



LARANA PIZZA

WELCOME TO  
**LARANA PIZZA**  
BY GURU





LARANA PIZZA



# VISSION & MISSION

## VISSION

The vision of this project is to leverage data-driven insights from pizza sales to enhance decision-making, optimize inventory management, and boost overall profitability. By analyzing customer preferences and sales trends, we aim to drive more targeted marketing and operational efficiency.

## MISSION

In this data analysis project, we utilized MySQL to analyze pizza sales data by querying key metrics such as total sales by day, average order value, and popular pizza types. We created SQL queries to answer specific questions like "Which pizza flavor generates the highest revenue?" and "What is the average number of pizzas sold per transaction?" This approach helped uncover trends, optimize inventory, and provide actionable insights for improving sales strategies.



LARANA PIZZA

# HELLO!!

MY NAME IS GURUDAS U. JADHAV  
HERE ONWARDS PROJECT  
BEGINS!



-- RETRIEVE THE TOTAL  
NUMBER OF ORDERS  
PLACED.

The screenshot shows a MySQL Workbench interface with a query editor and a result grid. The query is:

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

The result grid displays a single row with the column 'total\_orders' containing the value '21350'.

total_orders
21350



-- CALCULATE THE  
TOTAL REVENUE  
GENERATED FROM PIZZA  
SALES.

Query 1 × order\_details pizza\_types pizzas orders

11  
12 -- Calculate the total revenue generated from pizza sales.  
13 • SELECT  
14     ROUND(SUM(order\_details.quantity \* pizzas.price),  
15                         2)  
16 FROM  
17     order\_details  
18     JOIN  
19     pizzas ON order\_details.pizza\_id = pizzas.pizza\_id;

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

round(sum(order_details.quantity * pizzas.price),2)
817860.05

Result 4 ×



LARANA PIZZA

## -- IDENTIFY THE HIGHEST-PRICED PIZZA.

Query 1 x order\_details pizza\_types pizzas orders

21 -- Identify the highest-priced pizza.

22 • SELECT

23 pizza\_types.name, pizzas.price

24 FROM

25 pizza\_types

26 JOIN

27 pizzas ON pizza\_types.pizza\_type\_id = pizzas.pizza\_type\_id

28 ORDER BY pizzas.price DESC

29 LIMIT 1;

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

	name	price
▶	The Greek Pizza	35.95



LARANA PIZZA

-- IDENTIFY THE MOST  
COMMON PIZZA SIZE  
ORDERED.

Query 1    order\_details    pizza\_types    pizzas    orders

39

40 • SELECT

41        pizzas.size, COUNT(order\_details\_id)

42 FROM

43        pizzas

44        JOIN

45        order\_details ON pizzas.pizza\_id = order\_details.pizza\_id

46 GROUP BY pizzas.size

47 LIMIT 1;

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

	size	COUNT(order_details_id)
▶	M	15385



## -- LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES

Query 1    order\_details    pizza\_types    pizzas    orders

49 -- List the top 5 most ordered pizza types along with their quantities.  
50 • SELECT  
51     pizza\_types.name, SUM(order\_details.quantity) AS quantity  
52 FROM  
53     pizza\_types  
54     JOIN  
55     pizzas ON pizza\_types.pizza\_type\_id = pizzas.pizza\_type\_id  
56     JOIN  
57     order\_details ON order\_details.pizza\_id = pizzas.pizza\_id  
58 GROUP BY pizza\_types.name  
59 ORDER BY quantity DESC  
60 LIMIT 5;

Result Grid | Filter Rows: [ ] | Export: [ ] | Wrap Cell Content: [ ] | Fetch rows: [ ] | Result Grid | Form Editor | Read Only

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



## -- USE NECESSARY JOIN TO FIND TOTALQUANTITY OF EACH PIZZA CATEGORY ORDERED

```
62
63      -- use necessary join to find totalquantity of each pizza category ordered
64
65 •  select pizza_types.category,
66      count(order_details.quantity) as quantity
67      from pizza_types join pizzas
68      on pizza_types.pizza_type_id=pizzas.pizza_type_id
69      join order_details on order_details.pizza_id=pizzas.pizza_id
70      group by pizza_types.category order by quantity desc;
71
```

The screenshot shows a database query results grid with the following data:

	category	quantity
▶	Classic	14579
	Supreme	11777
	Veggie	11449
	Chicken	10815



LARANA PIZZA

## -- DETERMINE THE DISTRIBUTION OF ORDERS NY HOURS PER DAY

```
76  
77      -- determine the distribution of orders ny hours per day  
78  
79 •   SELECT  
80      HOUR(order_time), COUNT(order_id)  
81      FROM  
82      orders  
83      GROUP BY HOUR(order_time);  
84
```

Result Grid | Filter Rows:  Export: Wrap Cell Content

Result Grid

Form Editor

Field Types

Query Stats

	HOUR(order_time)	COUNT(order_id)
▶	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



LARANA PIZZA

## -- CATEGORY WISE DISTRIBUTION OF PIZZAS

```
-- category wise distribution of pizzas  
select category ,count(name)from pizza_types  
group by category;
```

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

The screenshot shows a database query results window. At the top, there are buttons for 'Result Grid' (highlighted in blue), 'Filter Rows', 'Export' (with icons for CSV and Excel), and 'Wrap Cell Content'. On the right, there is a sidebar with two options: 'Result Grid' (selected) and 'Form Editor'. The main area displays a table with the following data:

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



LARANA PIZZA

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

```
-- group the orders by date and calculate the average
-- number of pizzas order per day

SELECT
    ROUND(AVG(quantity), 0)
FROM
    (
SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

The screenshot shows a database query results window. At the top, there are buttons for 'Result Grid' (selected), 'Filter Rows:', 'Export:', and 'Wrap Cell Content:'. Below the buttons is a table with one row. The first column contains the formula 'ROUND(AVG(quantity), 0)' and the second column contains the value '138'. To the right of the table is a toolbar with icons for 'Result Grid' and 'Form'.

ROUND(AVG(quantity), 0)	138
-------------------------	-----



LARANA PIZZA

## -- DETERMINE THE TOP 3 MOST ORDER PIZZA TYPES BASED ON REVENUE

```
103      -- determine the top 3 most order pizza types based on revenue
104
105 •   SELECT
106     pizza_types.name,
107     SUM(order_details.quantity * pizzas.price) AS revenue
108   FROM
109     pizza_types
110     JOIN
111       pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
112     JOIN
113       order_details ON order_details.pizza_id = pizzas.pizza_id
114   GROUP BY pizza_types.name
115   ORDER BY revenue DESC
116   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



LARANA PIZZA

## -- CALCULATE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TATAL REVENUE

```
118      -- calculate percentage contribution of each pizza type to tatal revenue
119
120 •   SELECT
121     pizza_types.category,
122     ROUND(SUM(order_details.quantity * pizzas.price) /
123           (SELECT ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_sales
124            FROM order_details
125            JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id) * 100, 2) AS revenue
126   FROM
127     pizza_types
128   JOIN
129     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
130   JOIN
131     order_details ON order_details.pizza_id = pizzas.pizza_id
132   GROUP BY
133     pizza_types.category
134   ORDER BY
135     revenue DESC;
136
```

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

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68



LARANA PIZZA

## -- ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

```
138      -- analyze the cumulative revenue generated over time
139 •   SELECT order_date,
140           SUM(revenue) OVER (ORDER BY order_date) AS cum_revenue
141   FROM (
142       SELECT orders.order_date,
143               SUM(order_details.quantity * pizzas.price) AS revenue
144       FROM order_details
145       JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id
146       JOIN orders ON orders.order_id = order_details.order_id -- Adjusted this line
147       GROUP BY orders.order_date
148   ) AS sales;
149
150
```

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

Result Grid

	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
	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	34210.500000000004

Form Editor

Field Types

Query Stats



LARANA PIZZA

-- DETERMINE TOP 3  
MOST ORDER PIZZA  
TYPES  
-- BASED ON REVENUE  
FOR EACH PIZZA  
CATEGORY

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

	name	revenue
▶	The Southwest Chicken Pizza	34705.75
	The Chicken Alfredo Pizza	16900.25
	The Chicken Pesto Pizza	16701.75
	The Greek Pizza	28454.100000000013
	The Italian Capocollo Pizza	25094
	The Napolitana Pizza	24087
	The Big Meat Pizza	22968
	The Pepperoni, Mushroom, and Peppers Pizza	18834.5
	The Pepper Salami Pizza	25529
	The Prosciutto and Arugula Pizza	24193.25
	The Soppressata Pizza	16425.75
	The Calabrese Pizza	15934.25
	The Spinach Supreme Pizza	15277.75
	The Brie Carre Pizza	11588.4999999999

```
153      -- determine top 3 most order pizza types
154      -- based on revenue for each pizza category
155 •   select name,revenue from
156     (SELECT category,
157      name,
158      revenue,
159      RANK() OVER (PARTITION BY category ORDER BY revenue DESC) AS rn
160   FROM (
161     SELECT pizza_types.category,
162           pizza_types.name AS name,
163           SUM(order_details.quantity * pizzas.price) AS revenue
164     FROM pizza_types
165     JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
166     JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
167     GROUP BY pizza_types.category, pizza_types.name
168   ) AS a)AS b
169 where rn>3;
170
171
172
```



LARANA PIZZA

# THANK YOU!



\$6



\$6



\$6



\$6