ALIF

PIZZA

SALES ANALYSIS

MySQL®

PROJECT OVERVIEW

Objective

- Alif Pizza Sales Analysis using MySQL
- To transform raw transactional data into actionable business insight
- Identify sales trends, optimize inventory, and understand customer behavior.

TECHNICAL METHODOLOGY

TOOL STAK

- **MySQL** Workbench / SQL Server
- **Data Structure:** Managed a relational database with multiple linked tables.
- **Key Techniques:** Used Complex Joins, Subqueries, and Common Table Expressions (CTEs) to aggregate data.

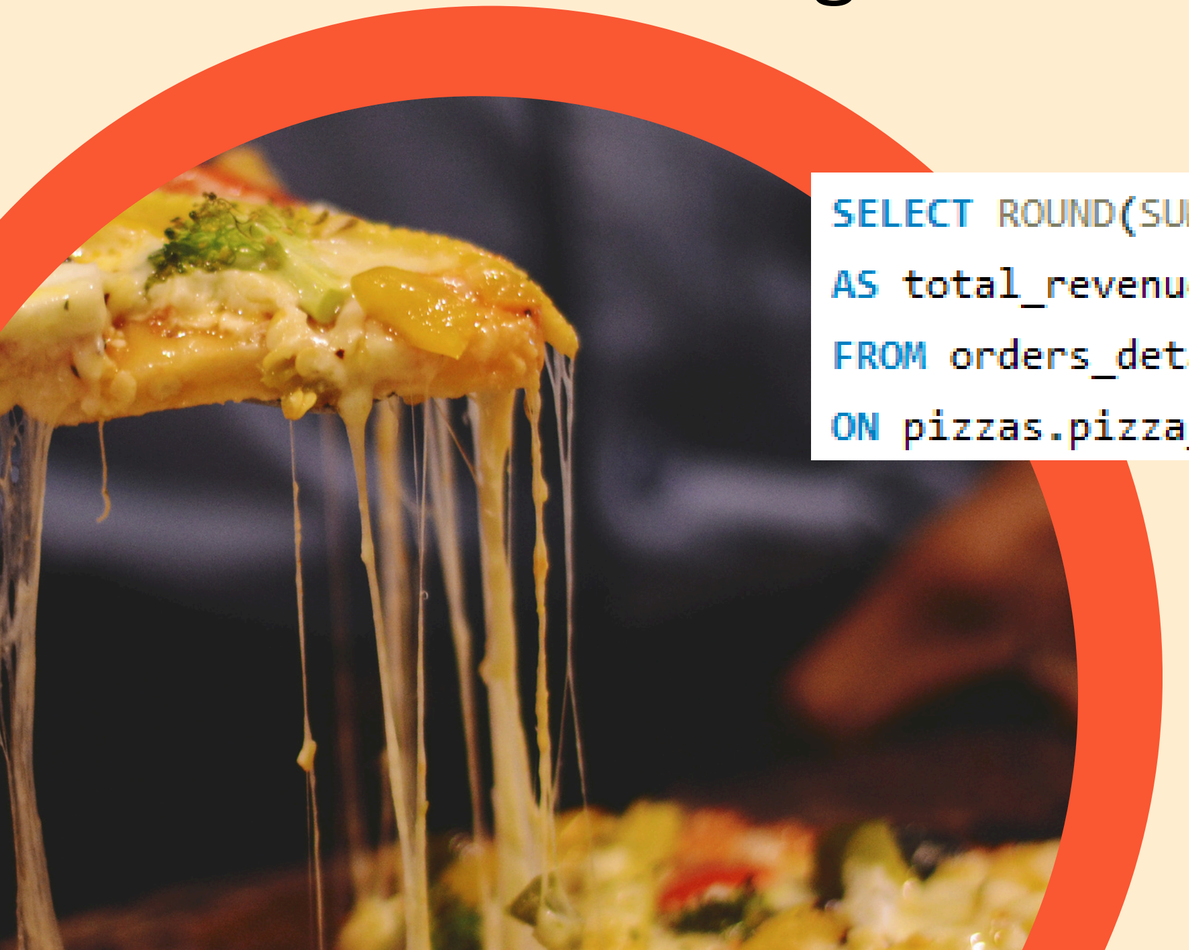
Retrieve the total number of orders placed.

```sql
SELECT count(order_id) FROM orders;
```

Output

count(order_id)
28394

Calculate the total revenue generated from pizza sales.



```sql
SELECT ROUND(SUM(orders_details.quantity*pizzas.price),2)
AS total_revenue
FROM orders_details JOIN pizzas
ON pizzas.pizza_id = orders_details.pizza_id;
```

Identify the highest-priced pizza.

```
SELECT pizzas.price , pizza_types.name , pizza_types.category
FROM pizzas JOIN pizza_types
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY price DESC LIMIT 1;
```

Output

price	name	category
35.95	The Greek Pizza	Classic

Identify the most common pizza size ordered.

```sql
SELECT pizzas.size, COUNT(orders_details.order_details_id)
AS most_order
FROM pizzas JOIN orders_details
ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY most_order DESC
LIMIT 1;
```

List the top 5 most ordered pizza types along with their quantities.

```sql
SELECT pizza_types.name , SUM(orders_details.quantity) AS quantity
FROM pizza_types JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY name ORDER BY quantity DESC LIMIT 5;
```

Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
SELECT pizza_types.category , SUM(orders_details.quantity)
AS total_quantity
FROM pizza_types JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details
ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY category order by total_quantity DESC
```

Determine the distribution of orders by hour of the day.

```sql
SELECT hour(order_time) AS hour, count(order_id)
AS distribution
FROM orders
GROUP BY hour;
```

Join relevant tables to find the category-wise distribution of pizzas

```sql
SELECT category, COUNT(name) AS category
FROM pizza_types GROUP BY category;
```

Group the orders by date and calculate the average number of pizzas ordered per day.

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

Determine the top 3 most ordered pizza types based on revenue.

```
SELECT pizza_types.name , SUM(pizzas.price*orders_details.quantity)
AS revenue
FROM pizza_types JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details
ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY name ORDER BY revenue DESC LIMIT 3;
```

Output

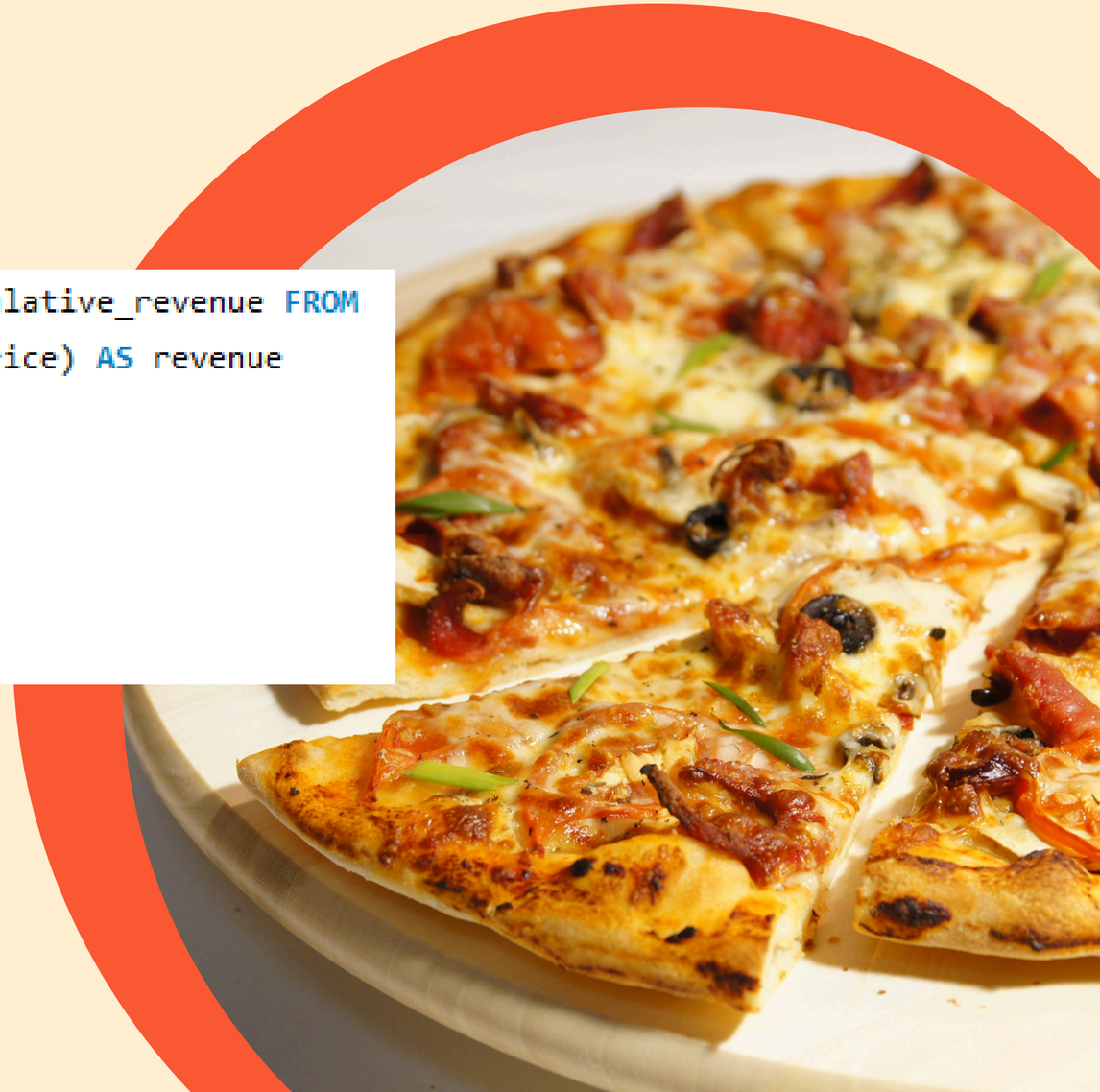
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Calculate the percentage contribution of each pizza type to total revenue.

```sql
SELECT pizza_types.category , ROUND(SUM(pizzas.price*orders_details.quantity)
 / (SELECT SUM(orders_details.quantity*pizzas.price)
FROM orders_details JOIN pizzas
ON orders_details.pizza_id = pizzas.pizza_id)*100 ,2) AS revenue
FROM pizzas JOIN orders_details
ON pizzas.pizza_id = orders_details.pizza_id
JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY category ORDER BY revenue DESC;
```

Analyze the cumulative revenue generated over time

```sql
SELECT order_date, SUM(revenue) OVER(ORDER BY order_date) AS cumulative_revenue FROM
(SELECT orders.order_date , SUM(orders_details.quantity*pizzas.price) AS revenue
FROM orders_details JOIN pizzas
ON orders_details.pizza_id = pizzas.pizza_id
JOIN orders
ON orders.order_id = orders_details.order_id
GROUP BY order_date) AS sales;
```



Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
SELECT name , category , revenue  FROM
(SELECT category,name,revenue, RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS rankk FROM
(SELECT pizza_types.category , pizza_types.name , SUM(pizzas.price*orders_details.quantity) AS revenue
FROM pizza_types JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details
ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY category,name) AS a) AS b
WHERE rankk <=3 ;
```


MySQL™