ASSIGNMENT 9

Deadline: 25.06.2021

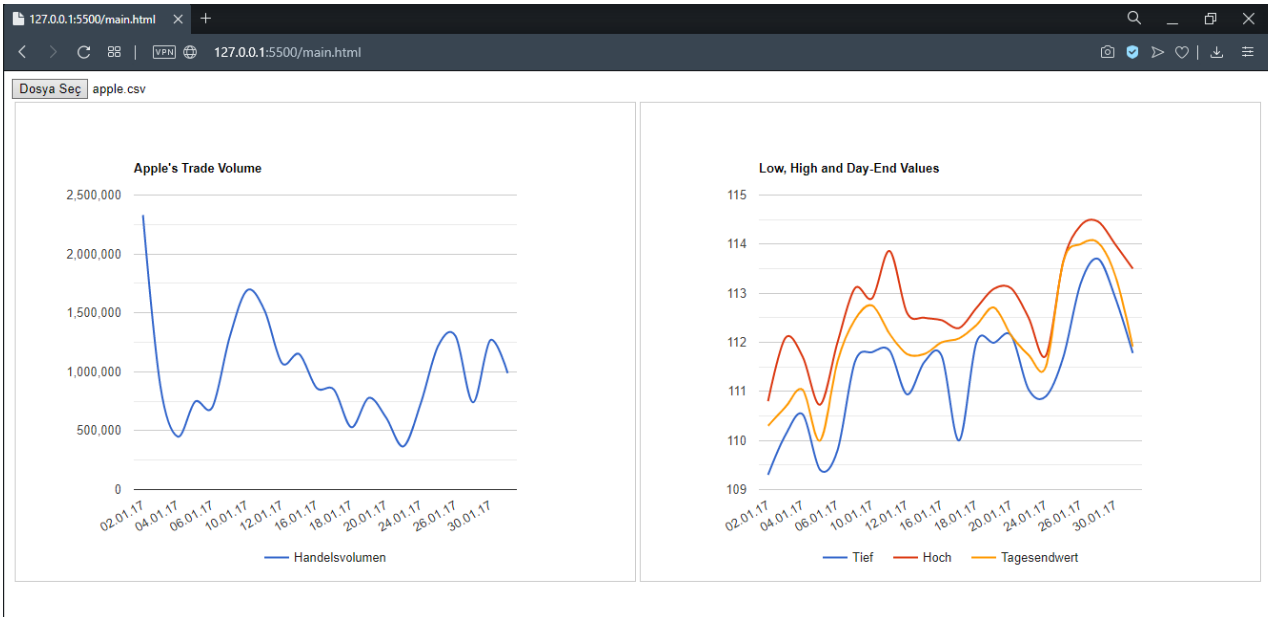
Group members: Ali Omarov, Simon Ruber, Esra Gücükbel
Our Github link: https://github.com/esragucukbel/FU_DBS_A09

Exercise 1. 3 p.

a) Download the apple.csv file from the KVV/Whiteboard. You can find the file under Resources/Assignments.
b) Create a (simple) visualization of the chart of Apple stock using Java Script.

Summary:

An HTML page which includes a two column layout was designed to get a CSV file as input. When the CSV file is uploaded, two charts are drawn. We used two charts because values of “Trade Volume” are much bigger than “Low, High and End-of-Day” values.

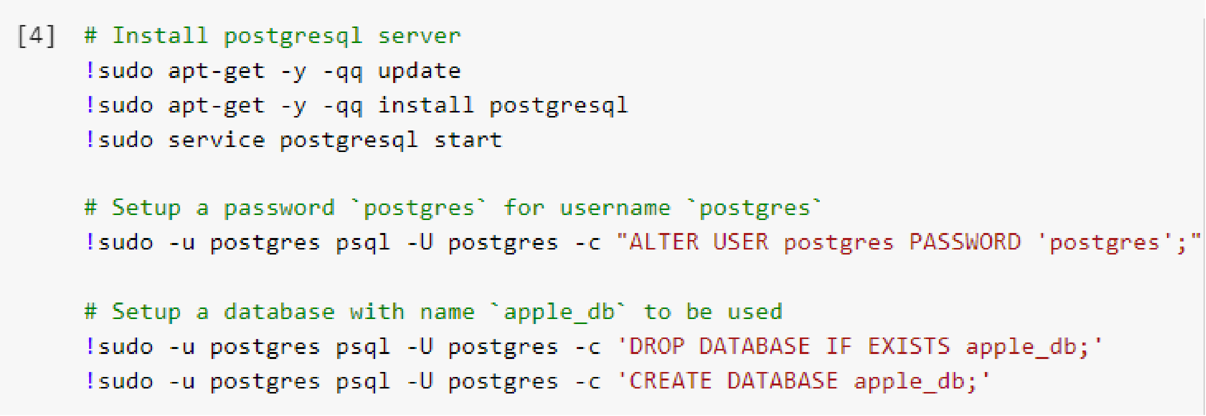


Exercise 2. 6 p.

a) Write a program in a programming language of your choice that connects to a Postgresql database and retrieves and displays data from it.
b) Modify your program so that the data is changed by the program before it is displayed.

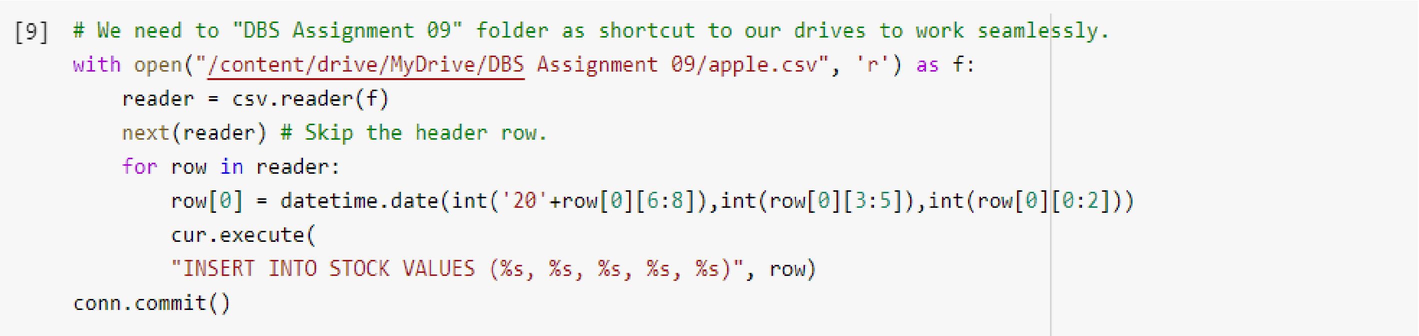
Summary:

We used Google Colab(python) to get a connection with PostgreSQL. All environment variables were setted up and the database was created.



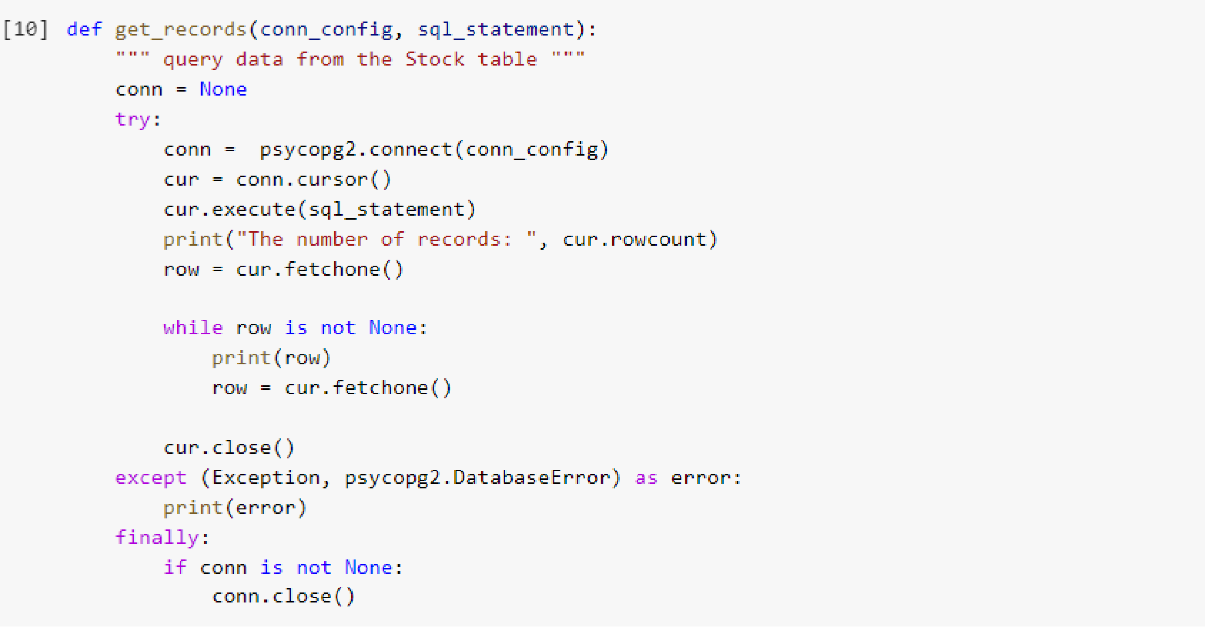
```
[4]  # Install postgresql server
     !sudo apt-get -y -qq update
     !sudo apt-get -y -qq install postgresql
     !sudo service postgresql start

     # Setup a password `postgres` for username `postgres`
     !sudo -u postgres psql -U postgres -c "ALTER USER postgres PASSWORD 'postgres';"

     # Setup a database with name `apple_db` to be used
     !sudo -u postgres psql -U postgres -c 'DROP DATABASE IF EXISTS apple_db;'
     !sudo -u postgres psql -U postgres -c 'CREATE DATABASE apple_db;'
```

Data was obtained by using Google Drive. Date field is changed when reading file to be compatible with PostgreSQL date format.



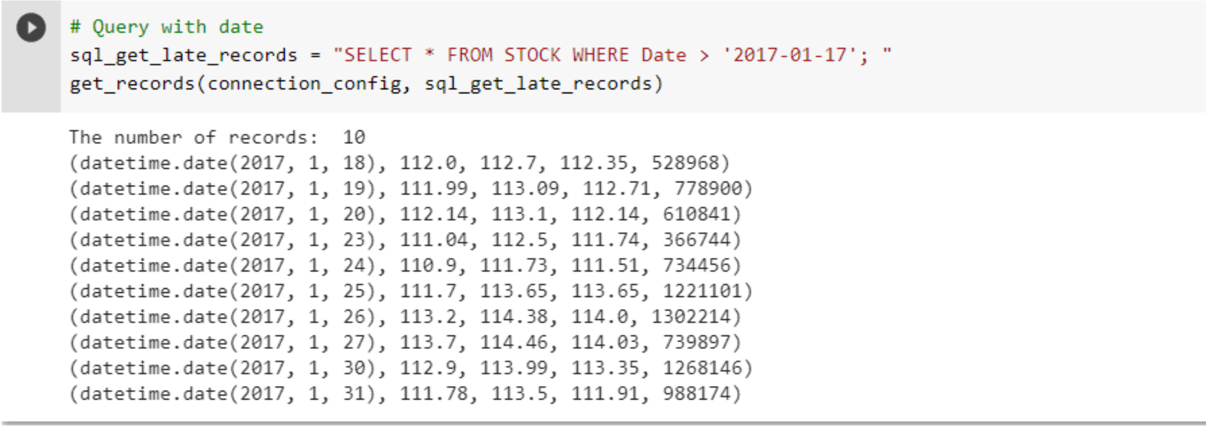
```
[9]  # We need to "DBS Assignment 09" folder as shortcut to our drives to work seamlessly.
     with open("/content/drive/MyDrive/DBS Assignment 09/apple.csv", 'r') as f:
         reader = csv.reader(f)
         next(reader) # Skip the header row.
         for row in reader:
             row[0] = datetime.date(int('20'+row[0][6:8]),int(row[0][3:5]),int(row[0][0:2]))
             cur.execute(
             "INSERT INTO STOCK VALUES (%s, %s, %s, %s, %s)", row)
     conn.commit()
```

The code is modified to run different SQL queries.



```
[10] def get_records(conn_config, sql_statement):
         """ query data from the Stock table """
         conn = None
         try:
             conn =  psycopg2.connect(conn_config)
             cur = conn.cursor()
             cur.execute(sql_statement)
             print("The number of records: ", cur.rowcount)
             row = cur.fetchone()

             while row is not None:
                 print(row)
                 row = cur.fetchone()

             cur.close()
         except (Exception, psycopg2.DatabaseError) as error:
             print(error)
         finally:
             if conn is not None:
                 conn.close()
```

An example from queries. The query shows the records whose dates are later than ‘2017-01-17’.



```
# Query with date
sql_get_late_records = "SELECT * FROM STOCK WHERE Date > '2017-01-17'; "
get_records(connection_config, sql_get_late_records)
```

```
The number of records:  10
(datetime.date(2017, 1, 18), 112.0, 112.7, 112.35, 528968)
(datetime.date(2017, 1, 19), 111.99, 113.09, 112.71, 778900)
(datetime.date(2017, 1, 20), 112.14, 113.1, 112.14, 610841)
(datetime.date(2017, 1, 23), 111.04, 112.5, 111.74, 366744)
(datetime.date(2017, 1, 24), 110.9, 111.73, 111.51, 734456)
(datetime.date(2017, 1, 25), 111.7, 113.65, 113.65, 1221101)
(datetime.date(2017, 1, 26), 113.2, 114.38, 114.0, 1302214)
(datetime.date(2017, 1, 27), 113.7, 114.46, 114.03, 739897)
(datetime.date(2017, 1, 30), 112.9, 113.99, 113.35, 1268146)
(datetime.date(2017, 1, 31), 111.78, 113.5, 111.91, 988174)
```