



PIZZERIA

FRESH & TASTY

SQL INSIGHTS INTO PIZZA SALES





PIZZERIA
FRESH & TASTY



INTRODUCTION

In this presentation, we will explore pizza sales data using SQL queries at three levels of complexity: Basic, Intermediate, and Advanced. The goal is to extract meaningful insights from the data, ranging from simple counts and totals to more complex patterns like distribution by category, contribution to revenue, and cumulative analysis.

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED



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

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

	total_orders
▶	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES



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

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	total_sales			
▶	817860.05			

IDENTIFY THE HIGHEST-PRICED PIZZA

```
1  -- Identify the highest-priced pizza.  
2  
3 • select 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 limit 1;
```

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

name	price
The Greek Pizza	35.95


ESTD 2019
PIZZERIA
FRESH & TASTY

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```
1      -- Identify the most common pizza size ordered.  
2  
3 •  select pizzas.size, count(orders_details.order_details_id) as order_count  
4    from pizzas join orders_details  
5      on pizzas.pizza_id = orders_details.pizza_id  
6    group by pizzas.size order by order_count desc;  
7
```

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

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

```
1  -- List the top 5 most ordered pizza types along with their quantities.  
2  
3 • select pizza_types.name,  
4     sum(orders_details.quantity) as quantity  
5     from pizza_types join pizzas  
6     on pizza_types.pizza_type_id = pizzas.pizza_type_id  
7     join orders_details  
8     on orders_details.pizza_id = pizzas.pizza_id  
9     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

JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED

The screenshot shows a MySQL Workbench interface with a query editor and a results grid.

Query Editor:

```
1 -- Join the necessary tables to find the total quantity of each pizza category ordered.  
2  
3 • select pizza_types.category,  
4     sum(orders_details.quantity) as quantity  
5     from pizza_types join pizzas  
6     on pizza_types.pizza_type_id = pizzas.pizza_type_id  
7     join orders_details  
8     on orders_details.pizza_id = pizzas.pizza_id  
9     group by pizza_types.category order by quantity desc;  
10
```

Results Grid:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

The screenshot shows a MySQL query editor interface. At the top, there is a toolbar with various icons for file operations, search, and database management. Below the toolbar, a status bar displays "Limit to 1000 rows". The main area contains a SQL query:

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

Below the query, a "Result Grid" is displayed, showing the results of the executed query:

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

The screenshot shows a database interface with a toolbar at the top containing various icons for file operations, search, and navigation. A message bar indicates "Limit to 1000 rows". The main area displays a numbered SQL script:

```
1 -- Join relevant tables to find the category-wise distribution of pizzas.  
2  
3 • select category, count(name) from pizza_types  
4 group by category;  
5
```

Below the script is a results grid with the following data:

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

At the bottom right, there is a small pizzeria logo with the text "PIZZERIA FRESH & TASTY".

GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY



```
1  -- Group the orders by date and calculate the average number of pizzas ordered per day.  
2  
3  • select round(avg(quantity),0) from  
4  (select orders.order_date, sum(orders_details.quantity) as quantity  
5  from orders join orders_details  
6  on orders.order_id = orders_details.order_id  
7  group by orders.order_date) as orders_quantity  
8
```

Result Grid		Filter Rows:	<input type="text"/>	Export:		Wrap Cell Content:	
			round(avg(quantity),0)				
▶			138				

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE



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

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

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE

```
1  -- Calculate the percentage contribution of each pizza type to total revenue.
2
3 • SELECT pizza_types.category,
4   ROUND(SUM(orders_details.quantity * pizzas.price) /
5     (SELECT SUM(orders_details.quantity * pizzas.price)
6      FROM orders_details
7        JOIN pizzas 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
```

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

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68



THANK YOU