Data X Berkeley

Plaksha SQL assignment

Submission details:

Please submit this as a Jupyter Notebook and a PDF of your results (both should show output). Also push your solutions to Github.

For the submision create a local database with `sqlite3` or `sqlalchemy` in a Jupyter notebook and make the queries either with a cursor object (and then print the results) or by using pandas `pd.read_sql_query()` .

When completing this homework you can experiment with SQL commands by utilizing this great online editor:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_all

There are already some tables in the online Database, namely:

`Categories, Employees, OrderDetails, Orders, Products, Shippers, and Suppliers.`

If you want you can drop them by running `DROP TABLE [table-name];` (or just keep them).

Exercises:

First create a table called students. It has the columns: 'student_id', 'name', 'major', 'gpa' and 'enrollment_date' We will use a new form of `CREATE TABLE` expression to produce this table.

Note that you can improve this and are welcome to do so -- e.g. by specifying for example a PRIMARY KEY and a FOREIGN KEY in Q2 :)

```
CREATE TABLE students AS
    SELECT 1 AS student_id, "John" AS name, "Computer Science" AS major, 3.5 AS gpa, "01-01-2022" AS enrollment_date UNION
    SELECT 2, "Jane", "Physics", 3.8, "01-02-2022" UNION
    SELECT 3, "Bob", "Engineering", 3.0, "01-03-2022" UNION
    SELECT 4, "Samantha", "Physics", 3.9, "01-04-2022" UNION
    SELECT 5, "James", "Engineering", 3.7, "01-05-2022" UNION
    SELECT 6, "Emily", "Computer Science", 3.6, "01-06-2022" UNION
    SELECT 7, "Michael", "Computer Science", 3.2, "01-07-2022" UNION
    SELECT 8, "Jessica", "Engineering", 3.8, "01-08-2022" UNION
    SELECT 9, "Jacob", "Physics", 3.4, "01-09-2022" UNION
    SELECT 10, "Ashley", "Physics", 3.9, "01-10-2022";
```

Q1 Simple SELECTS (on the students table)

1. SELECT all records in the table.
2. SELECT students whose major is "Computer Science".
3. SELECT all unique majors (use SELECT DISTINCT) and order them by name, descending order (i.e. Physics first).
4. SELECT all students that have an 'e' in their name and order them by gpa in ascending order.

Q2 Joins

Create a new table called courses, which indicates the courses taken by the students.

Create the table by running:

```
CREATE TABLE courses AS
    SELECT 1 AS course_id, "Python programming" AS course_name, 1 AS student_id, "A" AS grade UNION
    SELECT 2, "Data Structures", 2, "B" UNION
    SELECT 3, "Database Systems", 3, "B" UNION
    SELECT 1, "Python programming", 4, "A" UNION
    SELECT 4, "Quantum Mechanics", 5, "C" UNION
    SELECT 1, "Python programming", 6, "F" UNION
    SELECT 2, "Data Structures", 7, "C" UNION
    SELECT 3, "Database Systems", 8, "A" UNION
    SELECT 4, "Quantum Mechanics", 9, "A" UNION
    SELECT 2, "Data Structures", 10, "F";
```

1. COUNT the number of unique courses.
2. JOIN the tables students and courses and COUNT the number of students with the major Computer Science taking the course Python programming.
3. JOIN the tables students and courses and select the students who have grades higher than "C", only show their name, major, gpa, course_name and grade.

Q3 Aggregate functions, numerical logic and grouping

1. Find the average gpa of all students.
2. SELECT the student with the maximum gpa, display only their student_id, major and gpa
3. SELECT the student with the minimum gpa, display only their student_id, major and gpa
4. SELECT the students with a gpa greater than 3.6 in the majors of "Physics" and "Engineering", display only their student_id, major and gpa
5. Group the students by their major and retrieve the average grade of each major.
6. SELECT the top 2 students with the highest GPA in each major and order the results by major in ascending order, then by GPA in descending order

Your solution

OPEN CONNECTION TO DATABASE AND CREATE TABLE **STUDENT**

In [ ]:
```
import sqlite3

connect=sqlite3.connect("assignment-3.db")
cursor=connect.cursor()
create_query=  """CREATE TABLE IF NOT EXISTS students AS
        SELECT 1 AS student_id, "John" AS name, "Computer Science" AS major, 3.5 AS gpa, "01-01-2022" AS enrollment_date UNION
        SELECT 2, "Jane", "Physics", 3.8, "01-02-2022" UNION
        SELECT 3, "Bob", "Engineering", 3.0, "01-03-2022" UNION
        SELECT 4, "Samantha", "Physics", 3.9, "01-04-2022" UNION
        SELECT 5, "James", "Engineering", 3.7, "01-05-2022" UNION
        SELECT 6, "Emily", "Computer Science", 3.6, "01-06-2022" UNION
        SELECT 7, "Michael", "Computer Science", 3.2, "01-07-2022" UNION
        SELECT 8, "Jessica", "Engineering", 3.8, "01-08-2022" UNION
        SELECT 9, "Jacob", "Physics", 3.4, "01-09-2022" UNION
        SELECT 10, "Ashley", "Physics", 3.9, "01-10-2022";"""
cursor.execute(create_query)
```

Out[ ]:
```
<sqlite3.Cursor at 0x2317270ea40>
```

Q1

1

In [ ]:
```
a1=cursor.execute("select * from students;")
for row in a1.fetchall():
        print(row)
connect.commit()
```

```
(1, 'John', 'Computer Science', 3.5, '01-01-2022')
(2, 'Jane', 'Physics', 3.8, '01-02-2022')
(3, 'Bob', 'Engineering', 3.0, '01-03-2022')
(4, 'Samantha', 'Physics', 3.9, '01-04-2022')
(5, 'James', 'Engineering', 3.7, '01-05-2022')
(6, 'Emily', 'Computer Science', 3.6, '01-06-2022')
(7, 'Michael', 'Computer Science', 3.2, '01-07-2022')
(8, 'Jessica', 'Engineering', 3.8, '01-08-2022')
(9, 'Jacob', 'Physics', 3.4, '01-09-2022')
(10, 'Ashley', 'Physics', 3.9, '01-10-2022')
```

2

In [ ]:
```
a2=cursor.execute("select * from students where major='Computer Science';")
for row in a2.fetchall():
        print(row)
connect.commit()
```

```
(1, 'John', 'Computer Science', 3.5, '01-01-2022')
(6, 'Emily', 'Computer Science', 3.6, '01-06-2022')
(7, 'Michael', 'Computer Science', 3.2, '01-07-2022')
```

3

In [ ]:
```
a3=cursor.execute("SELECt DISTINCT major from students order by major DESC;")
for row in a3.fetchall():
        print(row)
connect.commit()
```

```
('Physics',)
('Engineering',)
('Computer Science',)
```

4

In [ ]:
```
a4=cursor.execute("SELECt * FROM students WHERE name LIKE '%e%' ORDER BY gpa;")
for row in a4.fetchall():
        print(row)
connect.commit()
```

```
(7, 'Michael', 'Computer Science', 3.2, '01-07-2022')
(6, 'Emily', 'Computer Science', 3.6, '01-06-2022')
(5, 'James', 'Engineering', 3.7, '01-05-2022')
(2, 'Jane', 'Physics', 3.8, '01-02-2022')
(8, 'Jessica', 'Engineering', 3.8, '01-08-2022')
(10, 'Ashley', 'Physics', 3.9, '01-10-2022')
```

CREATE TABLE COURSES

In [ ]:
```
create_query=  """     CREATE TABLE IF NOT EXISTS courses AS
        SELECT 1 AS course_id, "Python programming" AS course_name, 1 AS student_id, "A" AS grade UNION
        SELECT 2, "Data Structures", 2, "B" UNION
        SELECT 3, "Database Systems", 3, "B" UNION
        SELECT 1, "Python programming", 4, "A" UNION
        SELECT 4, "Quantum Mechanics", 5, "C" UNION
        SELECT 1, "Python programming", 6, "F" UNION
        SELECT 2, "Data Structures", 7, "C" UNION
        SELECT 3, "Database Systems", 8, "A" UNION
        SELECT 4, "Quantum Mechanics", 9, "A" UNION
        SELECT 2, "Data Structures", 10, "F";"""
cursor.execute(create_query)
```

Out[ ]:
```
<sqlite3.Cursor at 0x2317270ea40>
```

Q2

1

In [ ]:
```
b1=cursor.execute("SELECt COUNT(DISTINCT(course_id)) FROM courses;")
for row in b1.fetchall():
        print(row)
connect.commit()
```

```
(4,)
```

2

In [ ]:
```
b2=cursor.execute("SELECt COUNT(DISTINCT(s.student_id)) FROM students s,courses c ON s.student_id=c.student_id WHERE s.major='Computer Science' and c.course_name='Python programming';")
for row in b2.fetchall():
        print(row)
connect.commit()
```

```
(2,)
```

3

In [ ]:
```
b3=cursor.execute("SELECt s.name,s.major,s.gpa,c.course_name,c.grade FROM students s,courses c ON s.student_id=c.student_id WHERE c.grade<'C';")
for row in b3.fetchall():
        print(row)
connect.commit()
```

```
('John', 'Computer Science', 3.5, 'Python programming', 'A')
('Samantha', 'Physics', 3.9, 'Python programming', 'A')
('Jane', 'Physics', 3.8, 'Data Structures', 'B')
('Bob', 'Engineering', 3.0, 'Database Systems', 'B')
('Jessica', 'Engineering', 3.8, 'Database Systems', 'A')
('Jacob', 'Physics', 3.4, 'Quantum Mechanics', 'A')
```

Q3

1

```
In [ ]: c1=cursor.execute("SELECt AVG(gpa) FROM students;")
        for row in c1.fetchall():
                print(row)
        connect.commit()
```

```
(3.5800000000000005,)
```

2

```
In [ ]: c2=cursor.execute("SELECT student_id,major,gpa FROM students WHERE gpa IN (SELECT MAX(gpa) FROM students);")
        for row in c2.fetchall():
                print(row)
        connect.commit()
```

```
(4, 'Physics', 3.9)
(10, 'Physics', 3.9)
```

3

```
In [ ]: c3=cursor.execute("SELECT student_id,major,gpa FROM students WHERE gpa IN (SELECT MIN(gpa) FROM students);")
        for row in c3.fetchall():
                print(row)
        connect.commit()
```

```
(3, 'Engineering', 3.0)
```

4

```
In [ ]: c4=cursor.execute("SELECT student_id,major,gpa FROM students WHERE gpa>3.6 and major in ('Physics','Engineering');")
        for row in c4.fetchall():
                print(row)
        connect.commit()
```

```
(2, 'Physics', 3.8)
(4, 'Physics', 3.9)
(5, 'Engineering', 3.7)
(8, 'Engineering', 3.8)
(10, 'Physics', 3.9)
```

5

```
In [ ]: c5=cursor.execute("SELECt AVG(gpa) FROM STUDENTS GROUP BY major;")
        for row in c5.fetchall():
                print(row)
        connect.commit()
```

```
(3.4333333333333336,)
(3.5,)
(3.75,)
```

6

```
In [ ]: query="""WITH rows AS
        (SELECT *,ROW_NUMBER() OVER(PARTITION BY major ORDER BY major ASC,gpa DESC) as position  FROM students)
        SELECT student_id,major,gpa FROM rows WHERE position<3;"""
        c6=cursor.execute(query)
        for row in c6.fetchall():
                print(row)
        connect.commit()
```

```
(6, 'Computer Science', 3.6)
(1, 'Computer Science', 3.5)
(8, 'Engineering', 3.8)
(5, 'Engineering', 3.7)
(4, 'Physics', 3.9)
(10, 'Physics', 3.9)
```

CLOSING THE CONNECTION TO DATABASE

```
In [ ]: connect.close()
```