

SQL Database Project

Utilizing SQL Queries for Database Problem Solving

[Utkarsh Srivastava]

[Project]

Project Overview

- Project Objectives
 - Demonstrate practical application of SQL queries
 - Solve complex database-related problems
 - Apply various SQL operations and techniques
 - Focus on data retrieval, analysis, and optimization
 - Develop practical database management skills

Database Structure

- Database Design Principles
 - Designed normalized database schema
 - Implemented multiple related tables
 - Ensured data integrity and consistency

Technologies Used

- Technical Stack
 - SQL (Structured Query Language)
 - MySQL/PostgreSQL/SQL Server Database
 - Database Management Tools (phpMyAdmin, SQL Server Management Studio)
 - Query optimization techniques
 - Data modeling and normalization
 - ERD (Entity Relationship Diagram) tools

SQL Queries Implemented

- Types of SQL Operations Used
 - SELECT statements for data retrieval
 - JOIN operations (INNER, LEFT, RIGHT, FULL)
 - Aggregate functions (COUNT, SUM, AVG, MAX, MIN)
 - WHERE clauses for data filtering
 - GROUP BY and ORDER BY operations
 - Subqueries and nested queries

Data set

- Data Set

Link=[https://github.com/uttu3690/My SQL Pizza Queries Project](https://github.com/uttu3690/My_SQL_Pizza_Queries_Project)

The Queries

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.

1 Queries

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

Result Grid			 Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	total_id				
▶	21350				

2 Queries

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

Result Grid		 Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	total_sales			
▶	817860.05			

3 Queries

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

Result Grid



Filter Rows:

Export:



Wrap Cell Content:



Fetch rows:

	name	price
▶	The Greek Pizza	35.95

4 Queries

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

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	size	order_c			
▶	L	18526			
	M	15385			
	S	14137			
	XL	544			
	XXL	28			

5 Queries

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

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				

6 Queries

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

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

7 Queries

```
1  -- Determine the distribution of orders by hour of the day--
2  ●  SELECT
3      HOUR(order_time) AS hour, COUNT(order_id) AS order_count
4  FROM
5      orders
6  GROUP BY HOUR(order_time);
```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	hour	order_count			
▶	11	1231			
	12	2520			
	13	2455			
	14	1472			
	15	1468			

8 Queries

```
1  -- Join relevant tables to find the category-wise distribution of pizzas.
2  •  SELECT
3      category, COUNT(name)
4  FROM
5      pizza_types
6  GROUP BY category;
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

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

9 Queries

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

Result Grid



Filter Rows:

Export:








Wrap Cell Content:

	avg_pizza_order_perday
▶	138

10 Queries

```
1  -- Determine the top 3 most ordered pizza types based on revenue.
2  • SELECT
3      pizza_types.name,
4      SUM(order_details.quantity * pizzas.price) AS revenue
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
9      JOIN
10     order_details ON order_details.pizza_id = pizzas.pizza_id
11 GROUP BY pizza_types.name
12 ORDER BY revenue DESC
13 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

11 Queries

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

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

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

12 Queries

```
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
6      (select orders.order_date,
7         sum(order_details.quantity* pizzas.price)as revenue
8         from order_details join pizzas
9         on order_details.pizza_id=pizzas.pizza_id
10        join orders
11        on orders.order_id=order_details.order_id
12        group by orders.order_date) as sales;
```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	order_date	cum_revenue			
▶	2015-01-01	2713.85000000000004			
	2015-01-02	5445.75			
	2015-01-03	8108.15			
	2015-01-04	9863.6			
	2015-01-05	11929.55			
	2015-01-06	14250.5			

13 Queries

```
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,rank() over(partition by category order by revenue desc) as rn from
4  (select pizza_types.category,pizza_types.name,
5   sum((order_details.quantity)*pizzas.price)as revenue
6   from pizza_types join pizzas
7   on pizza_types.pizza_type_id =pizzas.pizza_type_id
8   join order_details
9   on order_details.pizza_id=pizzas.pizza_id
10  group by pizza_types.category,pizza_types.name) as a) as b
11  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			
	The			

Results and Analysis

- Project Outcomes
 - Successfully retrieved and analyzed large datasets
 - Identified key business insights through data queries
 - Optimized query performance using indexes
 - Generated meaningful reports for decision-making
 - Improved data accuracy and consistency
 - Reduced query execution time by 40%

Conclusion

- Project Summary
 - Successfully demonstrated SQL proficiency
 - Solved real-world database problems
 - Developed scalable and efficient solutions
 - Ready to apply skills in professional environment
 - Future: Explore advanced SQL features and big data tools
 - Prepared for database administrator/analyst roles