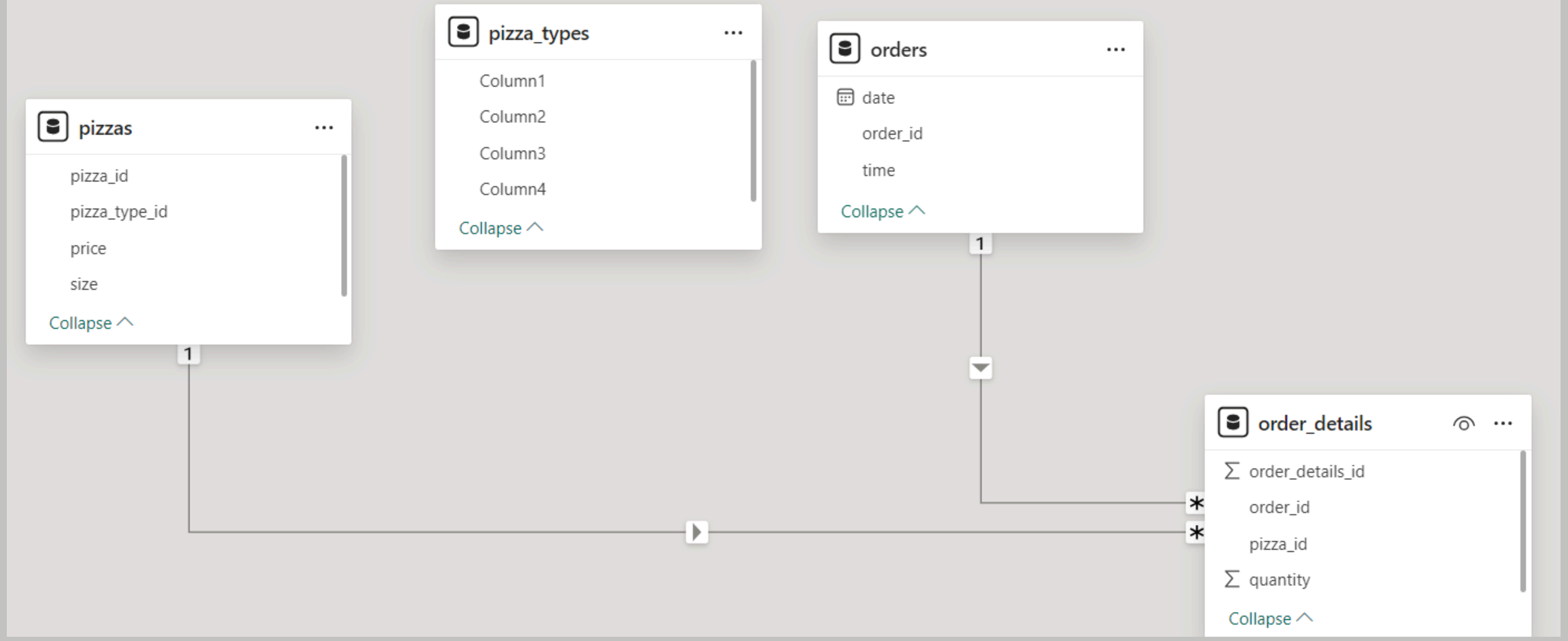


Hello, my name is Hemant Mishra, and in this project, I have utilized SQL queries to efficiently analyze and solve various data-related questions. My approach involved leveraging SQL's powerful querying capabilities to extract insights, perform data manipulation, and answer specific business questions from the dataset. This project demonstrates my proficiency in SQL and my ability to work with complex data to derive meaningful results.

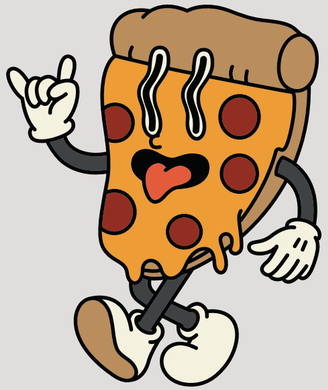


# DataSet




pizza_types
Column1
Column2
Column3
Column4
Collapse
orders
date
order_id
time
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
1
1
order_details
order_details_id
order_id
pizza_id
quantity
Collapse

# Questions

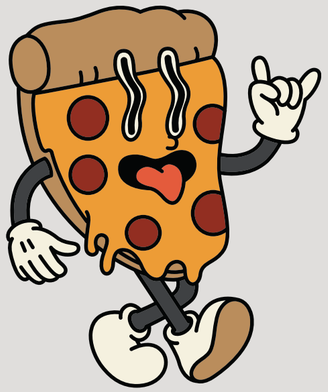


**Basic:**

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.

**Intermediate:**

- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.



**Advanced:**

- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

# Question 1:

MySQL Workbench

Local MySQL (pizza) - Warnin... | Local MySQL (pizza) - Warning - n...

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- credit_card
- **pizza**
- sys
- world

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8*

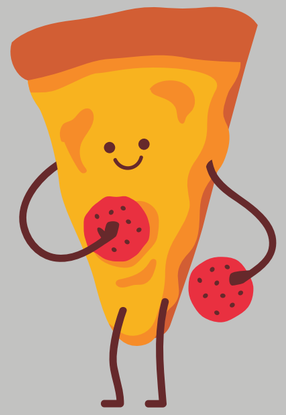
Limit to 1000 rows

```
1   # Question 1: Retrieve the total number of orders placed
2
3   SELECT
4       COUNT(order_id) AS total_orders
5   FROM
6       orders
7
```

Result Grid | Filter Rows: | Export: | Wrap

| total_orders |
| --- |
| 21350 |

# Answer



# QUESTION 2:

MySQL Workbench

Local MySQL (pizza) - Warnin... | Local MySQL (pizza) - Warning - n...

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- credit_card
- **pizza**
- sys
- world

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8*

Limit to 1000 rows

```
1  # Question 2: Calculate the total revenue generated from pizza sales.
2
3  SELECT
4      ROUND(SUM(order_details.quantity * pizzas.price),
5              2) AS Revenue
6  FROM
7      order_details
8          JOIN
9      pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

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

| Revenue |
|---|
| 817860.05 |

# ANSWER

# QUESTION 3:

MySQL Workbench

Local MySQL (pizza) - Warnin... | Local MySQL (pizza) - Warning - n...

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- credit_card
- **pizza**
- sys
- world

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7*

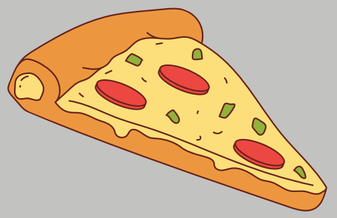
Limit to 1000 rows

```
1   # Question 3: 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   WHERE
10      pizzas.price = (SELECT
11              MAX(price)
12          FROM
13              pizzas);
```

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

| name | price |
|---|---|
| The Greek Pizza | 35.95 |

# ANSWER



# QUESTION 4:

MySQL Workbench

Local MySQL (pizza) - Warnin... | Local MySQL (pizza) - Warning - n...

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- credit_card
- **pizza**
- sys
- world

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7*

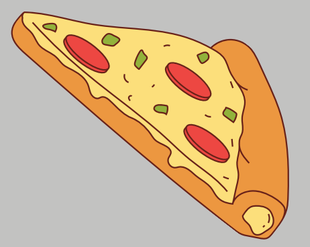
Limit to 1000 rows

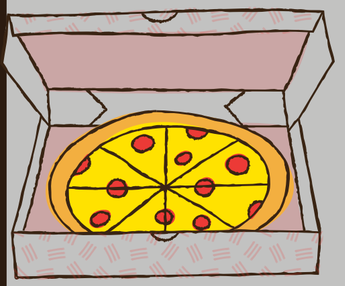
```
1   # Question 4: Identify the most common pizza size ordered.
2   SELECT
3       pizzas.size,
4       COUNT(order_details.order_details_id) AS order_count
5   FROM
6       pizzas
7           JOIN
8       order_details ON pizzas.pizza_id = order_details.pizza_id
9   GROUP BY pizzas.size
10  ORDER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

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

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

# ANSWER





# Question 5:

MySQL Workbench

Local MySQL (pizza) - Warnin... | Local MySQL (pizza) - Warning - n...

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- credit_card
- **pizza**
- sys
- world

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8*

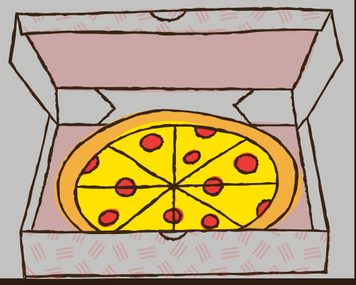
Limit to 1000 rows

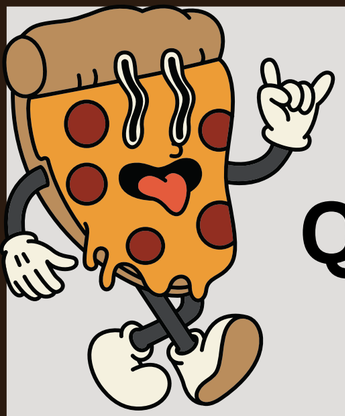
```sql
#  Question 5: List the top 5 most ordered pizza types along with their quantities.
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```

Result Grid | Filter 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 |

# Answer





# Questions 6:

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | SQL File 1*

Limit to 1000 rows

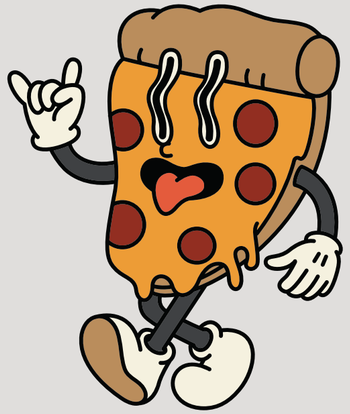
```sql
# Question 6: Join the necessary tables to find the total quantity of each pizza category ordered.

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

Result Grid | Filter Rows:

| category | quantity |
|---|---|
| Classic | 14888 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

# Answer



# Question 7:

SQL File 5 | SQL File 6 | SQL File 7 | SQL File 8 | SQL File 9 | SQL File 10

Limit to 1000 rows

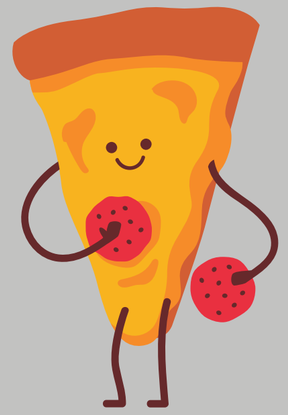
```sql
# Question 7: Determine the distribution of orders by hour of the day.

SELECT
    HOUR(order_time) as hour_of_the_day, COUNT(order_id) as total
FROM
    orders
    group by hour(order_time)
    order by hour_of_the_day ASC;
```

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

| hour_of_the_day | total |
|---|---|
| 9 | 1 |
| 10 | 8 |
| 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 |

# Answer



# QUESTION 8:

sql_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9*

Limit to 1000 rows

```sql
# Question 8: Join relevant tables to find the category-wise distribution of pizzas.
select category, count(name) as how_many
from pizza_types
group by category;
```

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

| category | how_many |
|---|---|
| Chicken | 6 |
| Classic | 8 |
| Supreme | 9 |
| Veggie | 9 |

# ANSWER

# QUESTION 9:

l_pizza* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | SQL File 10

Limit to 1000 rows

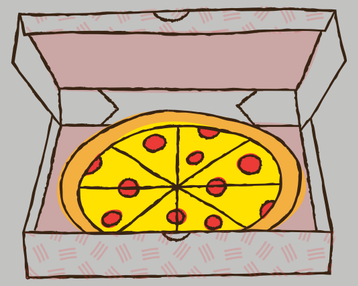
```sql
# Question 9: Group the orders by date and calculate the average number of pizzas ordered per day.

select round(avg(total),0)as average_order from
(select orders.order_date, sum(order_details.quantity) as total
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: | Exp

| | average_order |
|---|---|
| ▶ | 138 |

# ANSWER



# Question 10:

4* SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9* SQL File 10* SQL Fil

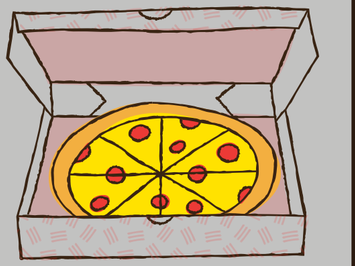
Limit to 1000 rows

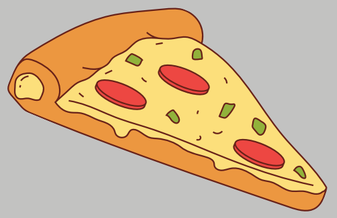
```
1  # Question 10: Determine the top 3 most ordered pizza types based on revenue.
2
3  select pizza_types.name,
4  sum(order_details.quantity*pizzas.price) as revenue
5  from pizza_types
6  join pizzas on pizzas.pizza_type_id=pizza_types.pizza_type_id
7  join order_details on order_details.pizza_id=pizzas.pizza_id
8  group by pizza_types.name
9  order by revenue desc limit 3;
```

Result Grid | Filter Rows:

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

# Answer





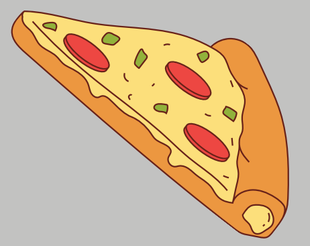
# QUESTION 11:

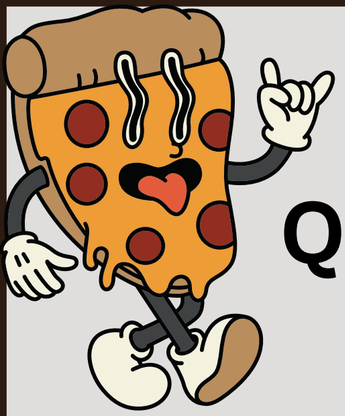
```
1  # Question 11: Calculate the percentage contribution of each pizza type to total revenue.
2
3  SELECT
4      pizza_types.category,
5      concat(round((SUM(order_details.quantity * pizzas.price) / (SELECT
6              ROUND(SUM(order_details.quantity * pizzas.price),
7                      2) AS total_sales
8          FROM
9              order_details
10             join pizzas on pizzas.pizza_id=order_details.pizza_id))*100,2),'%') as revenue
11 FROM
12     pizza_types
13         JOIN
14     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
15         JOIN
16     order_details ON pizzas.pizza_id = order_details.pizza_id
17 GROUP BY pizza_types.category
18 ORDER BY revenue DESC;
```

Result Grid | Filter Rows:

| category | revenue |
|---|---|
| Classic | 26.91% |
| Supreme | 25.46% |
| Chicken | 23.96% |
| Veggie | 23.68% |

# ANSWER





# Questions 12:

SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | SQL File 10* | SQL File 11*

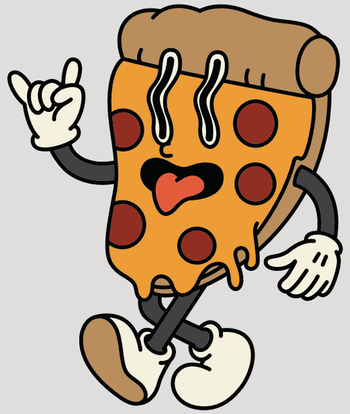
Limit to 1000 rows

```sql
# Question 12: Analyze the cumulative revenue generated over time.
select order_date, sum(revenue) over (order by order_date) as cumulative_ravenue
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 order_date) as sales;
```

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

| order_date | cumulative_ravenue |
|---|---|
| 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 |
| 2015-01-18 | 40978.600000000006 |
| 2015-01-19 | 43365.75000000001 |
| 2015-01-20 | 45763.65000000001 |
| 2015-01-21 | 47804.20000000001 |
| 2015-01-22 | 50300.90000000001 |
| 2015-01-23 | 52724.600000000006 |
| 2015-01-24 | 55013.850000000006 |

# Answer



# QUESTION 13:

SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | SQL File 10* | SQL File 11* | SQL File 12* | SQL

Limit to 1000 rows

```sql
# Question 13: 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: | Export: | Wr

| 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 |

# ANSWER

# THANK YOU!