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 (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 stude
nt_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

In [1]:

```
import sqlite3
import pandas as pd
```

In [2]:

```
connection = sqlite3.connect('college.db')
cursor = connection.cursor()
```

In [3]:

```
#creating the table

sql_command = """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";
    """
cursor.execute(sql_command)
```

Out[3]:

```
<sqlite3.Cursor at 0x7fd7dbd75c00>
```

In [4]:

```
## Q1, part 1
sql_command = """ SELECT* FROM students"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[4]:

	student_id	name	major	gpa	enrollment_date
0	1	John	Computer Science	3.5	01-01-2022
1	2	Jane	Physics	3.8	01-02-2022
2	3	Bob	Engineering	3.0	01-03-2022
3	4	Samantha	Physics	3.9	01-04-2022
4	5	James	Engineering	3.7	01-05-2022
5	6	Emily	Computer Science	3.6	01-06-2022
6	7	Michael	Computer Science	3.2	01-07-2022
7	8	Jessica	Engineering	3.8	01-08-2022
8	9	Jacob	Physics	3.4	01-09-2022
9	10	Ashley	Physics	3.9	01-10-2022

In [5]:

```
##Q1, part 2
sql_command = """SELECT* FROM students
WHERE major = 'Computer Science';
"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[5]:

	student_id	name	major	gpa	enrollment_date
0	1	John	Computer Science	3.5	01-01-2022
1	6	Emily	Computer Science	3.6	01-06-2022
2	7	Michael	Computer Science	3.2	01-07-2022

In [6]:

```
##Q1, part 3
sql_command = """SELECT DISTINCT major
FROM students
ORDER BY major DESC;"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[6]:

	major
0	Physics
1	Engineering
2	Computer Science

In [7]:

```
##Q1, part4
sql_command = """SELECT *
FROM students
WHERE name LIKE '%e%'
ORDER BY gpa ASC;"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[7]:

	student_id	name	major	gpa	enrollment_date
0	7	Michael	Computer Science	3.2	01-07-2022
1	6	Emily	Computer Science	3.6	01-06-2022
2	5	James	Engineering	3.7	01-05-2022
3	2	Jane	Physics	3.8	01-02-2022
4	8	Jessica	Engineering	3.8	01-08-2022
5	10	Ashley	Physics	3.9	01-10-2022

In [8]:

```
sql_command = """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"; """
cursor.execute(sql_command)
```

Out[8]:

```
<sqlite3.Cursor at 0x7fd7dbd75c00>
```

In [9]:

```
#Q2 part1
sql_command = """SELECT COUNT(DISTINCT course_name) FROM courses;
"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[9]:

	COUNT(DISTINCT course_name)
0	4

In [10]:

```
#Q2, part 2
sql_command = """SELECT COUNT(*)
FROM students
LEFT JOIN courses
ON students.student_id = courses.student_id
WHERE students.major = 'Computer Science' AND courses.course_name = 'Python prog
ramming';
"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[10]:

	COUNT(*)
0	2

In [11]:

```
#Q2, part 3
sql_command = """SELECT students.name, students.major, students.gpa, courses.course_name, courses.grade
FROM students
LEFT JOIN courses ON students.student_id = courses.student_id
WHERE courses.grade < 'C';"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[11]:

	name	major	gpa	course_name	grade
0	John	Computer Science	3.5	Python programming	A
1	Samantha	Physics	3.9	Python programming	A
2	Jane	Physics	3.8	Data Structures	B
3	Bob	Engineering	3.0	Database Systems	B
4	Jessica	Engineering	3.8	Database Systems	A
5	Jacob	Physics	3.4	Quantum Mechanics	A

In [12]:

```
#Q3, part 1
sql_command = """SELECT AVG(gpa)
FROM students;"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[12]:

	AVG(gpa)
0	3.58

In [13]:

```
#Q3, part 2
sql_command = """SELECT student_id, major, gpa
FROM students
WHERE gpa = (SELECT MAX(gpa) FROM students);"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[13]:

	student_id	major	gpa
0	4	Physics	3.9
1	10	Physics	3.9

In [14]:

```
#Q3, part 3
sql_command = """SELECT student_id, major, gpa
FROM students
WHERE gpa = (SELECT MIN(gpa) FROM students);"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[14]:

	student_id	major	gpa
0	3	Engineering	3.0

In [15]:

```
#Q3, part 4
sql_command = """SELECT student_id, major, gpa
FROM students
WHERE (major = 'Physics' OR major = 'Engineering') AND gpa > 3.6;"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[15]:

	student_id	major	gpa
0	2	Physics	3.8
1	4	Physics	3.9
2	5	Engineering	3.7
3	8	Engineering	3.8
4	10	Physics	3.9

In [16]:

```
#Q3, part 5
sql_command = """SELECT major, AVG(gpa) as average_of_gpa
FROM students
GROUP BY major;"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[16]:

	major	average_of_gpa
0	Computer Science	3.433333
1	Engineering	3.500000
2	Physics	3.750000

In [17]:

```
#Q3, part 6
sql_command = """ SELECT major, student_id, gpa
FROM (
  SELECT major, student_id, gpa,
         ROW_NUMBER() OVER (PARTITION BY major ORDER BY gpa DESC) AS rank_studen
ts
  FROM students
)
WHERE rank_students <= 2
ORDER BY major, gpa DESC;"""
cursor.execute(sql_command)
sql = pd.read_sql_query(sql_command,connection)
sql
```

Out[17]:

	major	student_id	gpa
0	Computer Science	6	3.6
1	Computer Science	1	3.5
2	Engineering	8	3.8
3	Engineering	5	3.7
4	Physics	4	3.9
5	Physics	10	3.9

In [17]: