

PIZZA SALES ANALYSIS USING SQL

UNLOCKING BUSINESS
INSIGHTS FROM RAW DATA

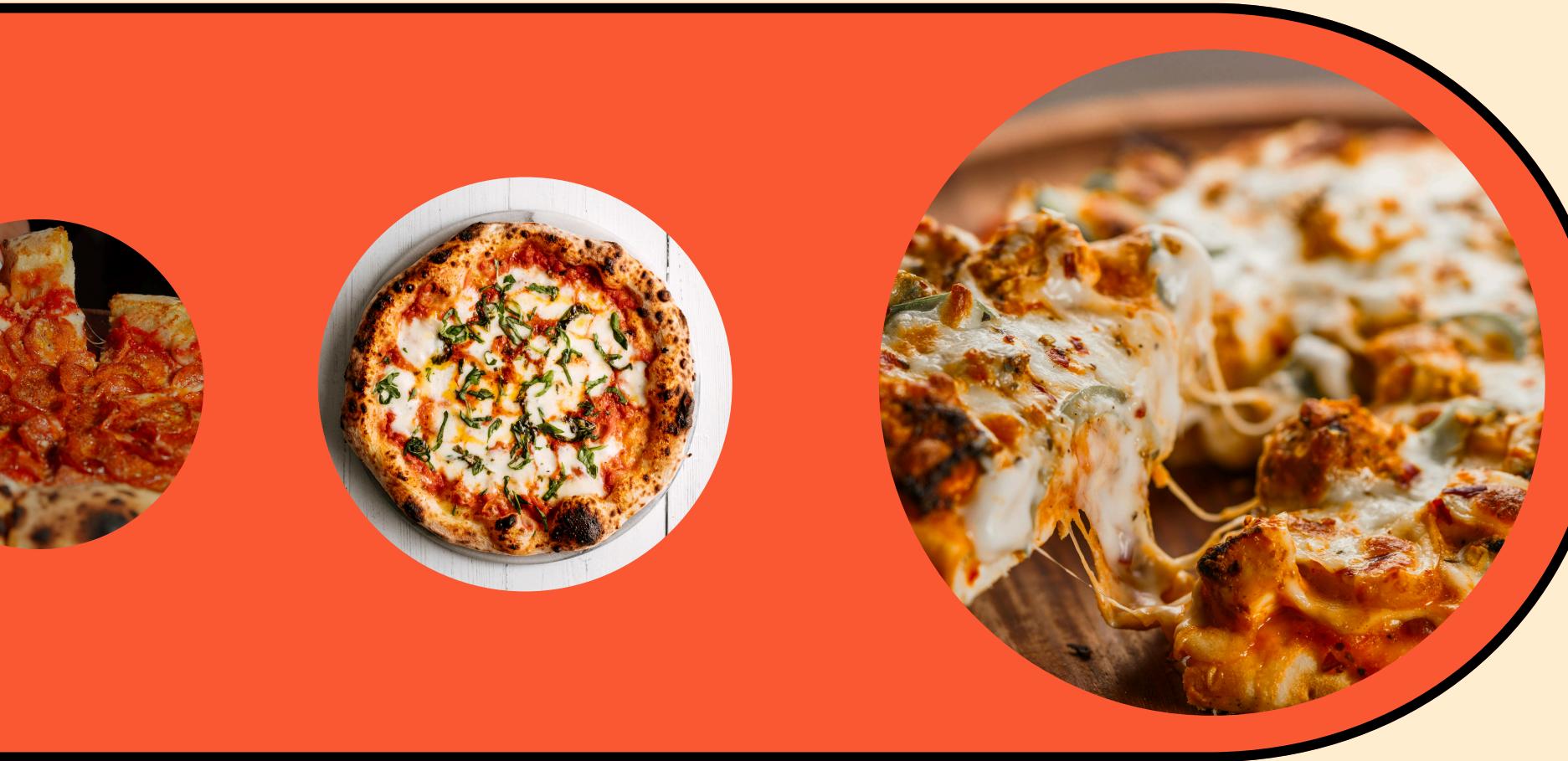
BY - ADHIRAJ GHOSH



WELCOME OUR PIZZA HUT

description here

ANALYZING 1 YEAR OF PIZZA
SALES DATA TO FIND OUT
TOP - SELLING
PRODUCTS,REVENUE,TREND
S AND CUSTOMER PRESENCE



DATABASE ARCHITECTURE

ENVIRONMENT - MySQL/SQL SERVER (MySQL Workbench 8.0 CE).

STRUCTURE - RELATIONAL DATABASE (RDBMS).

TABLES USED - 4 Tables (Orders, Order_details, Pizzas, Pizza Types).

DATA INTEGRITY - Primary keys & Foreign Keys were used to connect tables.

SCALE - Handling 48,000+ records for analysis.



1.RETRIVE THE TOTAL NUMBER OF ORDER PLACED.

```
SELECT count(order_id)as total_orders from orders;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	total_orders			
▶	21350	.		

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
3 •   SELECT
4     ROUND(SUM(order_details.quantity * pizzas.price),
5           2) AS total_sales
6   FROM
7     order_details
8   JOIN
9     pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Result Grid	
	total_sales
▶	34777.75

3. IDENTIFY THE HIGHEST PRICE OF PIZZA.

```
2
3 •   SELECT
4     pizza_types.name, pizzas.price
5   FROM
6     pizza_types|
7       JOIN
8     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9   ORDER BY pizzas.price DESC limit 1;
```

The screenshot shows a MySQL query results window. At the top, there are several buttons: 'Result Grid' (selected), 'Filter Rows:', 'Export:' (with a file icon), 'Wrap Cell Content:' (with a 'W' icon), and 'Fetch rows:' (with a grid icon). Below these buttons is a table with two columns: 'name' and 'price'. A single row is displayed, showing 'The Greek Pizza' in the 'name' column and '35.95' in the 'price' column. There is also a small arrow icon to the left of the first column.

	name	price
▶	The Greek Pizza	35.95

4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
3 • SELECT
4     Execute the selected portion of the script or everything, if there is no selection
5         COUNT(order_details.order_details_id) AS order_count
6     FROM
7         pizzas
8             JOIN
9                 order_details ON pizzas.pizza_id = order_details.pizza_id
10            GROUP BY pizzas.size
11            ORDER BY order_count DESC;
12
```

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

	size	order_count
▶	L	802
	M	664
	S	568
	XL	27

5. LIST THE FIVE MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

```
4 •   SELECT
5       pizza_types.name, SUM(order_details.quantity) AS quantity
6   FROM
7       pizza_types
8           JOIN
9           pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
10          JOIN
11          order_details ON order_details.pizza_id = pizzas.pizza_id
12      GROUP BY pizza_types.name
13      ORDER BY quantity DESC
14      LIMIT 5;
```

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

	name	quantity
▶	The Pepperoni Pizza	123
	The Barbecue Chicken Pizza	100
	The Thai Chicken Pizza	97
	The Classic Deluxe Pizza	96
	The Italian Supreme Pizza	95

6.TOTAL QUANTITY OF EACH PIZZA CATEGORY.

```
5  
4 •   SELECT  
5     pizza_types.category,  
5     SUM(order_details.quantity) AS quantity  
7   FROM  
8     pizza_types  
9       JOIN  
0       pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id  
1       JOIN  
2       order_details ON order_details.pizza_id = pizzas.pizza_id  
3   GROUP BY pizza_types.category  
4   ORDER BY quantity DESC;
```

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

	category	quantity
▶	Classic	615
	Supreme	534
	Veggie	501
	Chicken	452

7.DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```
3 •   SELECT  
4       HOUR(order_time), COUNT(order_id) AS order_count  
5   FROM  
6       orders  
7   GROUP BY (order_time);
```

	HOUR(order_time)	order_count
	12	2
	12	1
	12	2
	12	2
	12	1
	12	1
	12	1
	13	1
	13	3
	13	1
	13	2
	13	2
	13	1
	13	1
	14	1

8. CATEGORY WISE DISTRIBUTION OF PIZZAS.

```
2 •      SELECT
3             category, COUNT(name)
4
5             FROM
6             pizza_types
7
8             GROUP BY category
```

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

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

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

```
2
3 •   SELECT
4     ROUND(AVG(quantity), 0) as avg_pizzas_orders_per_day
5   FROM
6   (SELECT
7     orders.order_date, SUM(order_details.quantity) AS quantity
8   FROM
9     orders
0   JOIN order_details ON orders.order_id = order_details.order_id
1   GROUP BY orders.order_date) AS order_quantity;
```

The screenshot shows a MySQL query editor interface with the following details:

- Toolbar:** Includes buttons for Back, Forward, Refresh, Result Grid (selected), Filter Rows, Export, and Wrap Cell Content.
- Result Grid:** Displays the query results in a table format.
- Table Headers:** The first row contains the column name "avg_pizzas_orders_per_day".
- Data Row:** The second row contains the value "131".

10. TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```
2
3 •   SELECT
4     pizza_types.name,
5       SUM(order_details.quantity * pizzas.price) AS revenue
6   FROM
7     pizza_types
8       *
9     JOIN
10    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
11    JOIN
12    order_details ON order_details.pizza_id = pizzas.pizza_id
13  GROUP BY pizza_types.name
14  ORDER BY revenue DESC
15  LIMIT 3;
```

Result Grid | Filter Rows: Export:

	name	revenue
▶	The Thai Chicken Pizza	1808.75
	The Barbecue Chicken Pizza	1783
	The Italian Supreme Pizza	1684

THANK YOU FOR READING

I'M CONSTANTLY LEARNING AND BUILDING . IF YOU HAVE ANY FEEDBACK OR WANT TO DISCUSS DATA ANALYTICS,I'D LOVE TO CONNECT.

*CURRENTLY LEARNING :- WINDOW FUNCTIONS & POWER BI
NEXT PROJECT COMING VERY SOON.*

LET'S CONNECT GUYS..

~ ADHIRAJ GHOSH