Pizza Sales Analysis

This presentation delves into an in-depth analysis of pizza sales data. We will explore key insights and patterns revealed through SQL queries, complemented by data visualizations created using Tableau, paving the way for strategic improvements.

Presented By - Ankit Agrawal

content

- Introduction
- Dataset Overview
- Project Objectives
- Problem Analysis
- SQL Queries
- Tableau Overview
- Conclusion
- Future Work
- Acknowledgement

Introduction

In this project, we combine SQL and Tableau to analyze and visualize pizza sales data, enabling stakeholders to uncover actionable insights for strategic decision-making. The integration of these two tools provides a comprehensive view of the data, from extraction and analysis to visualization.

1 SQL: Data Analysis and Querying:

SQL forms the core of data analysis, allowing us to retrieve, aggregate, and filter pizza sales data. It helps analyze sales by category, identify order and revenue trends, and calculate percentage contributions by pizza size and category. Through SQL queries, we gain insights into total pizzas sold, revenue by pizza type, and sales trends, essential for identifying top sellers, peak periods, and areas for improvement.

2 Tableau: Data Visualization and Interpretation:

After processing the data with SQL, Tableau transforms it into interactive, visually compelling dashboards. Tableau helps explore sales patterns and trends, providing insights into hourly sales, weekly performance, and sales by pizza size and category. With two dashboards—general trends and best/worst-sellers—it enables data-driven decisions on product offerings and sales strategies, all presented in an easy-to-understand format.

Dataset Overview

Dataset Name - Pizza Sales Dataset

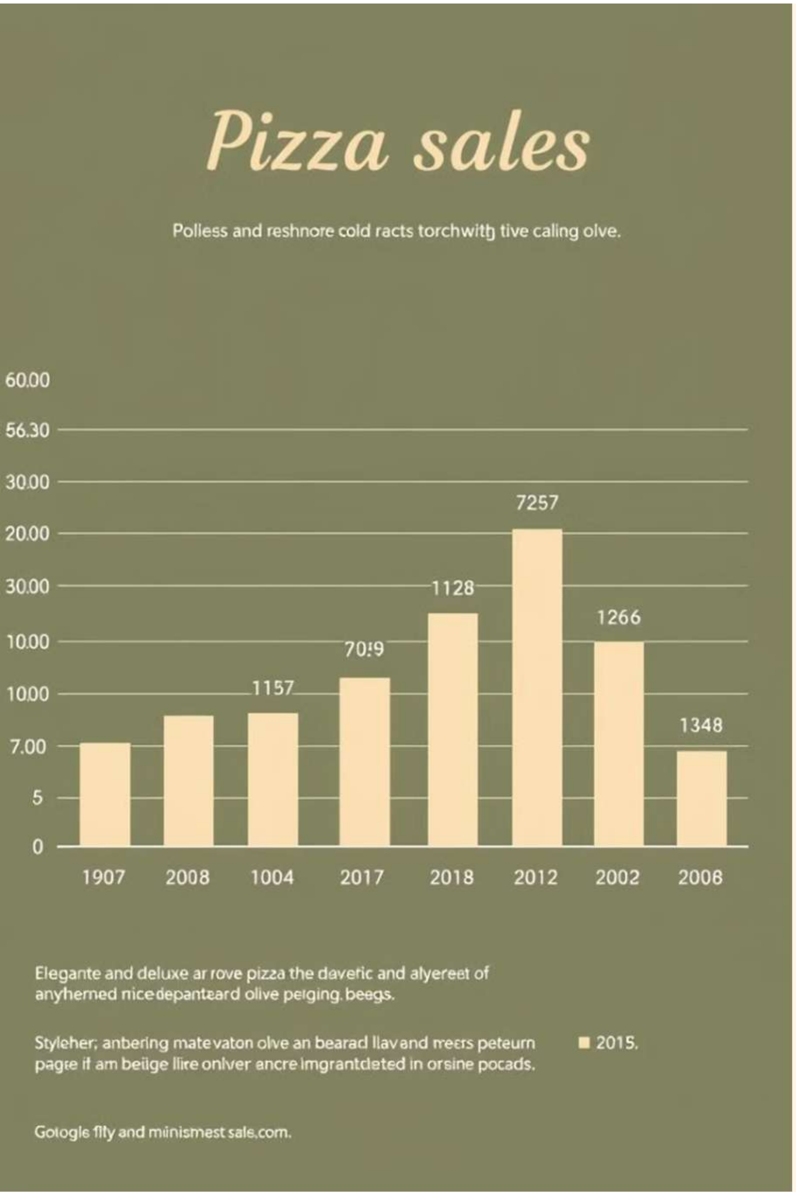
Data Source: The dataset was collected from internet , containing transaction-level details of pizza sales over a specific time period.

Key Features:

1. `order_id`: A unique identifier for each customer order, allowing us to track the number of distinct orders.
2. `pizza_name`: The name of the pizza sold, useful for analyzing the performance of different pizza types.
3. `quantity`: The number of pizzas sold per order, which provides insights into the volume of sales.
4. `total_price`: The total amount paid per order, allowing for revenue analysis.
5. `order_date`: The date the order was placed, critical for time-based trend analysis (e.g., daily, weekly, monthly trends).
6. `order_time`: The time the order was placed, useful for identifying peak hours and periods of high demand.
7. Tableau: This is used for Data Visualization

Data Summary:

- **Size**: The dataset contains **48,620 rows**, representing individual orders.
- **Timeframe**: The data spans a period of [insert duration, i.e. 1 year], allowing for a thorough analysis of seasonal patterns and long-term trends.


Pizza sales
Poliess and reshnore cold racts torchwitђ tive caling olve.
60.00
56.30
30.00
20.00
30.00
10.00
1000
7.00
5
0
7257
1128
1266
70!9
1157
1348
1907
2008
1004
2017
2018
2012
2002
2008
Elegante and deluxe ar rove pizza the daveric and alyereet of anyhemed nicedepanteard olive peiging beegs.
Styleher; anbering matevaton olve an bearad llavand mecrs peteum pagre it am beilge llire onlver encre imgrantdeted in orsine pocads.
■ 2015.
Gotogle flty and minismest sale.com.

Project Objectives

Main Objective: To perform a comprehensive SQL-based analysis and Tableau based Visualization of pizza sales, uncovering key patterns and trends. This includes identifying popular pizza types, understanding customer preferences, and analyzing sales performance to gain valuable insights that can drive business growth.

1 **Sales Performance:**

Analyze overall sales performance and identify areas for improvement.

2 **Performance Insights Segment:**

Analyze which pizzas contribute the most and the least to overall revenue, helping optimize the menu and product offerings.

3 **Product Insights:**

Understand the popularity of different pizza types and toppings to optimize product offerings.

4 **Peak Sales Insights:**

Identify the busiest hours and most profitable days for pizza sales, helping optimize staffing and promotional efforts.

Project Analysis

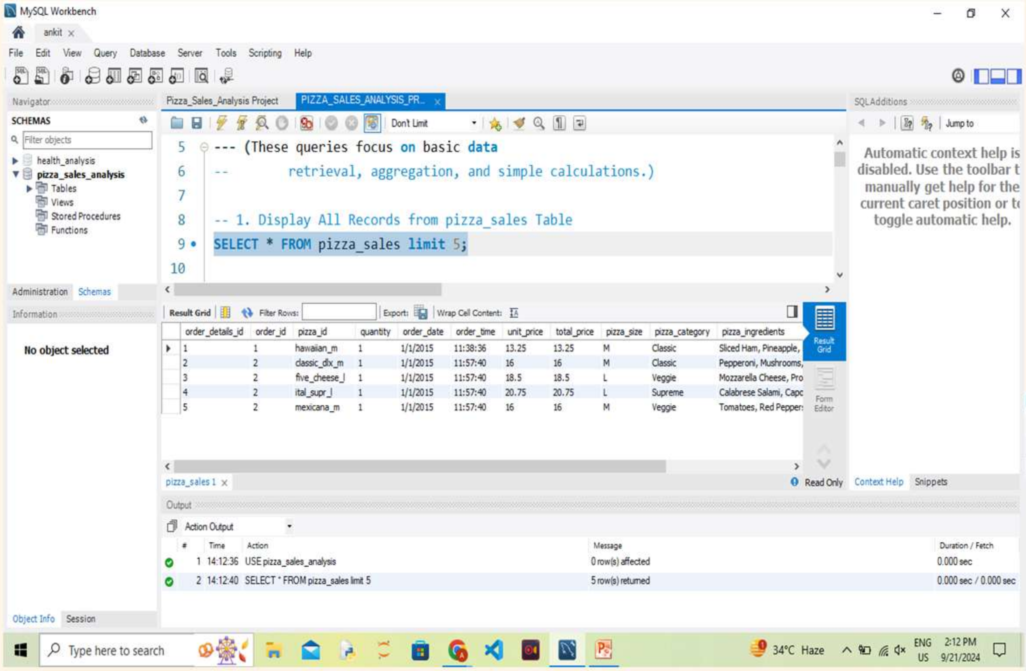
Challenges Addressed:

- **Sales Monitoring**: Identify which pizzas are generating the most and least revenue to focus on top-selling products and address underperforming items.
- **Trend Identification**: Discover daily, weekly, and monthly sales patterns to optimize resource allocation, inventory management, and staffing.
- **Customer Behavior**: Understand customer ordering habits, such as the average pizzas per order and peak ordering times, to improve service efficiency and customer satisfaction.

Opportunities:

- **Marketing**: Develop targeted promotions and special offers to boost sales of underperforming pizzas and attract customer attention.
- **Staffing Optimization**: Adjust staff schedules to ensure sufficient coverage during peak sales hours, improving operational efficiency and customer experience.
- **Promotions**: Plan promotional events on high-traffic days, leveraging sales trends to maximize engagement and increase revenue.

SQL Query Analysis:



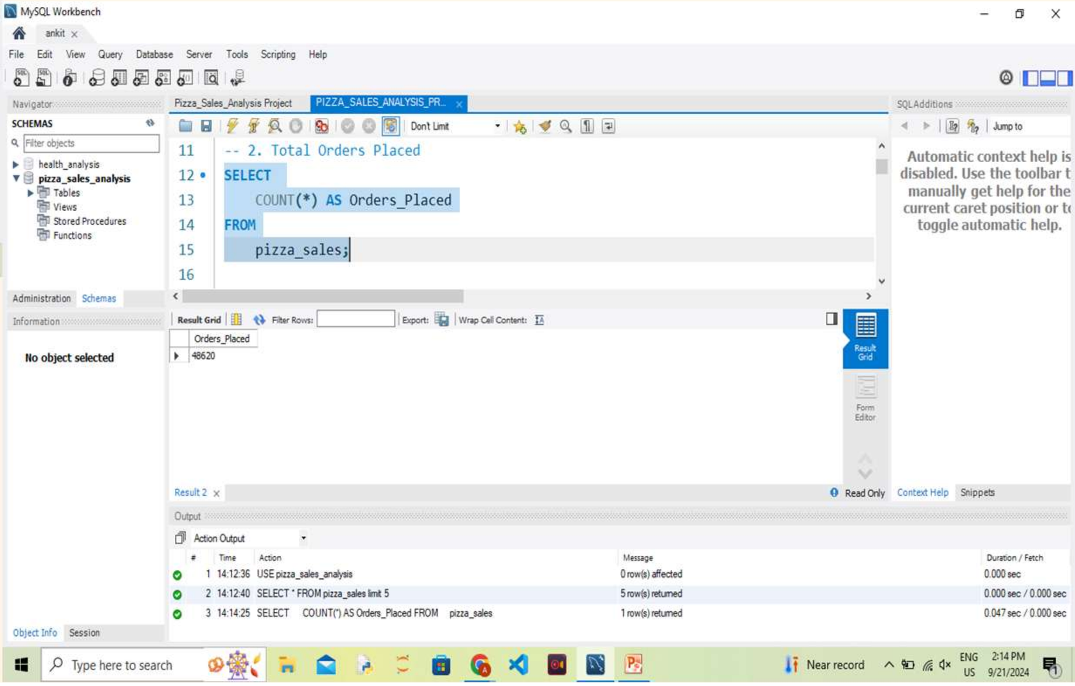
```
--- (These queries focus on basic data
--          retrieval, aggregation, and simple calculations.)

-- 1. Display All Records from pizza_sales Table
SELECT * FROM pizza_sales limit 5;
```

order_details_id	order_id	pizza_id	quantity	order_date	order_time	unit_price	total_price	pizza_size	pizza_category	pizza_ingredients
1	1	hawaiian_m	1	1/1/2015	11:38:36	13.25	13.25	M	Classic	Sliced Ham, Pineapple,
2	2	classic_dlx_m	1	1/1/2015	11:57:40	16	16	M	Classic	Pepperoni, Mushrooms,
3	2	five_cheese_l	1	1/1/2015	11:57:40	18.5	18.5	L	Veggie	Mozzarella Cheese, Pro
4	2	ital_supr_l	1	1/1/2015	11:57:40	20.75	20.75	L	Supreme	Calabrese Salami, Capc
5	2	mexicana_m	1	1/1/2015	11:57:40	16	16	M	Veggie	Tomatoes, Red Pepper:

```
#  Time      Action                               Message              Duration / Fetch
1  14:12:36  USE pizza_sales_analysis             0 row(s) affected    0.000 sec
2  14:12:40  SELECT * FROM pizza_sales limit 5    5 row(s) returned    0.000 sec / 0.000 sec
```

1.This query retrieves all records from the pizza_sales table, displaying a comprehensive view of the dataset. It allows us to examine all transactions and understand the structure of the table, including the various attributes related to each pizza order.

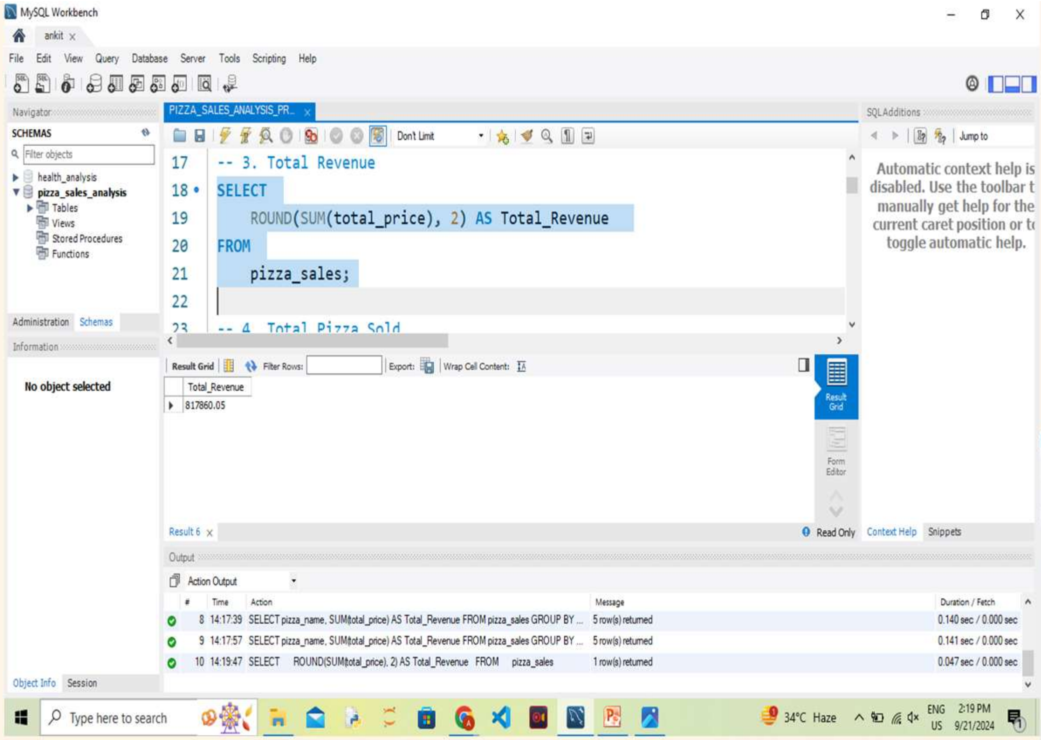
2. This query calculates the total number of orders placed. By using the COUNT(*) function, we capture the total number of rows in the dataset, reflecting how many individual transactions or sales were made.



```
-- 2. Total Orders Placed
SELECT
    COUNT(*) AS Orders_Placed
FROM
    pizza_sales;
```

Orders_Placed
48620

```
#  Time      Action                                             Message              Duration / Fetch
1  14:12:36  USE pizza_sales_analysis                           0 row(s) affected    0.000 sec
2  14:12:40  SELECT * FROM pizza_sales limit 5                  5 row(s) returned    0.000 sec / 0.000 sec
3  14:14:25  SELECT    COUNT(*) AS Orders_Placed FROM    pizza_sales    1 row(s) returned    0.047 sec / 0.000 sec
```



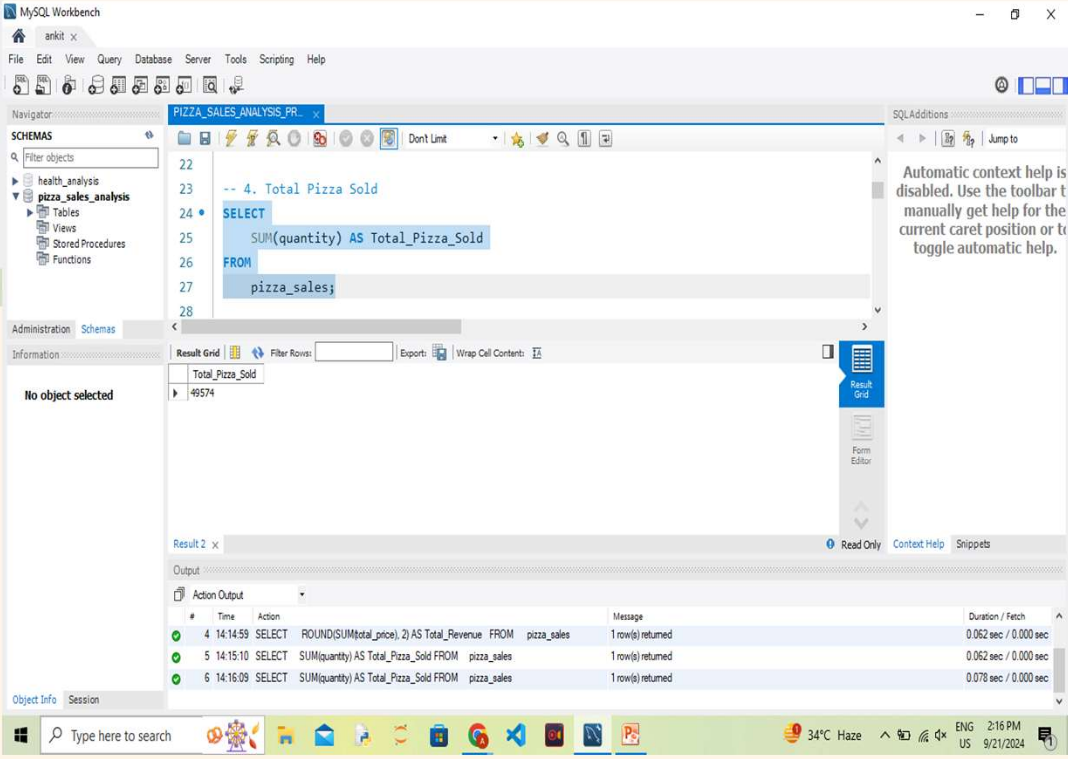
```sql
-- 3. Total Revenue
SELECT
    ROUND(SUM(total_price), 2) AS Total_Revenue
FROM
    pizza_sales;

-- 4. Total Pizza Sold
```

Total_Revenue
817860.05

```
#   Time      Action                                                                         Message            Duration / Fetch
8   14:17:39  SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales GROUP BY ...  5 row(s) returned  0.140 sec / 0.000 sec
9   14:17:57  SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales GROUP BY ...  5 row(s) returned  0.141 sec / 0.000 sec
10  14:19:47  SELECT    ROUND(SUM(total_price), 2) AS Total_Revenue   FROM    pizza_sales         1 row(s) returned  0.047 sec / 0.000 sec
```

3.Here, we calculate the total revenue generated from pizza sales. The SUM(total_price) function adds up all the values from the total_price column, and the result is rounded to two decimal places for better readability.

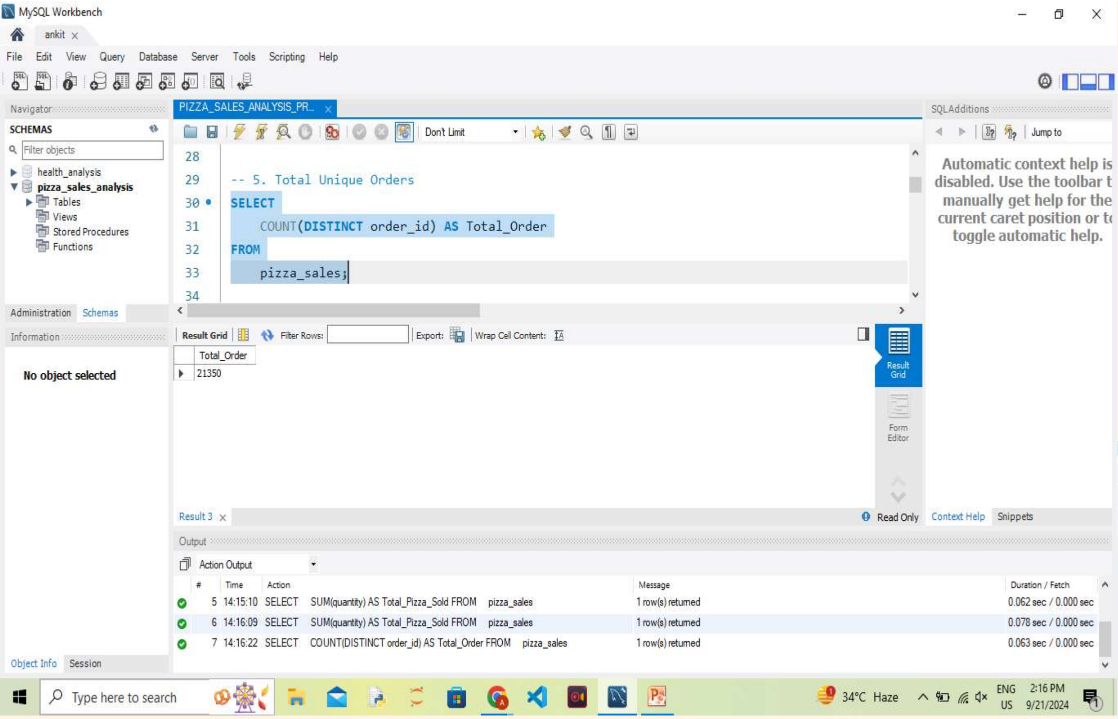
4.This query calculates the total number of pizzas sold by summing the quantity column. It provides a clear picture of the overall volume of pizzas sold across all orders.



```sql
-- 4. Total Pizza Sold
SELECT
    SUM(quantity) AS Total_Pizza_Sold
FROM
    pizza_sales;
```

Total_Pizza_Sold
49574

```
#  Time      Action                                                            Message            Duration / Fetch
4  14:14:59  SELECT    ROUND(SUM(total_price), 2) AS Total_Revenue   FROM    pizza_sales  1 row(s) returned  0.062 sec / 0.000 sec
5  14:15:10  SELECT    SUM(quantity) AS Total_Pizza_Sold FROM    pizza_sales              1 row(s) returned  0.062 sec / 0.000 sec
6  14:16:09  SELECT    SUM(quantity) AS Total_Pizza_Sold FROM    pizza_sales              1 row(s) returned  0.078 sec / 0.000 sec
```



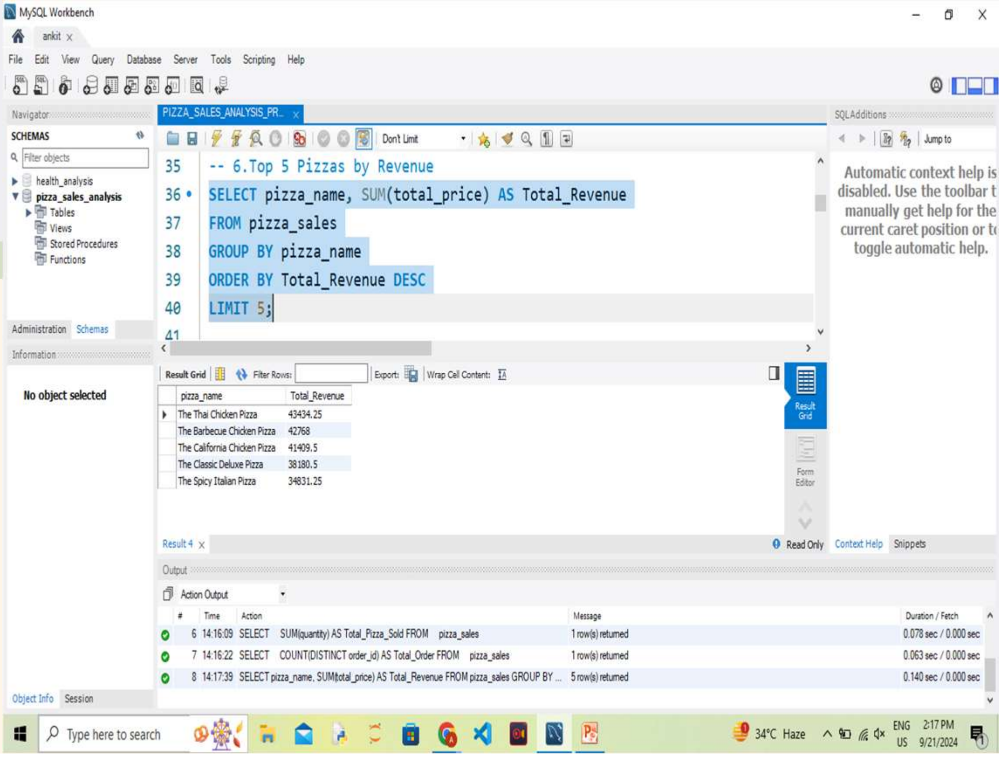
```
-- 5. Total Unique Orders
SELECT
    COUNT(DISTINCT order_id) AS Total_Order
FROM
    pizza_sales;
```

Total_Order
21350

```
#  Time      Action                                                          Message          Duration / Fetch
5  14:15:10  SELECT   SUM(quantity) AS Total_Pizza_Sold FROM   pizza_sales    1 row(s) returned  0.062 sec / 0.000 sec
6  14:16:09  SELECT   SUM(quantity) AS Total_Pizza_Sold FROM   pizza_sales    1 row(s) returned  0.078 sec / 0.000 sec
7  14:16:22  SELECT   COUNT(DISTINCT order_id) AS Total_Order FROM   pizza_sales   1 row(s) returned  0.063 sec / 0.000 sec
```

5.In this query, we determine the total number of unique orders placed using the COUNT(DISTINCT order_id) function. This ensures we only count distinct orders, even if an order contains multiple pizzas.

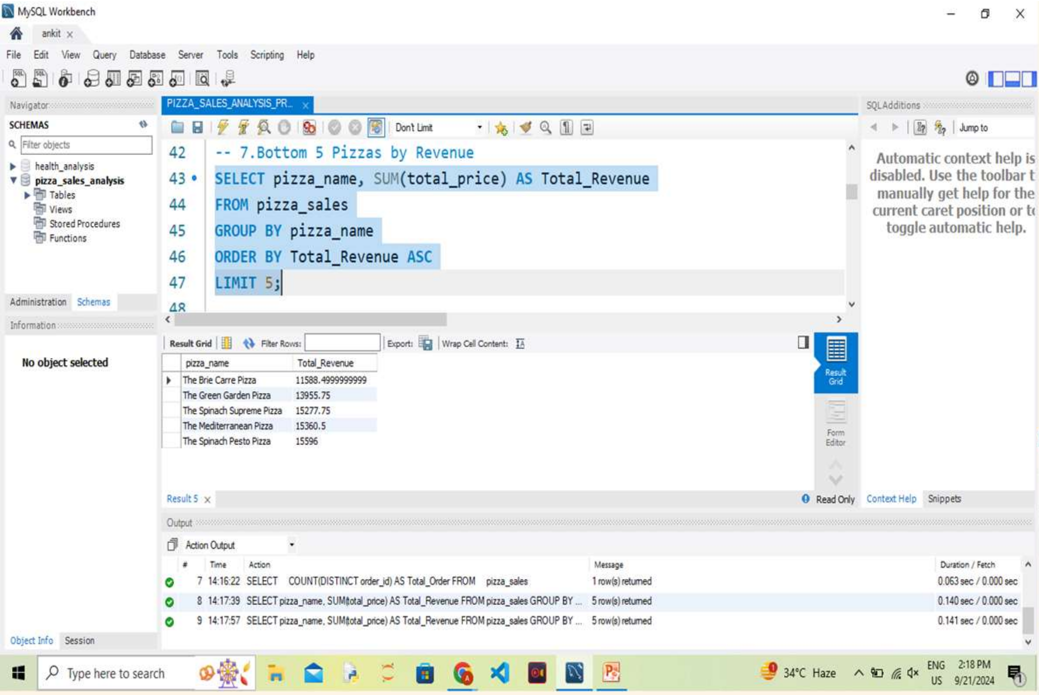
6.This query identifies the top 5 pizzas that generated the most revenue. By grouping the data by pizza name and summing up the total price for each pizza, we can sort the results in descending order to highlight the highest earners.



```
-- 6.Top 5 Pizzas by Revenue
SELECT pizza_name, SUM(total_price) AS Total_Revenue
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_Revenue DESC
LIMIT 5;
```

pizza_name	Total_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 Spicy Italian Pizza	34831.25

```
#  Time      Action                                                                          Message          Duration / Fetch
6  14:16:09  SELECT   SUM(quantity) AS Total_Pizza_Sold FROM   pizza_sales                    1 row(s) returned  0.078 sec / 0.000 sec
7  14:16:22  SELECT   COUNT(DISTINCT order_id) AS Total_Order FROM   pizza_sales              1 row(s) returned  0.063 sec / 0.000 sec
8  14:17:39  SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales GROUP BY ...  5 row(s) returned  0.140 sec / 0.000 sec
```



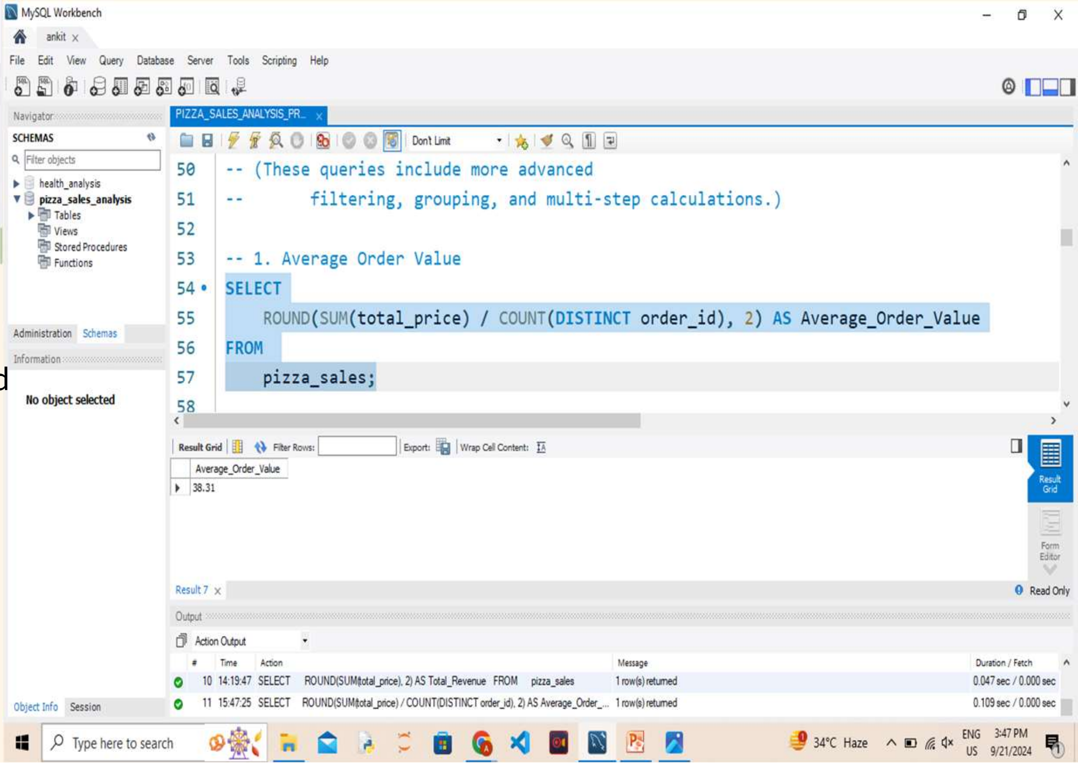
```sql
-- 7.Bottom 5 Pizzas by Revenue
SELECT pizza_name, SUM(total_price) AS Total_Revenue
FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_Revenue ASC
LIMIT 5;
```

pizza_name	Total_Revenue
The Brie Carre Pizza	11588.4999999999
The Green Garden Pizza	13955.75
The Spinach Supreme Pizza	15277.75
The Mediterranean Pizza	15360.5
The Spinach Pesto Pizza	15596

```
#  Time      Action                                                                      Message            Duration / Fetch
7  14:16:22  SELECT  COUNT(DISTINCT order_id) AS Total_Order FROM  pizza_sales          1 row(s) returned  0.063 sec / 0.000 sec
8  14:17:39  SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales GROUP BY ...  5 row(s) returned  0.140 sec / 0.000 sec
9  14:17:57  SELECT pizza_name, SUM(total_price) AS Total_Revenue FROM pizza_sales GROUP BY ...  5 row(s) returned  0.141 sec / 0.000 sec
```

7.Similar to the previous query, this one identifies the bottom 5 pizzas in terms of revenue. It helps us understand which pizzas contributed the least to overall sales, potentially informing decisions on menu adjustments or promotions.

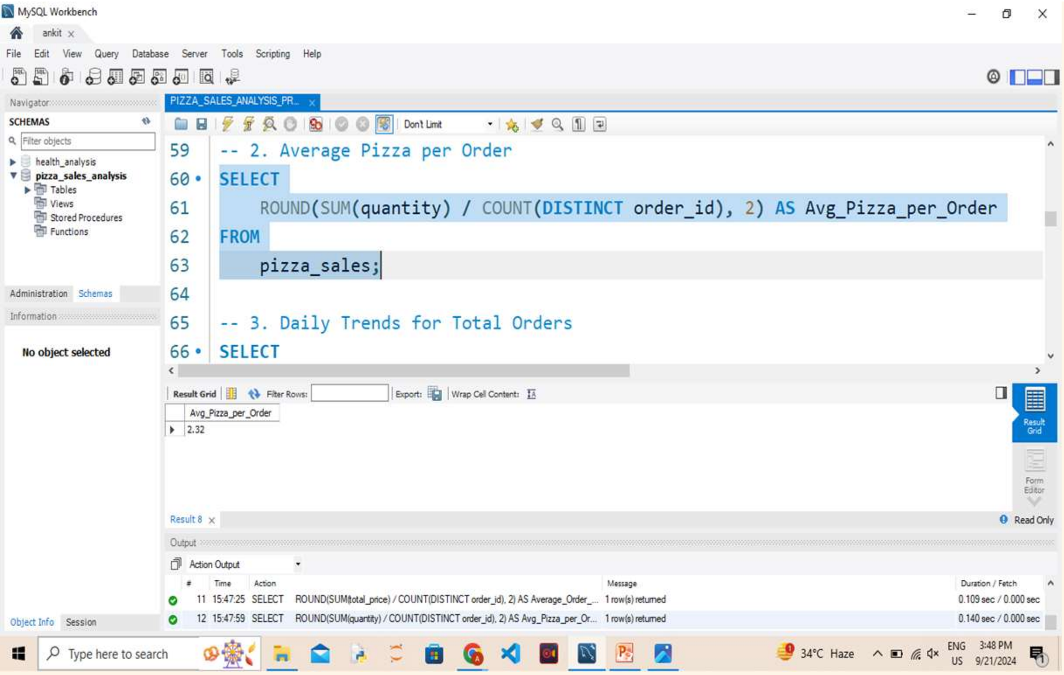
8. This query calculates the **Average Order Value (AOV)**, which is derived by dividing the total revenue by the number of unique orders. It gives an insight into how much revenue, on average, each order generates. Understanding AOV helps in strategizing pricing, discounts, or bundling promotions to increase the value of each order.



```sql
-- (These queries include more advanced
--       filtering, grouping, and multi-step calculations.)

-- 1. Average Order Value
SELECT
    ROUND(SUM(total_price) / COUNT(DISTINCT order_id), 2) AS Average_Order_Value
FROM
    pizza_sales;
```

Average_Order_Value
38.31

```
#   Time      Action                                                                  Message            Duration / Fetch
10  14:19:47  SELECT  ROUND(SUM(total_price), 2) AS Total_Revenue  FROM  pizza_sales  1 row(s) returned  0.047 sec / 0.000 sec
11  15:47:25  SELECT  ROUND(SUM(total_price) / COUNT(DISTINCT order_id), 2) AS Average_Order_...  1 row(s) returned  0.109 sec / 0.000 sec
```



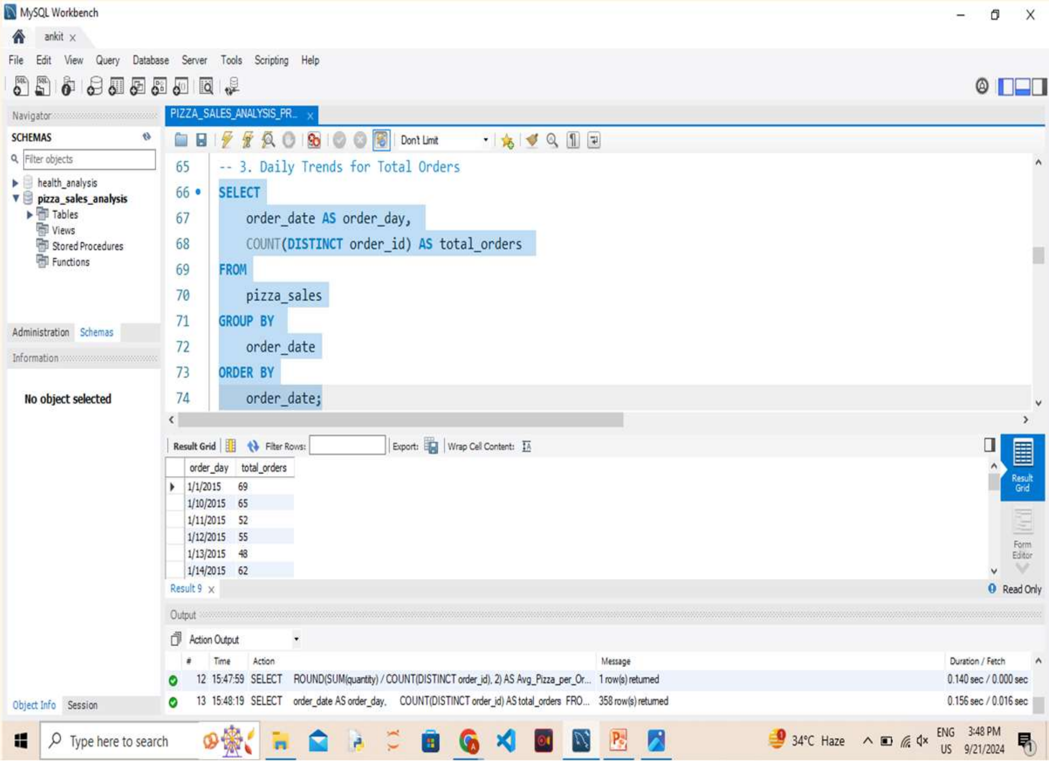
```
-- 2. Average Pizza per Order
SELECT
    ROUND(SUM(quantity) / COUNT(DISTINCT order_id), 2) AS Avg_Pizza_per_Order
FROM
    pizza_sales;

-- 3. Daily Trends for Total Orders
SELECT
```

	Avg_Pizza_per_Order
▶	2.32

```
   #   Time      Action   Message                                                                         Duration / Fetch
✓  11  15:47:25  SELECT   ROUND(SUM(total_price) / COUNT(DISTINCT order_id), 2) AS Average_Order_...  1 row(s) returned   0.109 sec / 0.000 sec
✓  12  15:47:59  SELECT   ROUND(SUM(quantity) / COUNT(DISTINCT order_id), 2) AS Avg_Pizza_per_Or...   1 row(s) returned   0.140 sec / 0.000 sec
```

9. This query calculates the **Average Pizza per Order** by dividing the total quantity of pizzas sold by the number of unique orders. This metric reveals customer behavior, such as whether they tend to order a single pizza or multiple pizzas. It can also guide upsell strategies.

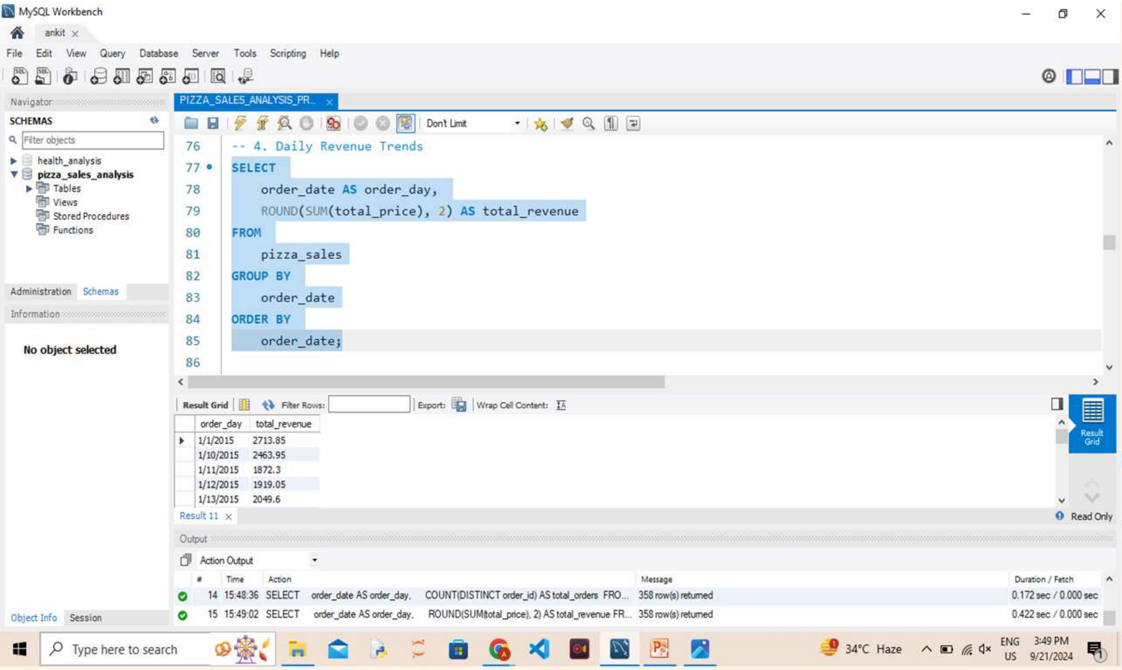
10. The query tracks **Daily Trends** by counting the total number of orders placed each day. This time-based analysis highlights patterns in customer ordering behavior. Analyzing daily order trends can help optimize staffing, inventory, and marketing campaigns around peak sales periods.



```
-- 3. Daily Trends for Total Orders
SELECT
    order_date AS order_day,
    COUNT(DISTINCT order_id) AS total_orders
FROM
    pizza_sales
GROUP BY
    order_date
ORDER BY
    order_date;
```

	order_day	total_orders
▶	1/1/2015	69
	1/10/2015	65
	1/11/2015	52
	1/12/2015	55
	1/13/2015	48
	1/14/2015	62

```
   #   Time      Action   Message                                                                         Duration / Fetch
✓  12  15:47:59  SELECT   ROUND(SUM(quantity) / COUNT(DISTINCT order_id), 2) AS Avg_Pizza_per_Or...   1 row(s) returned     0.140 sec / 0.000 sec
✓  13  15:48:19  SELECT   order_date AS order_day,    COUNT(DISTINCT order_id) AS total_orders  FRO...  358 row(s) returned   0.156 sec / 0.016 sec
```



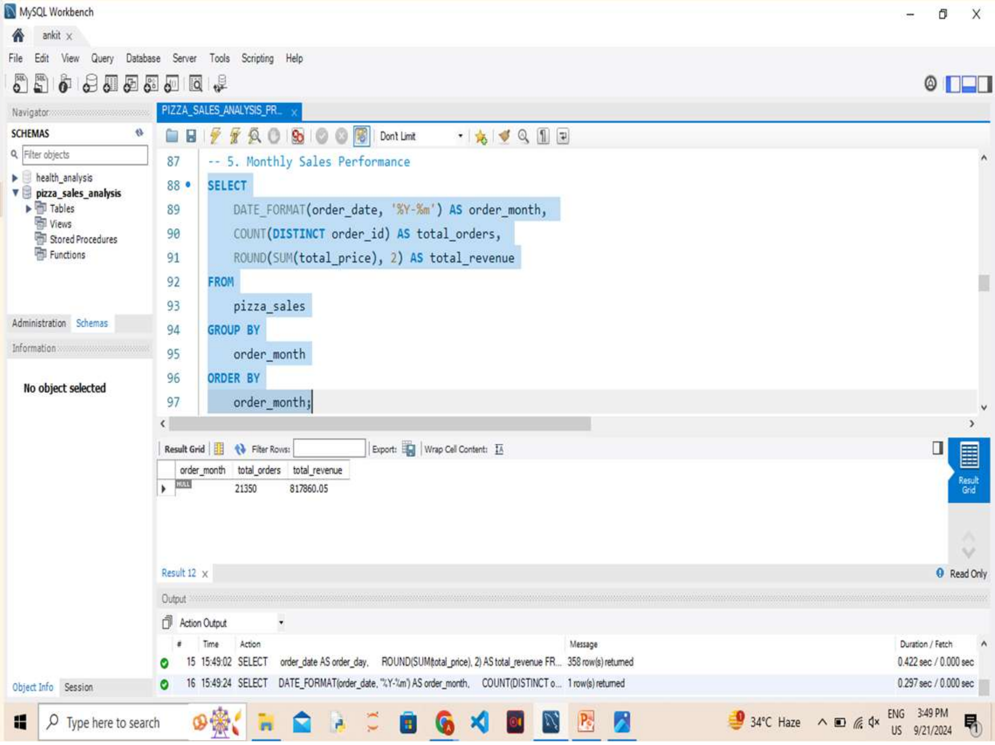
```sql
-- 4. Daily Revenue Trends
SELECT
    order_date AS order_day,
    ROUND(SUM(total_price), 2) AS total_revenue
FROM
    pizza_sales
GROUP BY
    order_date
ORDER BY
    order_date;
```

order_day	total_revenue
1/1/2015	2713.85
1/10/2015	2463.95
1/11/2015	1872.3
1/12/2015	1919.05
1/13/2015	2049.6

```
#   Time      Action                                                                        Message              Duration / Fetch
14  15:48:36  SELECT  order_date AS order_day,   COUNT(DISTINCT order_id) AS total_orders  FRO...  358 row(s) returned  0.172 sec / 0.000 sec
15  15:49:02  SELECT  order_date AS order_day,   ROUND(SUM(total_price), 2) AS total_revenue FR...  358 row(s) returned  0.422 sec / 0.000 sec
```

11. This query calculates **Daily Revenue Trends** by summing the total revenue for each day. Monitoring revenue on a daily basis is critical for detecting fluctuations in sales, understanding the impact of marketing efforts, and adjusting strategies as needed.

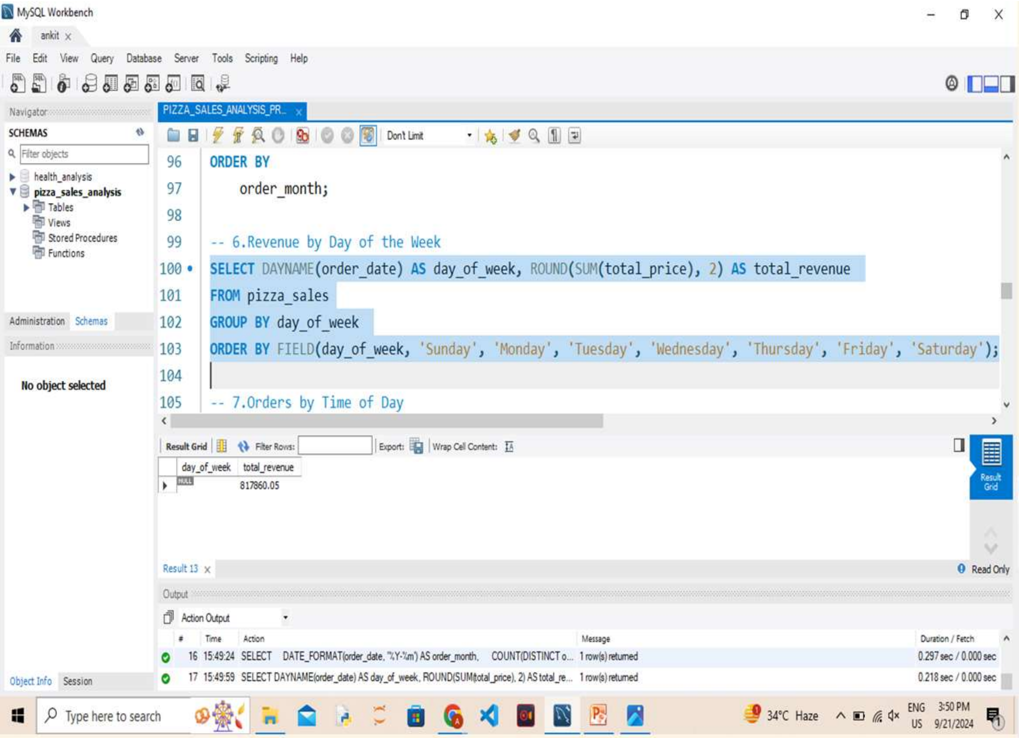
12.This query groups orders and revenue by month, summarizing **Monthly Sales Performance**. It provides a broader view of trends over time. Monthly analysis is important for evaluating the performance of long-term marketing campaigns and promotions, as well as seasonal changes in customer demand.



```sql
-- 5. Monthly Sales Performance
SELECT
    DATE_FORMAT(order_date, '%Y-%m') AS order_month,
    COUNT(DISTINCT order_id) AS total_orders,
    ROUND(SUM(total_price), 2) AS total_revenue
FROM
    pizza_sales
GROUP BY
    order_month
ORDER BY
    order_month;
```

order_month	total_orders	total_revenue
NULL	21350	817860.05

```
#   Time      Action                                                                             Message              Duration / Fetch
15  15:49:02  SELECT  order_date AS order_day,   ROUND(SUM(total_price), 2) AS total_revenue FR...   358 row(s) returned  0.422 sec / 0.000 sec
16  15:49:24  SELECT  DATE_FORMAT(order_date, '%Y-%m') AS order_month,   COUNT(DISTINCT o...        1 row(s) returned    0.297 sec / 0.000 sec
```



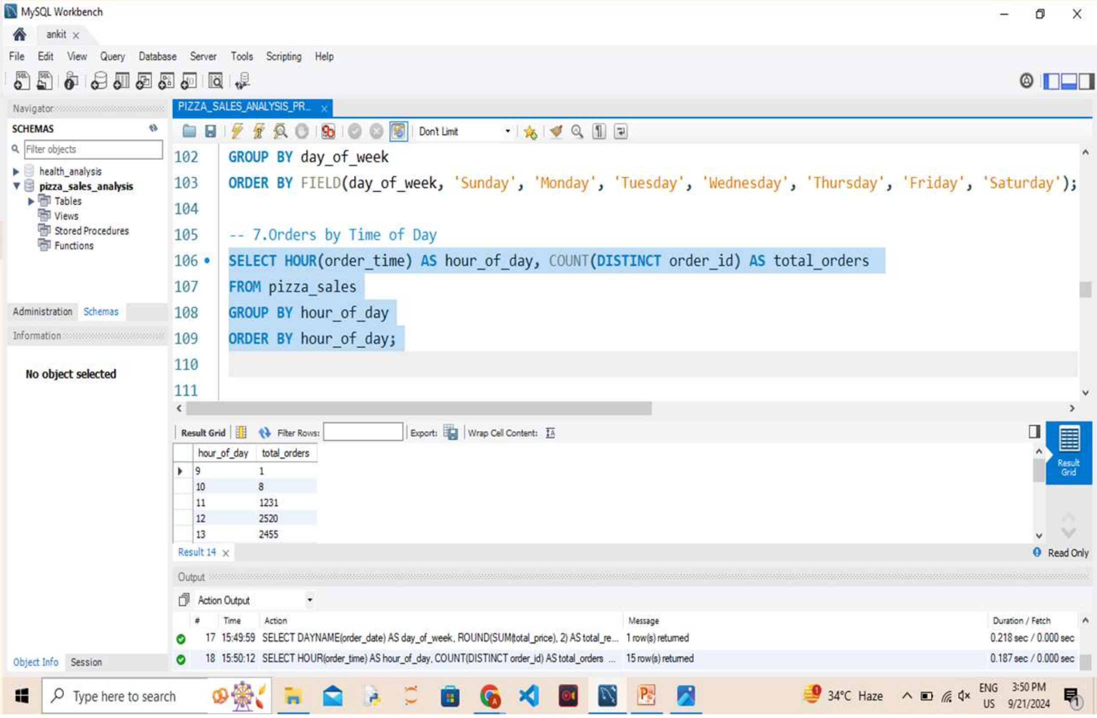
```sql
ORDER BY
    order_month;

-- 6.Revenue by Day of the Week
SELECT DAYNAME(order_date) AS day_of_week, ROUND(SUM(total_price), 2) AS total_revenue
FROM pizza_sales
GROUP BY day_of_week
ORDER BY FIELD(day_of_week, 'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday');

-- 7.Orders by Time of Day
```

day_of_week	total_revenue
NULL	817860.05

```
#   Time      Action                                                                        Message            Duration / Fetch
16  15:49:24  SELECT    DATE_FORMAT(order_date, '%Y-%m') AS order_month,     COUNT(DISTINCT o...  1 row(s) returned  0.297 sec / 0.000 sec
17  15:49:59  SELECT DAYNAME(order_date) AS day_of_week, ROUND(SUM(total_price), 2) AS total_re...  1 row(s) returned  0.218 sec / 0.000 sec
```

13. This query shows the total revenue generated for each day of the week, helping identify which days bring in the most revenue. By understanding **which days are most profitable**, businesses can plan promotions or special offers on slower days to boost sales and revenue.

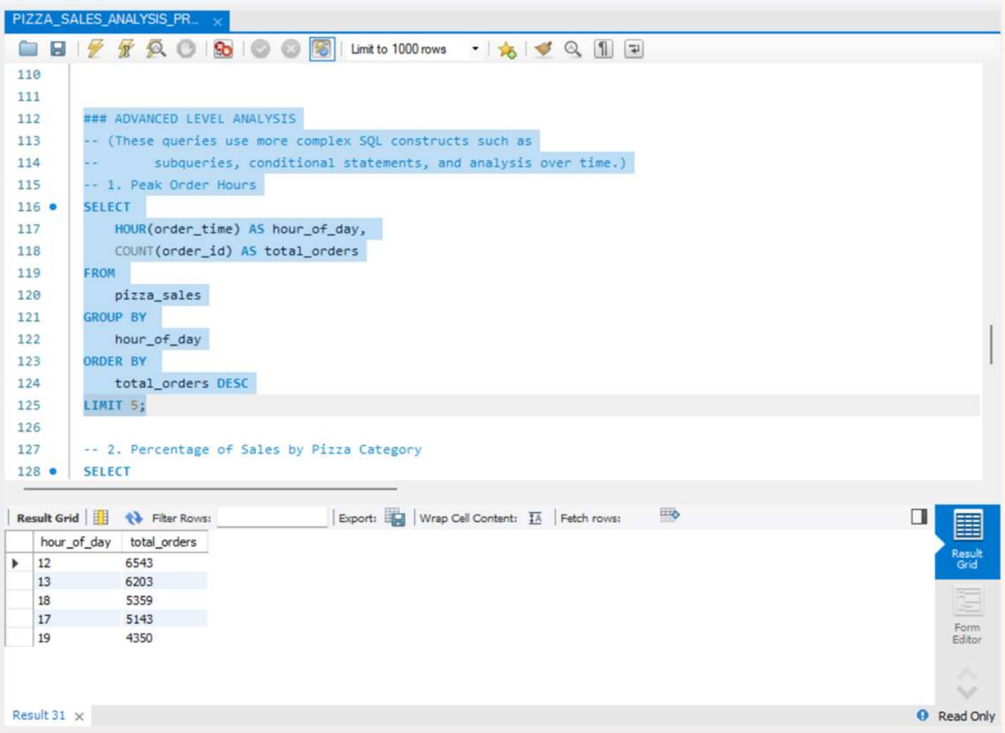
14. This query analyzes orders based on the time of day, revealing the hours during which most orders are placed. Knowing **peak order times** is essential for improving operations, such as adjusting staffing schedules, optimizing delivery times, and running time-sensitive promotions.



```sql
GROUP BY day_of_week
ORDER BY FIELD(day_of_week, 'Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday');

-- 7.Orders by Time of Day
SELECT HOUR(order_time) AS hour_of_day, COUNT(DISTINCT order_id) AS total_orders
FROM pizza_sales
GROUP BY hour_of_day
ORDER BY hour_of_day;
```

hour_of_day	total_orders
9	1
10	8
11	1231
12	2520
13	2455

```
#   Time      Action                                                                        Message             Duration / Fetch
17  15:49:59  SELECT DAYNAME(order_date) AS day_of_week, ROUND(SUM(total_price), 2) AS total_re...  1 row(s) returned   0.218 sec / 0.000 sec
18  15:50:12  SELECT HOUR(order_time) AS hour_of_day, COUNT(DISTINCT order_id) AS total_orders ...  15 row(s) returned  0.187 sec / 0.000 sec
```

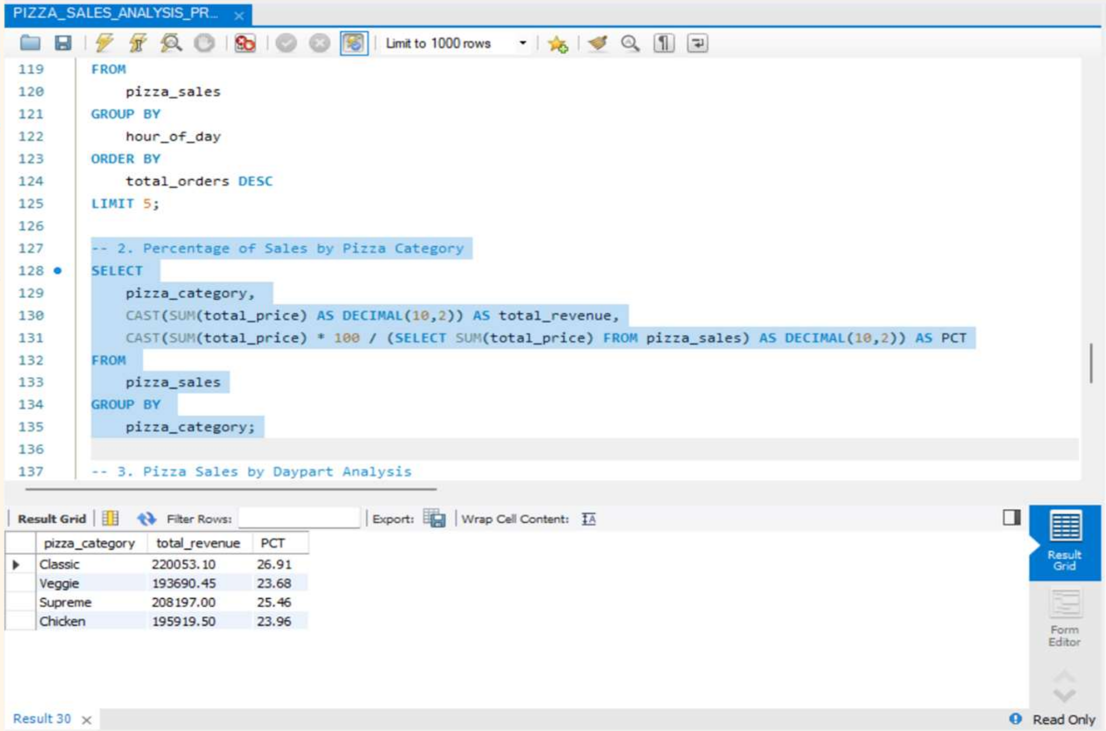


```sql
### ADVANCED LEVEL ANALYSIS
-- (These queries use more complex SQL constructs such as
--       subqueries, conditional statements, and analysis over time.)
-- 1. Peak Order Hours
SELECT
    HOUR(order_time) AS hour_of_day,
    COUNT(order_id) AS total_orders
FROM
    pizza_sales
GROUP BY
    hour_of_day
ORDER BY
    total_orders DESC
LIMIT 5;

-- 2. Percentage of Sales by Pizza Category
SELECT
```

hour_of_day	total_orders
12	6543
13	6203
18	5359
17	5143
19	4350

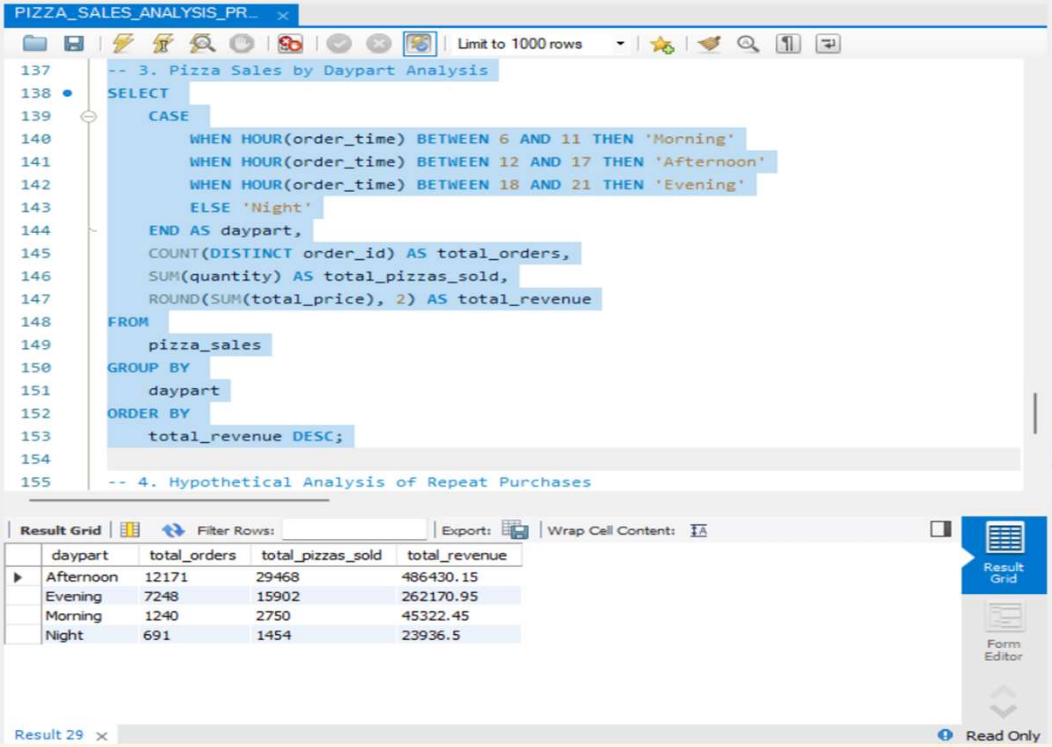
15.This query identifies the five busiest hours for pizza orders by analyzing order times and counting the number of orders during each hour. The results help in understanding customer demand patterns, enabling better resource allocation and staff scheduling during peak hours.

16. The query calculates the total revenue for each pizza category and determines its percentage share of overall sales. It provides insights into customer preferences, helping prioritize high-performing categories and refine marketing strategies for underperforming ones.



```sql
FROM
    pizza_sales
GROUP BY
    hour_of_day
ORDER BY
    total_orders DESC
LIMIT 5;

-- 2. Percentage of Sales by Pizza Category
SELECT
    pizza_category,
    CAST(SUM(total_price) AS DECIMAL(10,2)) AS total_revenue,
    CAST(SUM(total_price) * 100 / (SELECT SUM(total_price) FROM pizza_sales) AS DECIMAL(10,2)) AS PCT
FROM
    pizza_sales
GROUP BY
    pizza_category;

-- 3. Pizza Sales by Daypart Analysis
```

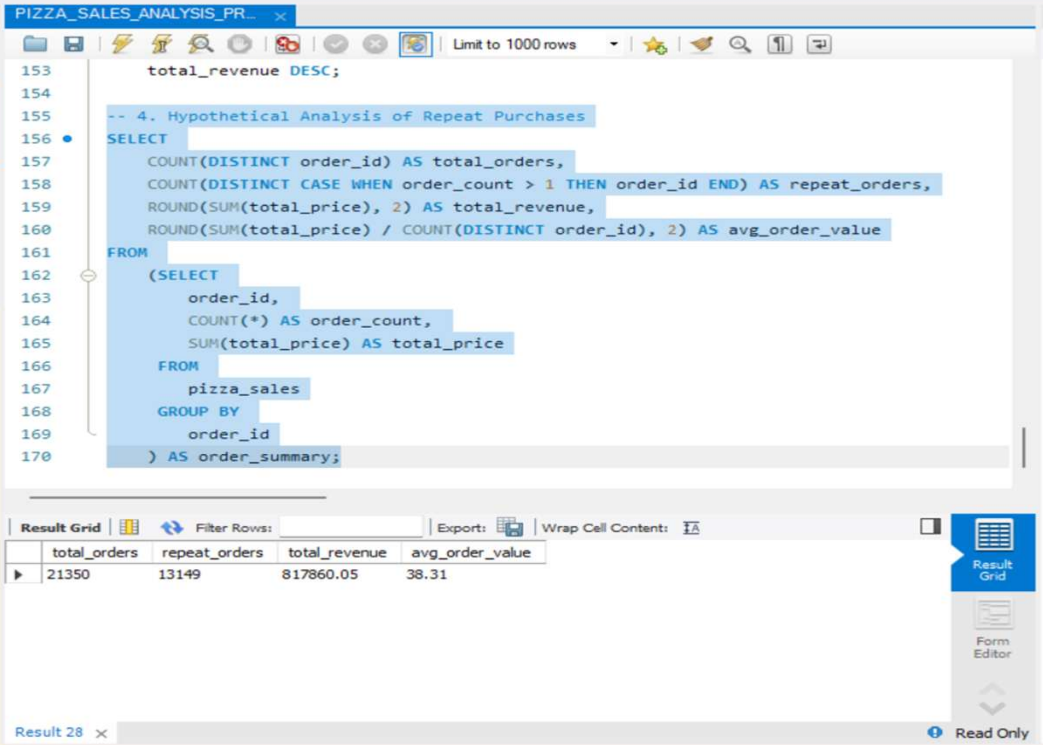
pizza_category	total_revenue	PCT
Classic	220053.10	26.91
Veggie	193690.45	23.68
Supreme	208197.00	25.46
Chicken	195919.50	23.96



```sql
-- 3. Pizza Sales by Daypart Analysis
SELECT
    CASE
        WHEN HOUR(order_time) BETWEEN 6 AND 11 THEN 'Morning'
        WHEN HOUR(order_time) BETWEEN 12 AND 17 THEN 'Afternoon'
        WHEN HOUR(order_time) BETWEEN 18 AND 21 THEN 'Evening'
        ELSE 'Night'
    END AS daypart,
    COUNT(DISTINCT order_id) AS total_orders,
    SUM(quantity) AS total_pizzas_sold,
    ROUND(SUM(total_price), 2) AS total_revenue
FROM
    pizza_sales
GROUP BY
    daypart
ORDER BY
    total_revenue DESC;

-- 4. Hypothetical Analysis of Repeat Purchases
```

daypart	total_orders	total_pizzas_sold	total_revenue
Afternoon	12171	29468	486430.15
Evening	7248	15902	262170.95
Morning	1240	2750	45322.45
Night	691	1454	23936.5

17. This query segments sales data into four dayparts—Morning, Afternoon, Evening, and Night—based on order times. By analyzing orders, pizzas sold, and revenue for each segment, it helps identify time-based trends, informing targeted promotions and operational adjustments.



```sql
    total_revenue DESC;

-- 4. Hypothetical Analysis of Repeat Purchases
SELECT
    COUNT(DISTINCT order_id) AS total_orders,
    COUNT(DISTINCT CASE WHEN order_count > 1 THEN order_id END) AS repeat_orders,
    ROUND(SUM(total_price), 2) AS total_revenue,
    ROUND(SUM(total_price) / COUNT(DISTINCT order_id), 2) AS avg_order_value
FROM
    (SELECT
        order_id,
        COUNT(*) AS order_count,
        SUM(total_price) AS total_price
     FROM
        pizza_sales
     GROUP BY
        order_id
    ) AS order_summary;
```

total_orders	repeat_orders	total_revenue	avg_order_value
21350	13149	817860.05	38.31

18. The query evaluates customer behavior by analyzing repeat purchases, total revenue, and average order value. This information is valuable for understanding customer loyalty and designing initiatives, such as loyalty programs, to encourage repeat purchases and increase revenue.

Tableau Overview

Tableau is a powerful data visualization tool that transforms raw data into insightful, interactive visualizations and dashboards. By integrating Tableau with MySQL Workbench, I was able to analyze and present pizza sales data in a more accessible and actionable format. The following two dashboards were developed to provide in-depth insights into sales trends, pizza performance, and customer preferences:

1. **Home Page Dashboard**: Presents an overview of key trends such as total sales, order frequency, and category performance.
2. **Best/Worst Sellers Dashboard**: Focuses on identifying the highest and lowest-performing pizzas based on revenue, orders, and total units sold, allowing for targeted strategies.

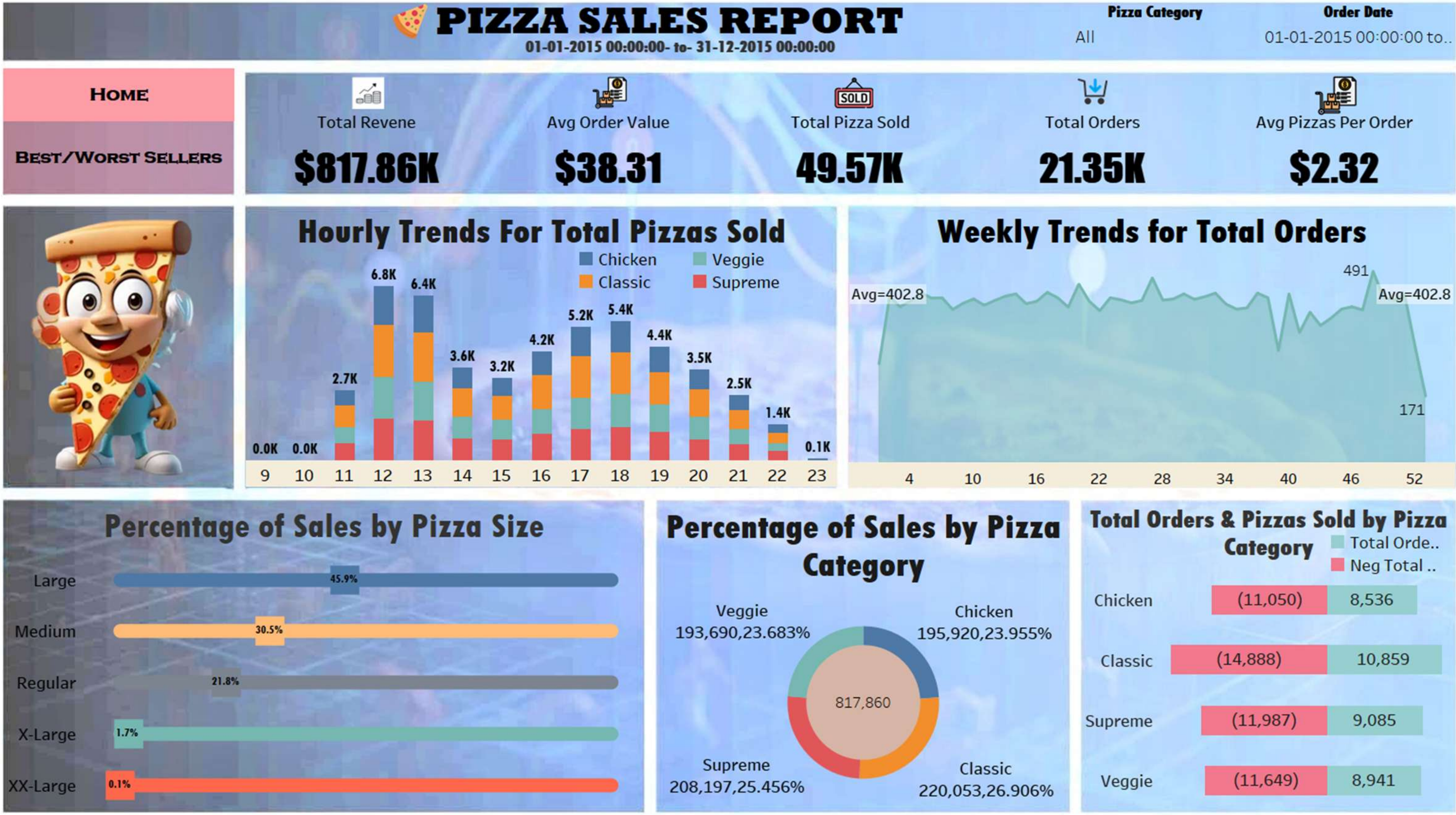
These dashboards are designed to offer a holistic understanding of the data, enabling informed decision-making for better business outcomes.

1. **Home Page Dashboard**

It provides a comprehensive summary of pizza sales trends and category-wise performance, including the following visualizations:

- ❑ **Hourly Trends for Total Pizzas Sold**: Displays how the total number of pizzas sold varies across different hours, identifying peak sales times for targeted promotions.
- ❑ **Weekly Trends for Total Orders**: Offers insights into how total orders fluctuate on a weekly basis, revealing seasonal or day-of-week patterns that influence ordering behavior.
- ❑ **Percentage of Sales by Pizza Size**: Breaks down the revenue share for different pizza sizes (Large, Medium, Regular, X-Large, XX-Large), helping to understand which sizes dominate the market.
- ❑ **Percentage of Sales by Pizza Category**: Shows the revenue distribution among pizza categories (Classic, Veggie, Supreme, Chicken), giving a clear view of which category is performing best.
- ❑ **Total Orders & Pizzas Sold by Pizza Category**: Highlights the total number of orders and pizzas sold by each category, offering a comprehensive view of overall demand across the product range.

This dashboard serves as an essential tool for monitoring overall performance and spotting trends that could inform pricing, marketing, and operational strategies, which is shown in the next page.



PIZZA SALES REPORT

01-01-2015 00:00:00- to- 31-12-2015 00:00:00

Pizza Category: All

Order Date: 01-01-2015 00:00:00 to..

HOME

BEST/WORST SELLERS

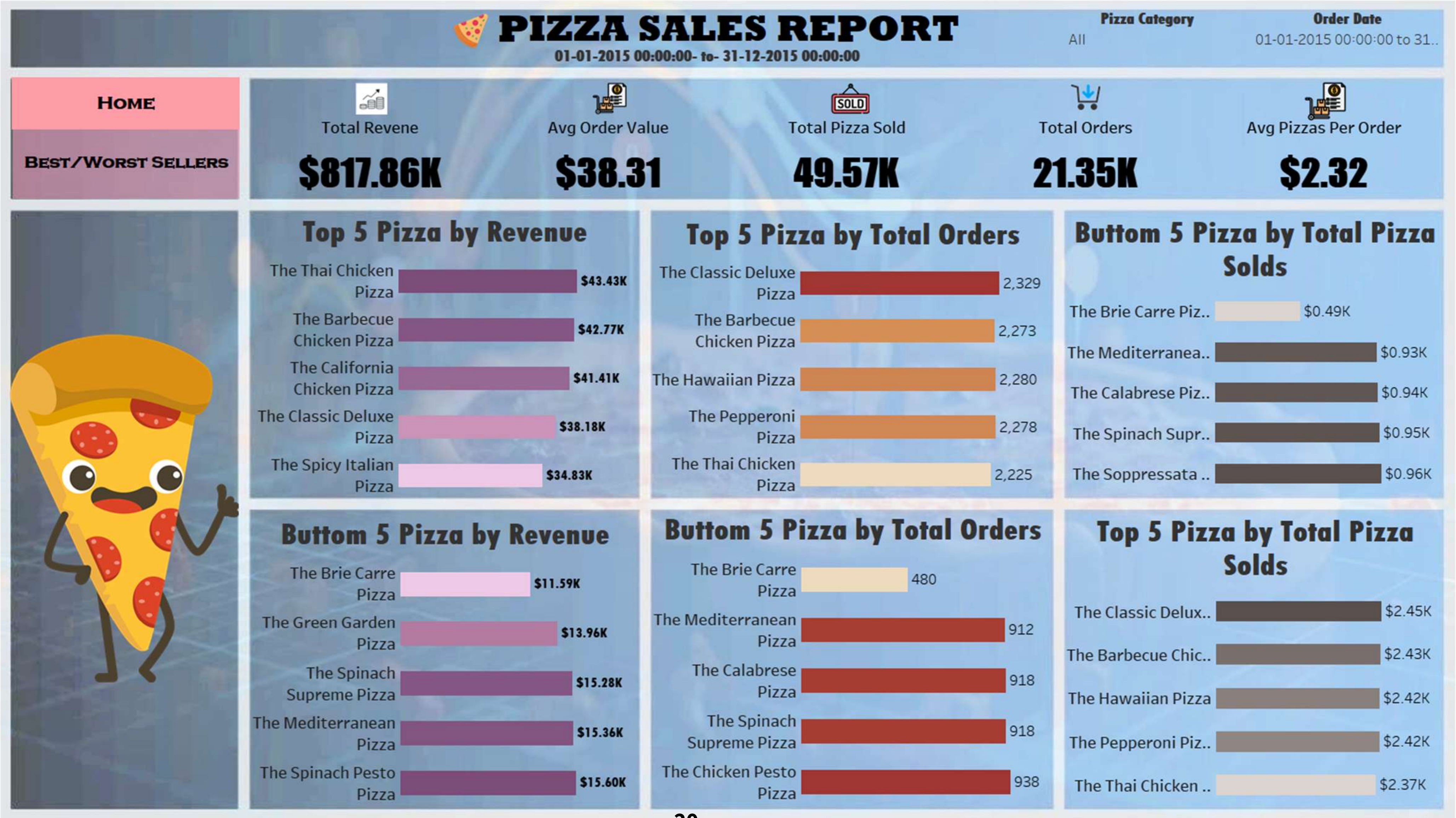
Total Revene	Avg Order Value	Total Pizza Sold	Total Orders	Avg Pizzas Per Order
$817.86K	$38.31	49.57K	21.35K	$2.32

2. **Best/Worst Sellers Dashboard**

It zooms in on individual pizza performance, allowing for deeper analysis of top and bottom performers. Key features include:

- ❑ **Top 5 Pizzas by Revenue**: Highlights the five pizzas that contribute the most revenue, allowing for a focus on the most lucrative products.
- ❑ **Bottom 5 Pizzas by Revenue**: Identifies the five pizzas that bring in the least revenue, providing an opportunity to assess whether adjustments are needed in ingredients, pricing, or marketing.
- ❑ **Top 5 Pizzas by Total Orders**: Lists the pizzas with the highest number of orders, indicating customer favorites and potentially guiding stock or promotional decisions.
- ❑ **Bottom 5 Pizzas by Total Orders**: Shows the least ordered pizzas, allowing for analysis of why certain pizzas may not be as popular and whether they should be repositioned or discontinued.
- ❑ **Top 5 Pizzas by Total Pizzas Sold**: Displays the top-selling pizzas in terms of total units sold, highlighting market demand and consumer preferences.
- ❑ **Bottom 5 Pizzas by Total Pizzas Sold**: Focuses on pizzas with the lowest sales volume, providing valuable insights into product underperformance.

This dashboard helps businesses prioritize high-performing products while addressing potential weaknesses in the product lineup. It enables data-driven decisions on menu optimization, marketing focus, and inventory management, which is shown in the next page.



PIZZA SALES REPORT

01-01-2015 00:00:00- to- 31-12-2015 00:00:00

Pizza Category: All

Order Date: 01-01-2015 00:00:00 to 31..

Total Revene	Avg Order Value	Total Pizza Sold	Total Orders	Avg Pizzas Per Order
$817.86K	$38.31	49.57K	21.35K	$2.32

Top 5 Pizza by Revenue

The Thai Chicken Pizza	$43.43K
The Barbecue Chicken Pizza	$42.77K
The California Chicken Pizza	$41.41K
The Classic Deluxe Pizza	$38.18K
The Spicy Italian Pizza	$34.83K

Top 5 Pizza by Total Orders

The Classic Deluxe Pizza	2,329
The Barbecue Chicken Pizza	2,273
The Hawaiian Pizza	2,280
The Pepperoni Pizza	2,278
The Thai Chicken Pizza	2,225

Buttom 5 Pizza by Total Pizza Solds

The Brie Carre Piz..	$0.49K
The Mediterranea..	$0.93K
The Calabrese Piz..	$0.94K
The Spinach Supr..	$0.95K
The Soppressata ..	$0.96K

Buttom 5 Pizza by Revenue

The Brie Carre Pizza	$11.59K
The Green Garden Pizza	$13.96K
The Spinach Supreme Pizza	$15.28K
The Mediterranean Pizza	$15.36K
The Spinach Pesto Pizza	$15.60K

Buttom 5 Pizza by Total Orders

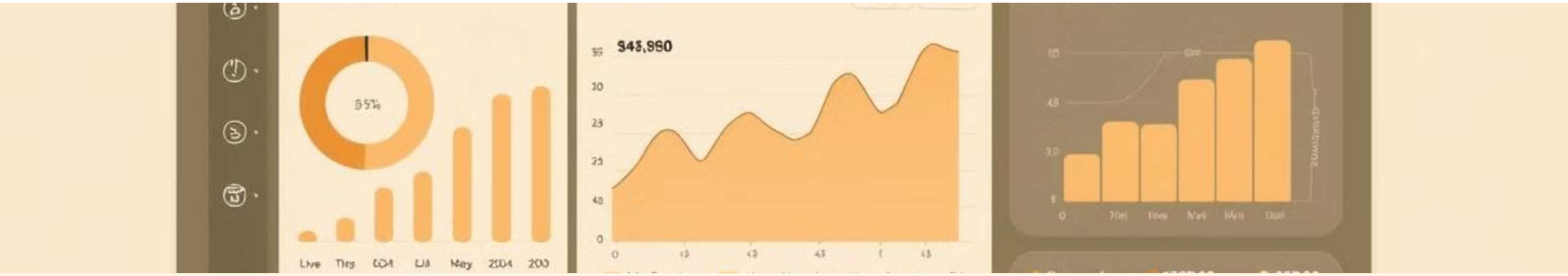
The Brie Carre Pizza	480
The Mediterranean Pizza	912
The Calabrese Pizza	918
The Spinach Supreme Pizza	918
The Chicken Pesto Pizza	938

Top 5 Pizza by Total Pizza Solds

The Classic Delux..	$2.45K
The Barbecue Chic..	$2.43K
The Hawaiian Pizza	$2.42K
The Pepperoni Piz..	$2.42K
The Thai Chicken ..	$2.37K

Conclusion

This project seamlessly integrated SQL and Tableau to deliver actionable insights from pizza sales data. Using SQL queries, we performed extensive data aggregation and filtering, allowing us to compute key metrics such as total orders, revenue, and pizza sales across different categories, sizes, and time periods. These SQL-based analyses formed the foundation for Tableau visualizations, where interactive dashboards were created to highlight critical trends, including hourly and weekly sales patterns, sales distribution by pizza category and size, and performance of both top and bottom-selling pizzas. The combination of precise SQL queries and Tableau's dynamic visual capabilities provided a comprehensive overview of sales performance, empowering stakeholders to make informed, data-driven decisions. Looking ahead, the incorporation of advanced predictive models could further refine sales forecasting, enhancing strategic planning and operational efficiency.



Future Work

For future work, will focus on integrating Advance predictive analytics, customer segmentation and Feedback Analysis and also Customer Segmentation to enhance decision-making and further optimize sales strategies."

Advanced Predictive Analytics:

Implement predictive models to forecast future sales trends, identify potential growth areas, and predict the impact of promotional campaigns. Machine learning algorithms such as time series forecasting or regression analysis can be used to predict demand and optimize inventory management.

Customer Satisfaction and Feedback Analysis:

Analyzing customer feedback (e.g., reviews, ratings) alongside sales data can provide a more holistic view of product performance. Sentiment analysis and text mining techniques could be applied to gain insights from customer comments and identify potential areas for product or service improvement.

Customer Segmentation:

Create customer segments based on purchase history and demographics to tailor marketing campaigns.

Acknowledgment

We would like to express our heartfelt gratitude to all those who supported and guided us throughout this project. Special thanks to our mentors for their invaluable insights and constant encouragement, which played a crucial role in shaping this analysis. We are also deeply appreciative of the online communities and resources that provided access to the pizza sales dataset, enabling this comprehensive study.