Documentation-2347107

Aryan Majhi 2347107

AKASA - Task 1 (Python + SQL)

Github repo : https://github.com/aryanmajhi75/AKASA

Native System Configuration

- AMD Ryzen 5 7000 series
- 16 GB RAM
- Ubuntu 24.04 LTS

Dependencies

- Python libraries used
  - mysql-connector
  - pandas
  - datetime
  - matplotlib.pyplot
  - numpy
  - seaborn
  - scripy.stats
- Docker image for mysql 8.1
- Python Interpreter version 3.12.3

Setup Instructions

> Note: the instructions are based on my own system i.e., Ubuntu Linux 24.04 LTS, if you are using any other system then follow the instructions for that OS.

- Install docker using cli, for ubuntu it works like this:

```
sudo apt-get update
sudo apt-get install ./docker-desktop-<arch>.deb
```

For installation guide for other OS, follow the documentation :
https://docs.docker.com/engine/install/

- Pull mysql 8.1 image to docker

```
docker pull mysql:8.1
```

- Run docker image mysql 8.1 with credentials and port number

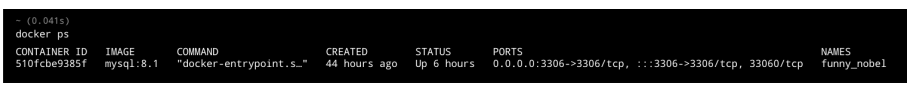
```
docker volume create akasa

docker volume inspect akasa

docker run -d \
-e MYSQL_ROOT_PASSWORD=1234 \
-e MYSQL_PASSWORD=1234 \
-e MYSQL_USER=aryan \
-e MYSQL_DATABASE=aviation_data \
-v mysql_volume:/var/lib/mysql \
-p 3306:3306 \
mysql:8.1
```

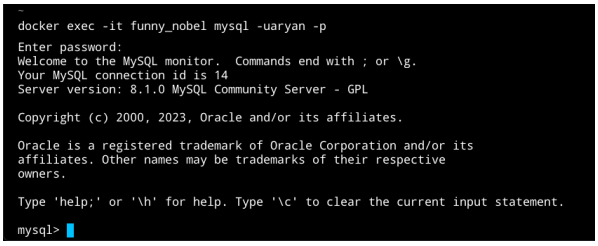
- Find the name of the image (last column), for my case it is called **funny_nobel**

```
docker ps
```



- Execute the docker image with the name of the image

```
docker exec -it funny_nobel m
```



- If the docker image was stopped, we need to start the image again for mysql to run.

```
docker start <image-name>
docker exec -it funny_nobel mysql -uaryan -p
```

- After mysql is setup, we need to make a table FlightSchedule and then fill the table.

```
// Create the table FlightSchedule
CREATE TABLE FlightSchedule (
    FlightNumber VARCHAR(10),
    DepartureDate VARCHAR(10),
    DepartureTime VARCHAR(8),
    ArrivalDate VARCHAR(10),
    ArrivalTime VARCHAR(8),
    Airline VARCHAR(50),
    DelayMinutes INT
);

// Insert data into the table
INSERT INTO FlightSchedule (FlightNumber, DepartureDate, Departu
VALUES
('AA1234', '09/01/2023', '08:30 AM', '09/01/2023', '10:45 AM',
('DL5678', '09/01/2023', '01:15 PM', '09/01/2023', '03:30 PM',
('UA9101', '09/01/2023', '05:00 PM', '09/01/2023', '07:15 PM',
('AA1234', '09/01/2023', '08:30 AM', '09/01/2023', '10:45 PM',
('DL5678', '09/02/2023', '02:00 PM', '09/02/2023', '04:10 PM',
```

```
('UA9101', '09/02/2023', '05:00 PM', '09/02/2023', '07:15 PM',
('AA1234', '09/02/2023', '08:30 PM', '09/03/2023', '10:45 AM',
('DL5678', '09/03/2023', '01:00 PM', '09/03/2023', '03:30 PM',
('UA9101', '09/03/2023', '03:00 PM', '09/03/2023', '05:20 PM',
('AA1234', '09/03/2023', '08:30 AM', '09/03/2023', '10:00 AM',
('DL5678', '09/04/2023', '12:30 PM', '09/04/2023', '02:40 PM',
('UA9101', '09/04/2023', '07:00 PM', '09/04/2023', '09:15 PM',
```

Since I’m using Ubuntu, the OS doesn’t recommend to install python packages to root, so I have an environment where the packages are saved

```
mkdir -p ~/.venvs
python3 -m venv ~/.venvs/<some-name>
~/.venvs/<some-name>/bin/python -m pip install <package-name>
```

The whole python script is running in a venv called **newenv**

```
python3 -m venv newenv
source newenv/bin/activate
```

To install any packages, we need to install it in the **newenv and in the venvs**

```
~/.venvs/mysql/bin/python -m pip install <package-name>
pip install <package-name>
```

After the installations are done, we need to run the python script named **analytics.py**

```
python3 analytics.py
```

After running all the graphs are stored in the same folder as png files and csv files for checking the changes in the dataset after every section - 1,2,3

The folder structure will be as follows: