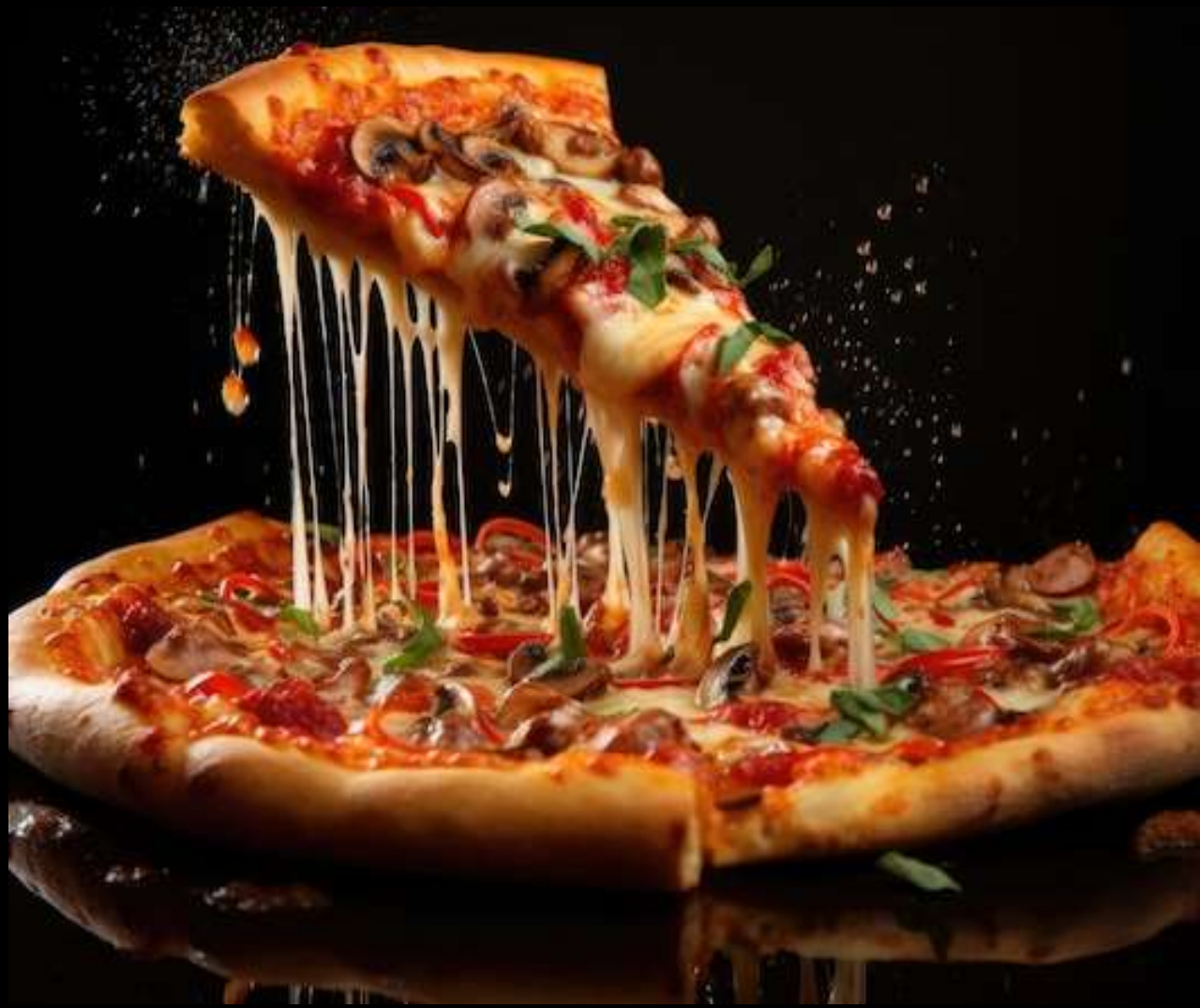
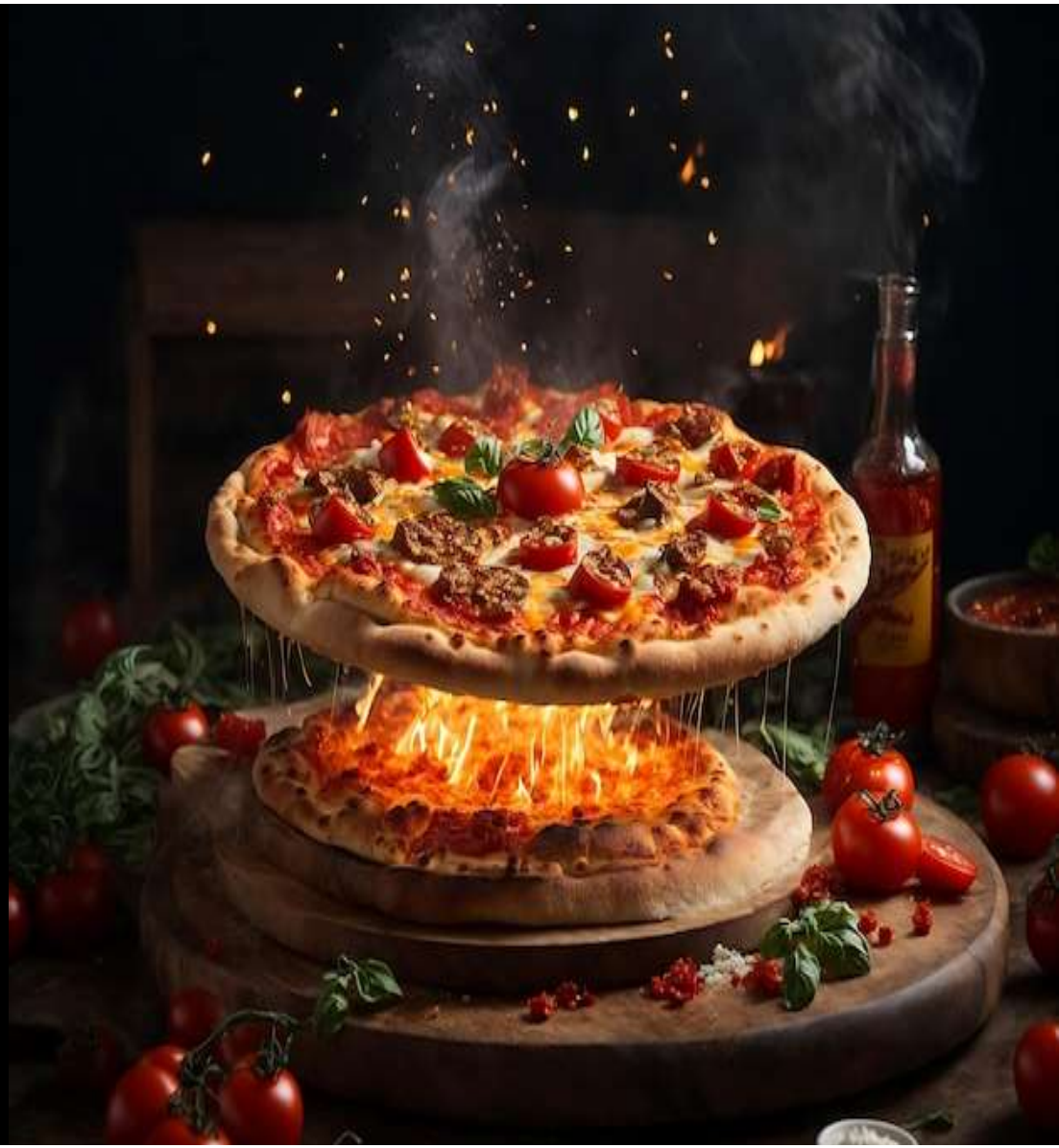
MySQL Project Pizza Sales Analysis

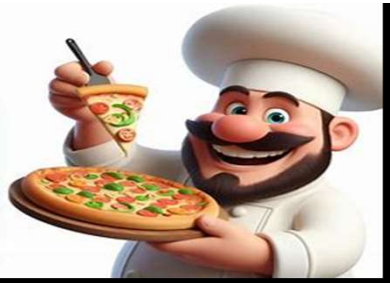
By :- Ashmi Hans



Objective

Analyze the pizza sales database to extract valuable insights about the sales trends and customer behaviour,examine the dataset and help to understand business-related questions and make the right strategies in the future.





Q1.Retrieve the total number of orders placed.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

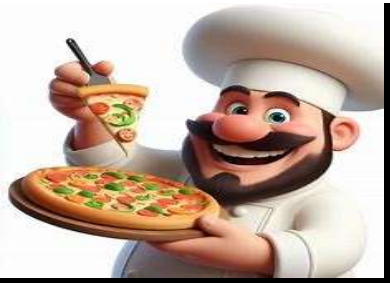
query-01 SQL File 3*

Limit to 1000 rows

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

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

total_orders
21350



Q2. Calculate the total revenue generated from pizza sales.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- ecommerce
- **pizzahut**
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

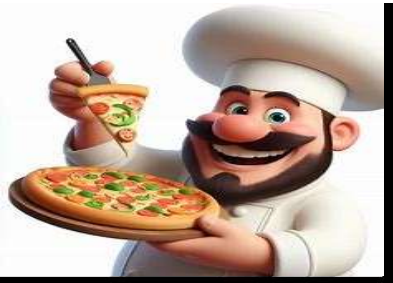
query-01 SQL File 3* SQL File 4*

Limit to 1000 rows

```
1   -- Calculate the total revenue generated from pizza sales.
2 • select
3   round(sum(orders_details.quantity*pizzas.price),2) as total_sales
4   from orders_details join pizzas
5   on pizzas.pizza_id = orders_details.pizza_id;
6
```

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

total_sales
817860.05



Q3. Identified the highest priced pizza.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- **pizzahut**
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

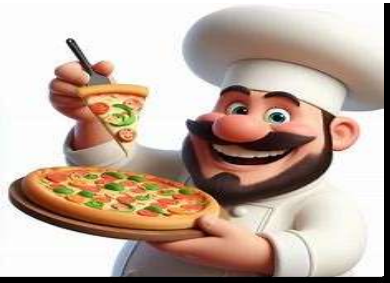
query-01 | SQL File 3* | SQL File 4* | SQL File 5*

Limit to 1000 rows

```
1   -- Identify the highest-priced pizza.
2   select
3   pizza_types.name,pizzas.price
4   from pizza_types join pizzas
5   on pizza_types.pizza_type_id = pizzas.pizza_type_id
6   order by pizzas.price desc
7   limit 1;
```

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

name	price
The Greek Pizza	35.95



Q4. Identified the most common pizza size ordered.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

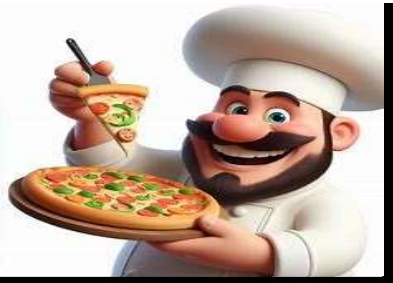
query-01 SQL File 3* SQL File 4* SQL File 5* SQL File 6*

Limit to 1000 rows

```
1   -- Identify the most common pizza size ordered.
2 • select pizzas.size,count(orders_details.order_details_id) as order_count
3   from pizzas join orders_details
4   on pizzas.pizza_id = orders_details.pizza_id
5   group by pizzas.size order by order_count desc;
```

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

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



Q5. List the top 5 most ordered pizza types along with their quantities.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

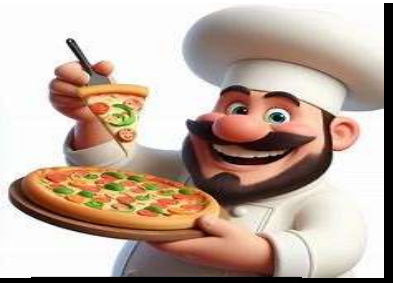
query-01 | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7*

Limit to 1000 rows

```
1   -- List the top 5 most ordered pizza types along with their quantities.
2 • select pizza_types.name,
3   sum(orders_details.quantity) as quantity
4   from pizza_types join pizzas
5   on pizza_types.pizza_type_id = pizzas.pizza_type_id
6   join orders_details
7   on orders_details.pizza_id = pizzas.pizza_id
8   group by pizza_types.name order by quantity desc limit 5;
```

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



Q6.Join the necessary tables to find the total quantity of each pizza category ordered.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

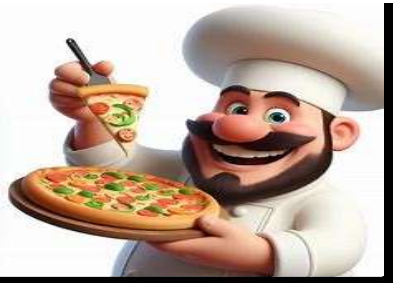
query-01 | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8*

Limit to 1000 rows

```
1  -- Join the necessary tables to find the total quantity of each pizza category ordered.
2  select pizza_types.category,
3  sum(orders_details.quantity) as quantity
4  from pizza_types join pizzas
5  on pizza_types.pizza_type_id = pizzas.pizza_type_id
6  join orders_details
7  on orders_details.pizza_id = pizzas.pizza_id
8  group by pizza_types.category order by quantity;
```

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

category	quantity
Chicken	11050
Veggie	11649
Supreme	11987
Classic	14888



Q7. Determine the distribution of orders by hour of the day.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

Administration Schemas

Information

Table: orders_details

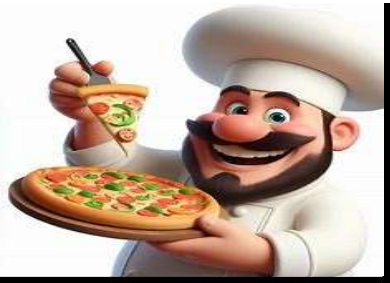
query-01 SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9*

Limit to 1000 rows

```
1   -- Determine the distribution of orders by hour of the day.
2 • select hour(order_time) as hours,
3   count(order_id) as order_count
4    from orders
5   group by
6    hour(order_time);
```

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

hours	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



Q8. Join relevant tables to find the category wise distribution of pizzas.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

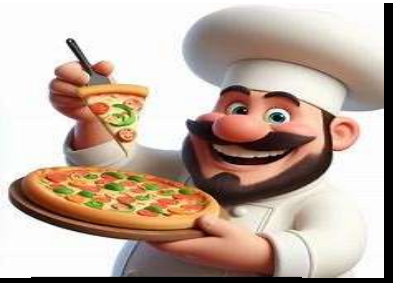
query-01 | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | SQL File 10*

Limit to 1000 rows

```
1  -- Join relevant tables to find the category-wise distribution of pizzas.
2  select category,count(name)
3  from pizza_types
4  group by category;
```

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

category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9



Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

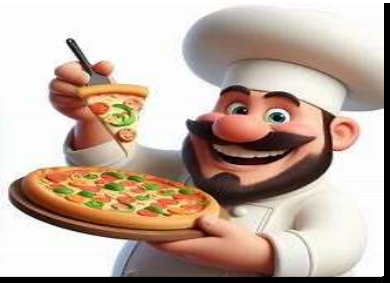
query-01 SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9* SQL File 10*

Limit to 1000 rows

```
1  -- Group the orders by date and calculate the average number of pizzas ordered per day.
2  SELECT round(AVG(daily_quantity),0) AS avg_pizzas_per_day
3  FROM (
4      SELECT orders.order_date, SUM(orders_details.quantity) AS daily_quantity
5      FROM orders
6      JOIN orders_details ON orders.order_id = orders_details.order_id
7      GROUP BY orders.order_date
8  ) AS order_quantity;
9
```

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

avg_pizzas_per_day
138



Q10. Determine the top 3 most ordered types based on revenue.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

Administration Schemas

Information

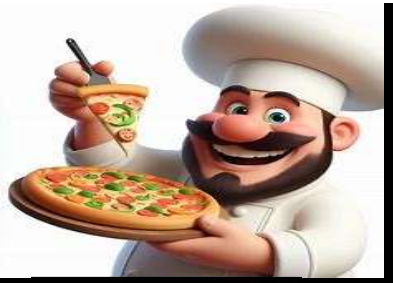
query-01 SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* SQL F

Limit to 1000 rows

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

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

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



Q11. Calculate the percentage contribution of each pizza type to total revenue.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator:

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

Administration Schemas

Information

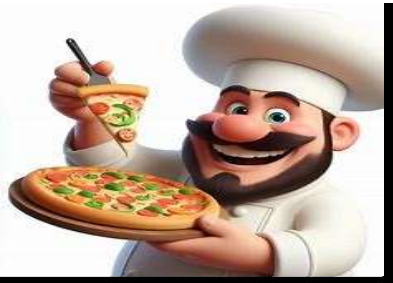
query-01 SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9*

Limit to 1000 rows

```
1   -- Calculate the percentage contribution of each pizza type to total revenue.
2   select pizza_types.category,
3   ROUND(
4       (sum(orders_details.quantity * pizzas.price) /
5       (select sum(orders_details.quantity * pizzas.price)
6       from orders_details join pizzas
7       on pizzas.pizza_id = orders_details.pizza_id) * 100), 2) as revenue
8   from pizza_types
9   join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id
10  join orders_details on orders_details.pizza_id = pizzas.pizza_id
11  group by pizza_types.category
12  order by revenue desc
13  LIMIT 0, 1000;
14
```

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

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68



Q12. Analyze the cumulative revenue generated over time.

MySQL Workbench

Local instance MySQL80 (pizzahut) | Local instance MySQL80 (piz...

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

Administration Schemas

Information

No object selected

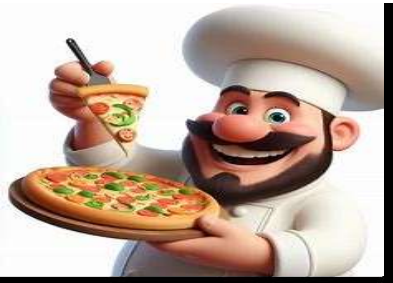
Query 1

Limit to 1000 rows

```
1   -- Analyze the cumulative revenue generated over time.
2 • select order_date,
3   sum(revenue) over(order by order_date) as cum_revenue
4   from
5   (select orders.order_date,
6   sum(orders_details.quantity*pizzas.price) as revenue
7   from orders_details join pizzas
8   on orders_details.pizza_id = pizzas.pizza_id
9   join orders
10  on orders.order_id = orders_details.order_id
11  group by orders.order_date) as sales;
```

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

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



Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

MySQL Workbench

Local instance MySQL80 (pizzahut) | Local instance MySQL80 (piz...

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- ecommerce
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions
- sys

Administration Schemas

Information

No object selected

Query 1 | SQL File 1*

Limit to 1000 rows

```
1   -- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
2   select name,revenue from
3   (select category,name,revenue,
4   rank() over(partition by category order by revenue desc) as rn
5   from
6   (select pizza_types.category,pizza_types.name,
7   sum((orders_details.quantity)*pizzas.price) as revenue
8   from pizza_types join pizzas
9   on pizza_types.pizza_type_id = pizzas.pizza_type_id
10  join orders_details
11  on orders_details.pizza_id = pizzas.pizza_id
12  group by pizza_types.category,pizza_types.name) as a) as b where rn<=3;
```

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

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

Result 2

THANK YOU