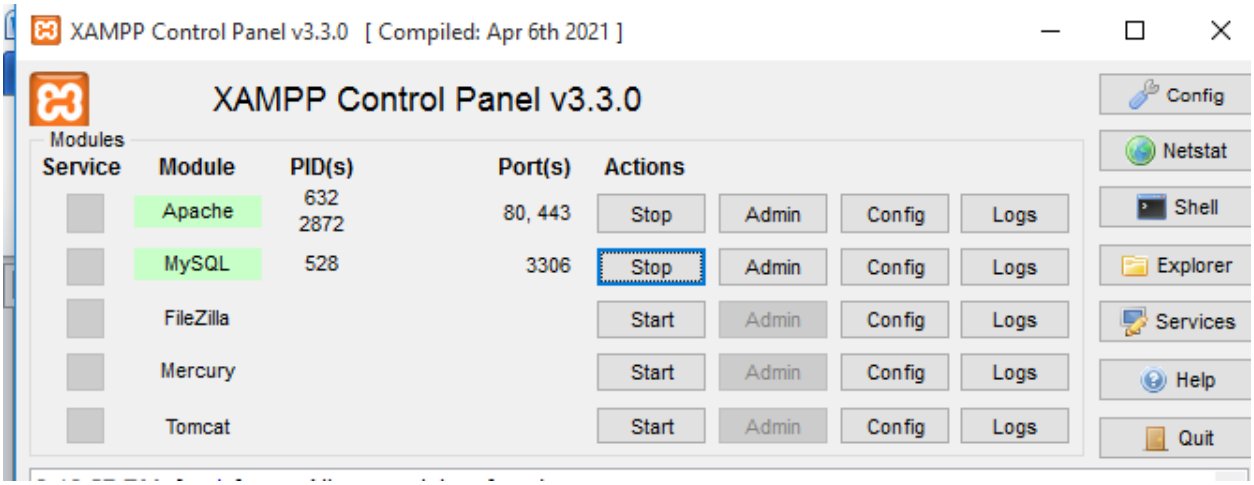
Data Analysis on MYSQL database :-

To perform data analysis in Python using a MySQL database, you need to connect to the database, execute SQL queries, fetch the data, and then analyze it. Here's a step-by-step guide:

Prerequisites

1. **Install Xampp**: Make sure MySQL is installed and running on your system.
2. **Install necessary libraries**: You need the following Python libraries:
    - `mysql-connector`: For connecting to the MySQL database.
    - `pandas`: For data manipulation and analysis.

Run your Xampp control panel and start Apache and MYSQL both as shown below



You can install them using pip:

```
pip install mysql-connector-python pandas
```

Create car database and table cars as shown below


localhost/phpmyadmin/index.php?route=/sql&pos=0&db=car&table=cars
phpMyAdmin
Recent
Favorites
New
amardeep
car
New
cars
demo
demo1
ecom
el
employee_management_system
information_schema
laravel
mahenoor
mydb
mysql
my_db
Server: 127.0.0.1 » Database: car » Table: cars
Browse
Structure
SQL
Search
Insert
Showing rows 0 - 3 (4 total, Query took 0.0003 seconds.)
SELECT * FROM `cars`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Extra options
id
model
price
Edit Copy Delete 1 bmw 12
Edit Copy Delete 2 y 656
Edit Copy Delete 3 y 123
Edit Copy Delete 4 y 1223
Check all
With selected:
Edit
Copy
Delete

For Data Analysis Python Code:-

```
import mysql.connector
import pandas as pd

# Step 1: Connect to the MySQL database
connection = mysql.connector.connect(
    host="localhost",        # Database host (e.g., 'localhost' or an IP address)
    user="root",             # Database username
    password="",     # Database password
    database="car" # Name of the database you want to use
)

# Step 2: Execute SQL query to fetch data
query = "SELECT * FROM cars"  # Replace with your table name and query
df = pd.read_sql(query, connection)  # Fetch the data and store it in a pandas
DataFrame

# Step 3: Perform data analysis using pandas

# Example 1: Display the first few rows of the dataset
print(df.head())

# Example 2: Get summary statistics
print(df.describe())

# Example 3: Perform data manipulation (e.g., filtering data)
filtered_data = df[df['price'] > 500]  # Filter rows where 'column_name' is
greater than 100

print("filered data ")
print(filtered_data)

# Example 5: Visualize the data (optional, requires matplotlib or seaborn)
import matplotlib.pyplot as plt
df['price'].hist()  # Example: Histogram for a specific column
plt.show()
```

Output:-

```
  df = pd.read_sql(query, connection)  # Fetch the data and store it in a pandas DataFrame
   id model  price
0   1   bmw     12
1   2     y    656
2   3     y    123
3   4     y   1223
             id        price
count  4.000000     4.000000
mean   2.500000   503.500000
std    1.290994   555.964327
min    1.000000    12.000000
25%    1.750000    95.250000
50%    2.500000   389.500000
75%    3.250000   797.750000
max    4.000000  1223.000000
filered data
   id model  price
1   2     y    656
3   4     y   1223
```

Breakdown of the Code:

- **Connecting to MySQL**: The `mysql.connector.connect()` function is used to establish a connection to your MySQL database.
- **Executing Queries**: We use `pandas.read_sql()` to directly execute the query and store the result in a pandas DataFrame. This allows you to manipulate and analyze the data using pandas functions.
- **Data Analysis**:
    - `df.head()`: Displays the first few rows of the data.
    - `df.describe()`: Provides summary statistics (mean, standard deviation, min, max, etc.) for numeric columns.
    - `df[df['column_name'] > 100]`: Filters rows based on a condition.

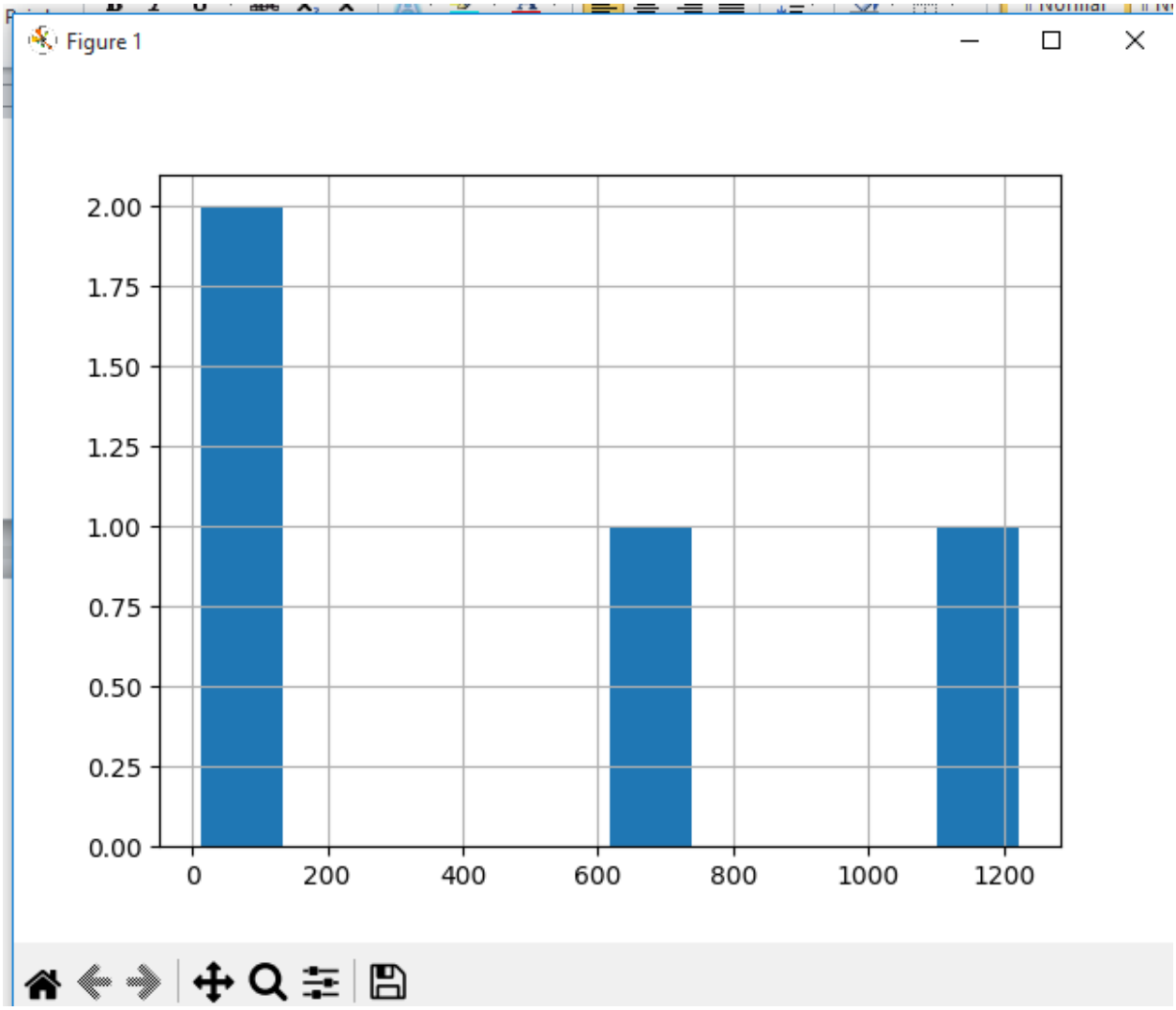


Figure 1
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
200
400
600
800
1000
1200