Where Every Slice is a Taste of Perfection

PIZZA RESTO

SQL Project

Pizza Restaurant Sales & Customer Analytics

Database: pizza_resto | Tool: MySQL

ORDER NOW

By: Loveleen Arora

OBJECTIVE

To analyze Pizza Restaurant data using SQL by exploring:

- ✅ Sales trends
- ✅ Customer behavior
- ✅ Product performance
- ✅ Advanced SQL analytics

DATABASE CREATED : `PIZZA_RESTO`

SQL projcet 1*

Don't Limit

```
1   /*## DATA BASE ##*/
2   CREATE DATABASE PIZZA_RESTO;
3   SHOW DATABASES;
4   use pizza_resto;
5   /*## TABLES ##*/
6   show tables;
7   /*## EXISTING  CUSTOMERS COLUMN DATA TYPE CHANGE ##*/
```

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

Database
information_schema
my_new
mysql
performance_schema
pizza_resto
sakila
sys
world

TABLE USED

1. pizza_cx – Customer Info
2. pizza_ord – Orders
3. pizza_cet – Pizza Categories

SQL projcet 1*

Don't Limit

```
4  use pizza_resto;
5  /*## TABLES ##*/
6  show tables;
7  /*## EXISTING  CUSTOMERS COLUMN DATA TYPE CHANGE ##*/
8  desc pizza_cx;
9  alter table pizza_cx modify customer_id varchar(100)not null;
10 alter table pizza_cx modify column Customer_Name varchar(100)not null;
```

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

Tables_in_pizza_resto
pizza_cet
pizza_cx
pizza_ord
pizza_revenue_view

DATA CLEANING

- Modified column types (VARCHAR, DECIMAL, DATE)
- Applied NOT NULL constraints

3. pizza_cet – Pizza Categories

SQL projcet 1*

Don't Limit

```
10 • alter table pizza_cx modify column Customer_Name varchar(100)not null;
11 • alter table pizza_cx modify column Phone_Number VARCHAR(15)not null;
12   /*## EXISTING  CATEGORY COLUMN DATA TYPE CHANGE ##*/
13 • desc pizza_cet;
14 • alter table pizza_cet modify pizza_id varchar(100)not null;
15 • alter table pizza_cet modify pizza_name varchar(100)not null;
16 • alter table pizza_cet modify category varchar(100)not null;
```

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

Field	Type	Null	Key	Default	Extra
pizza_id	varchar(100)	NO	PRI	NULL	
pizza_name	varchar(100)	NO		NULL	
category	varchar(100)	NO		NULL	
Quantity	int	YES		NULL	
customer_id	varchar(100)	NO		NULL	

DATA CLEANING

Verified indexes with `SHOW INDEX`

Goal: Structured and reliable data for analytics

SQL projcet 1*

Don't Limit

```
23 •  alter table pizza_ord modify order_date date not null;
24 •  alter table pizza_ord modify customer_ID varchar(100)not null;
25    /*## VERIFIED INDEXS ##*/
26 •  show index from pizza_cx;
27 •  show index from pizza_ord;
28 •  show index from pizza_cet;
29    /*## VIEW ALL DATA FROM TABLES */
```

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

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
pizza_ord	0	PRIMARY	1	Order_ID	A	500	NULL	NULL		BTREE			YES	NULL
pizza_ord	1	customer_ID	1	customer_ID	A	500	NULL	NULL		BTREE			YES	NULL

SALES & REVENUE ANALYSIS

Key Sales Metrics:

- Total Customers
- Total Orders:
- Total Revenue
- Average Order Value
- Total Pizzas Sold

Used SQL functions: COUNT(), SUM(), AVG()

Total Customers

SQL projcet 1*

Don't Limit

```
32 •  select* from pizza_ord;
33    /*## SALES & CUSTOMER ANALYSIS ##*/
34    /*## Query 1: Total Customers ##*/
35 •  select count(*)  as total_customers from pizza_cx;
36    /*## Query 2:Total Orders ##*/
37 •  select count(*)  as total_orders from pizza_ord;
38    /*## Query 3:Total Revenue ##*/
```

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

total_customers
500

SALES & REVENUE ANALYSIS

Total Orders

SQL projcet 1*

Don't Limit

```
32 •  select* from pizza_ord;
33    /*## SALES & CUSTOMER ANALYSIS ##*/
34    /*## Query 1: Total Customers ##*/
35 •  select count(*)  as total_customers from pizza_cx;
36    /*## Query 2:Total Orders ##*/
37 •  select count(*)  as total_orders from pizza_ord;
38    /*## Query 3:Total Revenue ##*/
```

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

total_orders
500

SALES & REVENUE ANALYSIS

Total Revenue

SQL projcet 1*

Don't Limit

```
35 •  select count(*)  as total_customers from pizza_cx;
36    /*## Query 2:Total Orders ##*/
37 •  select count(*)  as total_orders from pizza_ord;
38    /*## Query 3:Total Revenue ##*/
39 •  select sum(order_amount) as  total_revenue from pizza_ord;
40    /*## Query 4:Total Pizzas Sold ##*/
41 •  select sum(quantity ) as total_quantity from pizza_cet;
```

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

total_revenue
551310.780

SALES & REVENUE ANALYSIS

Average Order Value

SQL projcet 1*

Don't Limit

```
38    /*## Query 3:Total Revenue ##*/
39 •  select sum(order_amount) as  total_revenue from pizza_ord;
40    /*## Query 4:Total Pizzas Sold ##*/
41 •  select sum(quantity ) as total_quantity from pizza_cet;
42    /*## Query 5:Average  Revenue ##*/
43 •  select avg(order_amount) as  average_revenu from pizza_ord;
44    /*## PRODUCT INSIGHTS  ##*/
```

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

average_revenu
1102.6215600

SALES & REVENUE ANALYSIS

Total Pizzas Sold

SQL projcet 1*

Don't Limit

```
38    /*## Query 3:Total Revenue ##*/
39 •  select sum(order_amount) as  total_revenue from pizza_ord;
40    /*## Query 4:Total Pizzas Sold ##*/
41 •  select sum(quantity ) as total_quantity from pizza_cet;
42    /*## Query 5:Average  Revenue ##*/
43 •  select avg(order_amount) as  average_revenu from pizza_ord;
44    /*## PRODUCT INSIGHTS  ##*/
```

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

total_quantity
1492

PRODUCT INSIGHTS

Top 5 Best-Selling Pizzas

Used GROUP BY + ORDER BY + JOIN

SQL projcet 1*

Don't Limit

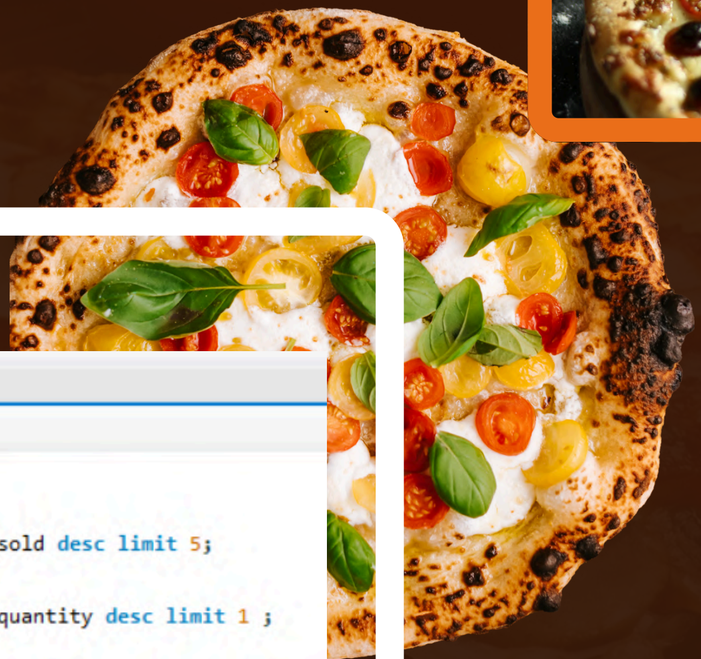
```
44  /*## PRODUCT INSIGHTS  ##*/
45  /*## Query 1: Top 5 Best-Selling Pizzas ##*/
46  select pizza_name ,sum(quantity) as total_sold from pizza_cet group by pizza_name order by total_sold desc limit 5;
47  /*## Query 2: Most Ordered Category ##*/
48  select category, sum(quantity) as total_quantity from pizza_cet group by category order by total_quantity desc limit 1 ;
49  /*## Query 3:Least Selling Pizzas##*/
50  select pizza_name, sum(quantity) as total_quantity from pizza_cet group by pizza_name order by total_quantity asc limit 1
```

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

pizza_name	total_sold
Peppy Paneer	152
Paneer Tikka	126
Veggie Delight	124
Deluxe Veggie	119
Mexican Green Wave	119

PRODUCT INSIGHTS

Least Selling Pizza



SQL projcet 1*

Don't Limit

```sql
/*## PRODUCT INSIGHTS  ##*/
/*## Query 1: Top 5 Best-Selling Pizzas ##*/
select pizza_name ,sum(quantity) as total_sold from pizza_cet group by pizza_name order by total_sold desc limit 5;
/*## Query 2: Most Ordered Category ##*/
select category, sum(quantity) as total_quantity from pizza_cet group by category order by total_quantity desc limit 1 ;
/*## Query 3:Least Selling Pizzas##*/
select pizza_name, sum(quantity) as total_quantity from pizza_cet group by pizza_name order by total_quantity asc limit 1 ;
```

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

pizza_name	total_quantity
Cheese Burst	93

PRODUCT INSIGHTS

Most Ordered Category

SQL projcet 1*

Don't Limit

```
44    /*## PRODUCT INSIGHTS  ##*/
45    /*## Query 1: Top 5 Best-Selling Pizzas ##*/
46 •  select pizza_name ,sum(quantity) as total_sold from pizza_cet group by pizza_name order by total_sold desc limit 5;
47    /*## Query 2: Most Ordered Category ##*/
48 •  select category, sum(quantity) as total_quantity from pizza_cet group by category order by total_quantity desc limit 1 ;
49    /*## Query 3:Least Selling Pizzas##*/
50 •  select pizza_name, sum(quantity) as total_quantity from pizza_cet group by pizza_name order by total_quantity asc limit 1 ;
```

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

category	total_quantity
Non-Veg	864

PRODUCT INSIGHTS

Revenue by Pizza

SQL projcet 1*

Don't Limit

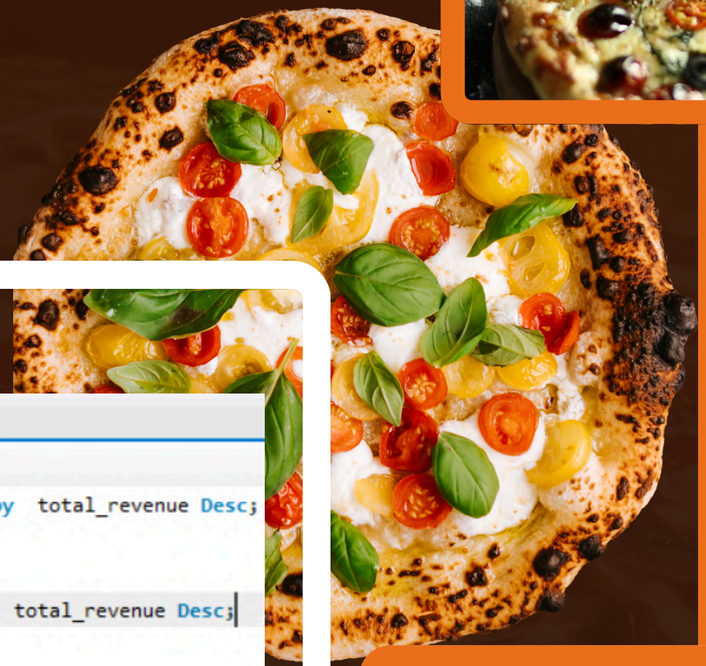
```
47    /*## Query 2: Most Ordered Category ##*/
48 •  select category, sum(quantity) as total_quantity from pizza_cet group by category order by total_quantity desc limit 1 ;
49    /*## Query 3:Least Selling Pizzas##*/
50 •  select pizza_name, sum(quantity) as total_quantity from pizza_cet group by pizza_name order by total_quantity asc limit 1 ;
51    /*##Query 4:Revenue by Pizzas ##*/
52 •  Select  pizza_cet.pizza_name, SUM(pizza_ord.order_amount) as total_revenue from pizza_cet
53    inner join pizza_ord on pizza_cet.customer_id = pizza_ord.customer_id group by  pizza_cet.pizza_name order by  total_revenue Desc;
```

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

pizza_name	total_revenue
Mexican Green Wave	49286.810
Peppy Paneer	49052.330
Chicken Dominator	48211.410
Paneer Tikka	47530.270
Veggie Delight	43369.050
Spicy Triple Tango	42839.970
Farmhouse	42094.620
Deluxe Veggie	42066.220
Margherita	41520.760
Golden Corn	38893.120
Pepperoni	36385.560

PRODUCT INSIGHTS

Revenue by Category



SQL projcet 1*

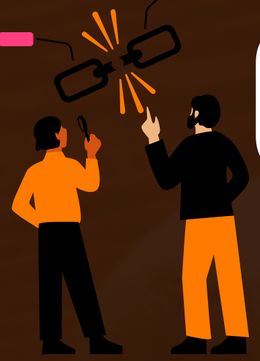
Don't Limit

```
53    inner join pizza_ord on pizza_cet.customer_id = pizza_ord.customer_id group by  pizza_cet.pizza_name order by  total_revenue Desc;
54    /*## Query 5:Revenue by Category ##*/
55    Select  pizza_cet.category, SUM(pizza_ord.order_amount) as total_revenue from pizza_cet
56    inner join pizza_ord on pizza_cet.customer_id = pizza_ord.customer_id group by  pizza_cet.category order by  total_revenue Desc;
57    /*## CUSTOMER BEHAVIOUR ##*/
58    /*## Query 1:Highest Spending Customers ##*/
59    Select  pizza_cx.customer_name, SUM(pizza_ord.order_amount) as total_revenue from pizza_cx
```

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

category	total_revenue
Non-Veg	330399.790
Veg	220910.990

CUSTOMER ANALYSIS



Highest Spending Customer

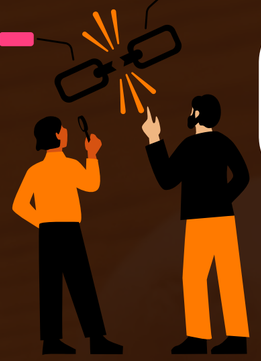
SQL projcet 1*

```
56  inner join pizza_ord on pizza_cet.customer_id = pizza_ord.customer_id group by  pizza_cet.category order by  total_revenue Desc;
57  /*## CUSTOMER BEHAVIOUR ##*/
58  /*## Query 1:Highest Spending Customers ##*/
59  Select  pizza_cx.customer_name, SUM(pizza_ord.order_amount) as total_revenue from pizza_cx
60  inner join pizza_ord on pizza_cx.customer_id = pizza_ord.customer_id group by  pizza_cx.customer_name order by  total_revenue Desc limit 1;
61  /*## Query 2:Lowest Spending Customers ##*/
62  Select  pizza_cx.customer_name, SUM(pizza_ord.order_amount) as total_revenue from pizza_cx
```

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

customer_name	total_revenue
Urvi Mall	1999.040

CUSTOMER ANALYSIS



Lowest Spending Customer

SQL projcet 1*

Don't Limit

```
59 •  Select  pizza_cx.customer_name, SUM(pizza_ord.order_amount) as total_revenue from pizza_cx
60    inner join pizza_ord on pizza_cx.customer_id = pizza_ord.customer_id group by  pizza_cx.customer_name order by  total_revenue Desc limit 1;
61    /*## Query 2:Lowest Spending Customers ##*/
62 •  Select  pizza_cx.customer_name, SUM(pizza_ord.order_amount) as total_revenue from pizza_cx
63    inner join pizza_ord on pizza_cx.customer_id = pizza_ord.customer_id group by  pizza_cx.customer_name order by  total_revenue asc limit 1;
64    /*##  Repeat Customers (More than 1 order) ##*/
65 •  select order_id, count(order_id) as order_count from pizza_ord group by order_id having count(order_id) > 1;
```

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

customer_name	total_revenue
Ela Vyas	203.740

ADVANCED QUERIES USED

Window Functions: RANK()

Real-world data analysis skills applied

SQL projcet 1*

```
65  select order_id, count(order_id) as order_count from pizza_ord group by order_id having count(order_id) > 1;
66  /*## Advanced Analysis ##*/
67  /*##  Query 1: Rank Customers by Total Spend ##*/
68  select pizza_cx.customer_name,  sum(pizza_ord.order_amount) as total_spend ,
69  rank() over (order by sum(pizza_ord.order_amount) desc) as customer_rank from pizza_cx
70  join pizza_ord on pizza_cx.customer_id=pizza_ord.customer_id group by pizza_cx.customer_name ;
71  /*## Query 2:Rank Pizzas by Quantity Sold ##*/
```

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

customer_name	total_spend	customer_rank
Urvi Mall	1999.040	1
Jivika Balasubramanian	1996.320	2
Mannat Lall	1987.560	3
Kanav Deol	1986.450	4
Divyansh Chand	1986.210	5
Nehmat Karan	1974.090	6
Adah Shan	1969.820	7
Misha Dora	1969.040	8
Tara Grover	1965.220	9
Navya Uppal	1965.190	10
Kaira Doshi	1963.700	11

ADVANCED QUERIES USED

Window Functions: DENSE_RANK()

SQL project 1*

Don't Limit

```
68 •  select pizza_cx.customer_name,  sum(pizza_ord.order_amount) as total_spend ,
69    rank() over (order by sum(pizza_ord.order_amount) desc) as customer_rank from pizza_cx
70    join pizza_ord on pizza_cx.customer_id=pizza_ord.customer_id group by pizza_cx.customer_name ;
71    /*## Query 2:Rank Pizzas by Quantity Sold ##*/
72 •  Select pizza_name, Sum(quantity) as total_quantity, dense_rank() over (order by sum(quantity) desc) as customer_rank
73    from pizza_cet group by pizza_name;
74    /*## VIEW CREATION ##*/
```

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

pizza_name	total_quantity	customer_rank
Peppy Paneer	152	1
Paneer Tikka	126	2
Veggie Delight	124	3
Mexican Green Wave	119	4
Deluxe Veggie	119	4
Margherita	119	4
Farmhouse	115	5
Pepperoni	110	6
Chicken Dominator	109	7
Golden Corn	107	8
Spicy Triple Tango	105	9

ADVANCED QUERIES USED

CREATE VIEW: Pizza-wise Revenue

SQL projcet 1*

```
74  /*## VIEW CREATION ##*/
75  /*##Query 3:View for Pizza-wise Revenue ##*/
76  Create view pizza_revenue_view as select pizza_cet.pizza_name , sum(pizza_ord.order_amount) as total_revenue from pizza_cet
77  join pizza_ord  on pizza_cet.customer_id=pizza_ord.customer_id group by pizza_name;
78  select * from pizza_revenue_view;
79  /*## Query 4:Revenue by Month ##*/
80  select month (order_date) as order_month ,sum(order_amount) as monthly_revenue from pizza_ord
```

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

pizza_name	total_revenue
Chicken Dominator	48211.410
Cheese Burst	36070.730
Farmhouse	42094.620
Paneer Tikka	47530.270
Peppy Paneer	49052.330
Mexican Green Wave	49286.810
BBQ Chicken	33989.930
Veggie Delight	43369.050
Deluxe Veggie	42066.220
Spicy Triple Tango	42839.970
Pepperoni	36385.560

ADVANCED QUERIES USED

Monthly Trends using GROUP BY Month()

SQL projcet 1*

```
77    join pizza_ord  on pizza_cet.customer_id=pizza_ord.customer_id group by pizza_name;
78    select * from pizza_revenue_view;
79    /*## Query 4:Revenue by Month ##*/
80    select month (order_date) as order_month ,sum(order_amount) as monthly_revenue from pizza_ord
81    group by month(order_date) order by order_month;
82    /*## Use of CTE (Common Table Expressions) ##*/
83    /*## Query :5 Repeat Customers (More than 1 Order)##*/
```

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

order_month	monthly_revenue
1	288654.870
2	262655.910

ADVANCED QUERIES USED

CTE (WITH): Repeat Customer Logic

SQL projcet 1*

Don't Limit

```
82  /*## Use of CTE (Common Table Expressions) ##*/
83  /*## Query :5 Repeat Customers (More than 1 Order)##*/
84  with order_count as (
85   select  customer_id, COUNT(order_id) as total_orders
86   from pizza_ord
87   group by customer_id
88   )
89
90    select  pizza_cx.customer_name, pizza_ord.order_id
91    from order_count
92    join pizza_cx on pizza_cx.customer_id = order_count.customer_id
93    join pizza_ord on pizza_ord.customer_id = pizza_cx.customer_id
94    where  order_count.total_orders > 1;
```

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

customer_name	order_id

CONCLUSION

In this SQL-based Pizza Restaurant Project, I performed deep-dive analysis in the following key areas:

Sales Analysis

Total Customers , Total Orders , Total Revenue , Average Order Value , Total Pizzas Sold

Customer Behavior

Highest Spending Customer & Lowest Spending Customer

Product Insights

Top 5 Best-Selling Pizzas , Least Selling Pizza , Most Ordered Category , Revenue by Pizza ,Revenue by Category

Advanced SQL Techniques

- Used **Window Functions (RANK, DENSE_RANK)**
- Created **Views** for Pizza-wise revenue
- Used **CTEs** to identify loyal customers

Outcome: Gained real-world business insights using only SQL. This project builds a strong base for creating a powerful **Power BI Dashboard** in the next step.