Pizza Sales Analysis (2015)

1. Loading CSV file into MySQL using Python

In [1]:
```python
import pandas as pd
from sqlalchemy import create_engine

# Step 1: Load the CSV file into a DataFrame
df = pd.read_csv("C:\\Users\\HP\\Downloads\\pizza_sales.csv")

# Step 2: Create the connection to MySQL
user = 'root'
password = 'Efootball@22'
host = 'localhost'
database = 'pizza_sales'

connection_string = f'mysql+pymysql://root:Efootball%4022@localhost/pizza_sales'
engine = create_engine(connection_string)

# Upload the DataFrame to MySQL
df.to_sql('pizza_sales', con=engine, if_exists='replace', index=False)

# Verify the data upload
result = pd.read_sql('SELECT * FROM pizza_sales LIMIT 3;', con=engine)
print(result)
```

```
   pizza_id  order_id  pizza_name_id  quantity  order_date order_time  \
0         1         1     hawaiian_m         1  01-01-2015   11:38:36
1         2         2  classic_dlx_m         1  01-01-2015   11:57:40
2         3         2  five_cheese_l         1  01-01-2015   11:57:40

   unit_price  total_price pizza_size pizza_category  \
0       13.25        13.25          M        Classic
1       16.00        16.00          M        Classic
2       18.50        18.50          L         Veggie

                                  pizza_ingredients                pizza_name
0           Sliced Ham, Pineapple, Mozzarella Cheese         The Hawaiian Pizza
1  Pepperoni, Mushrooms, Red Onions, Red Peppers,...  The Classic Deluxe Pizza
2  Mozzarella Cheese, Provolone Cheese, Smoked Go...     The Five Cheese Pizza
```

2. Changing data types of columns

```sql
CREATE DATABASE pizza_sales;
SELECT * FROM pizza_sales;
DESCRIBE pizza_sales;
SET SQL_SAFE_UPDATES = 0;
UPDATE pizza_sales
SET order_date = STR_TO_DATE(order_date, '%d-%m-%Y');
SET SQL_SAFE_UPDATES = 1;
ALTER TABLE pizza_sales
MODIFY COLUMN pizza_id INT,
MODIFY COLUMN order_id INT,
MODIFY COLUMN pizza_name_id VARCHAR(50),
MODIFY COLUMN quantity INT,
MODIFY COLUMN order_date DATE,
MODIFY COLUMN order_time TIME,
MODIFY COLUMN unit_price FLOAT,
MODIFY COLUMN total_price FLOAT,
MODIFY COLUMN pizza_size VARCHAR(50),
MODIFY COLUMN pizza_category VARCHAR(50),
MODIFY COLUMN pizza_ingredients VARCHAR(200),
MODIFY COLUMN pizza_name VARCHAR(50);
```

3. KPIs Requirement

```
26      -- 1. Total Revenue
27 •    SELECT SUM(total_price) AS Total_Revenue FROM pizza_sales;
```

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

Total_Revenue
817860.0508384705

```
29      -- 2.  Average Order Value
30 •    SELECT SUM(total_price) / COUNT(DISTINCT order_id) AS Average_order_value FROM pizza_sales;
```

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

Average_order_value
38.30726233435459

```
32      -- 3. Total Pizza Sold
33 •    SELECT SUM(quantity) AS Total_Pizza_Sold FROM pizza_sales;
```

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

Total_Pizza_Sold
49574

```
35      -- 4. Total Order Placed
36 •    SELECT COUNT(DISTINCT order_id) AS Total_order_placed FROM pizza_sales;
37
```

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

Total_order_placed
21350

```
38      -- 5. Average Pizzas per Order
39 •    SELECT SUM(quantity) / COUNT(DISTINCT order_id) AS Average_Pizzas_per_Order FROM pizza_sales;
```

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

Average_Pizzas_per_Order
2.3220

4. CHART REQUIREMENT

```
42
43      -- 1. Daily Trend for Total Order
44 •    SELECT DAYNAME(order_date) AS Order_day, COUNT(DISTINCT order_id) AS Total_Order FROM pizza_sales
45      GROUP BY DAYNAME(order_date);
```

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

Order_day	Total_Order
Friday	3538
Monday	2794
Saturday	3158
Sunday	2624
Thursday	3239
Tuesday	2973
Wednesday	3024

```
47      -- 1. Monthly Trend for Total Order
48 •    SELECT order_date FROM pizza_sales LIMIT 10;
49 •    SELECT
50        DATE_FORMAT(order_date, '%Y-%m') AS Month_Year,
51        COUNT(DISTINCT order_id) AS Total_Order
52      FROM pizza_sales
53      GROUP BY Month_Year
54      ORDER BY Total_Order DESC;
```

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

Month_Year	Total_Order
2015-07	1935
2015-05	1853
2015-01	1845
2015-08	1841
2015-03	1840
2015-04	1799
2015-11	1792
2015-06	1773
2015-02	1685
2015-12	1680
2015-09	1661
2015-10	1646

```sql
-- 3. Percentage of sales by pizza category
SELECT pizza_category, SUM(total_price) * 100 / (SELECT SUM(total_price) FROM pizza_sales) AS Precentage_of_sale
FROM pizza_sales
GROUP BY pizza_category;
```

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

pizza_category	Precentage_of_sale
Classic	26.905960230697634
Veggie	23.682591025867666
Supreme	25.456311211146232
Chicken	23.95513753228847

```sql
SELECT pizza_size, SUM(total_price) * 100 / (SELECT SUM(total_price) FROM pizza_sales WHERE MONTH(order_date)=1) AS Precentage_of_sale
FROM pizza_sales
WHERE MONTH(order_date)=1
GROUP BY pizza_size;
```

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

pizza_size	Precentage_of_sale
M	30.007894712238155
L	46.42193455128789
S	21.640329332997844
XL	1.8268229168049106
XXL	0.10301848667120285

```sql
-- 5. Total pizza sold by pizza category
SELECT pizza_category, SUM(quantity) AS Total_Pizzas_Sold
FROM pizza_sales
GROUP BY pizza_category
ORDER BY Total_Pizzas_Sold DESC;
```

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

pizza_category	Total_Pizzas_Sold
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

```sql
-- 6. Best 5 sellers
SELECT pizza_name, SUM(total_price) AS Total_revenue FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_revenue DESC LIMIT 5;
```

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

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

```sql
-- 7. Worst 5 sellers
SELECT pizza_name, SUM(total_price) AS Total_revenue FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_revenue ASC LIMIT 5;
```

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

pizza_name	Total_revenue
The Brie Carre Pizza	11588.499813079834
The Green Garden Pizza	13955.75
The Spinach Supreme Pizza	15277.75
The Mediterranean Pizza	15360.5
The Spinach Pesto Pizza	15596

```sql
-- 6. Best 5 sellers(wrt quantity)
SELECT pizza_name, SUM(quantity) AS Total_quantity FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_quantity DESC LIMIT 5;
```

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

pizza_name	Total_quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

```sql
-- 9. Worst 5 sellers(wrt quantity)
SELECT pizza_name, SUM(quantity) AS Total_quantity FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_quantity ASC LIMIT 5;
```

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

pizza_name	Total_quantity
The Brie Carre Pizza	490
The Mediterranean Pizza	934
The Calabrese Pizza	937
The Spinach Supreme Pizza	950
The Soppressata Pizza	961

```sql
-- 10. Best 5 sellers(wrt orders)
SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_orders FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_orders DESC LIMIT 5;
```

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

pizza_name	Total_orders
The Classic Deluxe Pizza	2329
The Hawaiian Pizza	2280
The Pepperoni Pizza	2278
The Barbecue Chicken Pizza	2273
The Thai Chicken Pizza	2225

```sql
-- 11. Worst 5 sellers(wrt orders)
SELECT pizza_name, COUNT(DISTINCT order_id) AS Total_orders FROM pizza_sales
GROUP BY pizza_name
ORDER BY Total_orders ASC LIMIT 5;
```

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

pizza_name	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