Data Wrangling Assignment Report

# Introduction

This report explains the workflow of two main components of the assignment: 'TMA\_data.sql' and 'Data\_Wrangling.ipynb'. The 'TMA\_data.sql' script creates and populates the database, while the 'Data\_Wrangling.ipynb' Jupyter notebook performs the data wrangling, table creation, and visualization tasks.

# 1. TMA\_data.sql

The 'TMA\_data.sql' script creates the database named 'TMA\_data', defines the structure of the 'TMA\_data' table, and populates the table with recruitment data. This table holds details about offers made and accepted for different locations and departments.

## SQL Code Explanation

The script first checks if the 'TMA\_data' database exists, and if not, it creates the database. Then, the 'TMA\_data' table is created with fields such as 'Location', 'Department', 'headcount', 'Offers\_Recruitment\_Firm1', and other relevant columns. The recruitment data is inserted into this table.

CREATE DATABASE IF NOT EXISTS TMA\_data;  
USE TMA\_data;  
  
CREATE TABLE IF NOT EXISTS TMA\_data (  
 Location VARCHAR(50),  
 Department VARCHAR(50),  
 headcount INT,  
 Offers\_Recruitment\_Firm1 VARCHAR(50),  
 Offers\_Recruitment\_Firm2 VARCHAR(50),  
 Offers\_Recruitment\_Firm3 VARCHAR(50),  
 Offers\_Total VARCHAR(50),  
 Acceptance\_Recruitment\_Firm1 VARCHAR(50),  
 Acceptance\_Recruitment\_Firm2 VARCHAR(50),  
 Acceptance\_Recruitment\_Firm3 VARCHAR(50),  
 Acceptance\_Total VARCHAR(50)  
);  
  
INSERT INTO `TMA\_data`  
 ...

# 2. Data\_Wrangling.ipynb

The 'Data\_Wrangling.ipynb' notebook performs data wrangling, table creation, and visualization using MySQL and Python. It connects to the 'TMA\_data' database, creates two new tables ('easy\_data' and 'fig1'), fetches the data, and then visualizes the data using Matplotlib.

## Notebook Code Explanation

1. MySQL Connection: A function is created to connect to the MySQL database using the MySQL Connector in Python.

def connect\_to\_db():  
 connection = mysql.connector.connect(  
 host='localhost',  
 user='root',  
 password='Toor',  
 database='TMA\_data'  
 )  
 return connection

2. Creating the 'easy\_data' Table: A new table named 'easy\_data' is created based on the 'TMA\_data' table. This table contains key columns for easier analysis.

def create\_easy\_data\_table():  
 cursor.execute('''  
 CREATE TABLE easy\_data AS  
 SELECT Location, Department, headcount AS Total\_Headcount, Offers\_Recruitment\_Firm1, ...  
 ''')

3. Creating the 'fig1' Table: Another table 'fig1' is created to aggregate the recruitment data for analysis. It sums up the offers made and accepted by location and department.

def create\_fig1\_table():  
 cursor.execute('''  
 CREATE TABLE fig1 AS  
 SELECT Location, Department, SUM(headcount), ...  
 ''')

## Data Visualization

The notebook fetches the data from the 'easy\_data' table, and the data is displayed in a custom table format using Matplotlib. The table is visually styled to make it easy to interpret.

def display\_custom\_table(df):  
 fig, ax = plt.subplots(figsize=(16, 8))  
 ax.table(cellText=df.values, colLabels=df.columns, cellLoc='center', loc='center')  
 plt.show()

# Conclusion

This assignment demonstrates the full workflow from setting up the database and populating it with data to performing data wrangling, creating new tables, and visualizing the results. The 'TMA\_data.sql' file handles the database creation, while the 'Data\_Wrangling.ipynb' provides the data processing and visualization pipeline.