Question 1. [7 marks]

Recall this schema, which we have used many times in class. Here we are adding one more relation called Club. It records the university clubs that students have joined (if any). For example, if a student joins the ‘Chess’ club, a tuple would appear in table Club indicating that this student is part of the Chess club.

Relations

- Student(sID, surName, firstName, campus, email, cgpa)
- Course(dept, cNum, name, breadth)
- Offering(oID, dept, cNum, term, instructor)
- Took(sID, oID, grade)
- Club(sID, clubName)

Integrity constraints

- Offering[dept, cNum] ⊆ Course[dept, cNum]
- Took[sID] ⊆ Student[sID]
- Took[oID] ⊆ Offering[oID]
- Club[sID] ⊆ Student[sID]

Part (a) [4 marks]

Write a query to find the sIDs of students at campus ‘UTM’ who are either (a) not part of any club, or (b) have a cgpa that is greater than 3. Use only the basic operators Π, σ, ⊥, ×, ⋃, △, −, ρ, and assignment.

Solution:

- UTMNotInClub(sID) := (ΠsIDσcampus=’UTM’ Student − ΠsIDClub)
- HighGPA(sID) := ΠsIDσcgpa>3∧campus=’UTM’ Student
- Answer(sID) := HighGPA ⋃ UTMNotInClub
Part (b) [3 marks]

Consider the following query:

\[ \text{Apple}(sID) := \Pi_{C1.sID} (\sigma_{C1.sID=C2.sID \land C1.clubName \neq C2.clubName} (\rho_{C1.Club \times C2.Club})) \]

\[ \text{Answer}(sID) := \Pi_{sID} (\text{Apple} \bowtie \sigma_{\text{cgpa} > 80} \text{Student}) \]

You are given below, instances of the relations that are relevant to this query. Add or remove the minimum number of rows to these relations so that student 2222 does appear in the results and students 3333 and 1111 do not.

<table>
<thead>
<tr>
<th>Club</th>
<th>Student:</th>
</tr>
</thead>
<tbody>
<tr>
<td>sID</td>
<td>clubName</td>
</tr>
<tr>
<td>1111</td>
<td>chess</td>
</tr>
<tr>
<td>2222</td>
<td>chess</td>
</tr>
<tr>
<td>1111</td>
<td>dance</td>
</tr>
<tr>
<td>3333</td>
<td>chess</td>
</tr>
<tr>
<td>1111</td>
<td>reading</td>
</tr>
</tbody>
</table>

Solution: 3333 does not appear in the result already. Remove any two rows with sID=1111 from Club relation. Add one row with sID=2222 and any club name to relation Club.
Question 2. [4 marks]

Part (a) [2 marks]
In the schema of question 1, relation Club(sID, clubName) records the university clubs that students have joined (if any). Assuming that the key of this relation is sID (instead of (sID,clubName)), how many clubs can a student join? Circle one answer:

- as many as the number of clubs
- at most one
- at least one

Solution: at most one

Part (b) [2 marks]
Consider this schema:

One(a,b,c)
Two(b,d)

Assuming relation One has 80 tuples and relation Two has 100 tuples, how many tuples could the result of One ⊟◁ Two have? Circle all possible answers.

100 0 80 8000 65

Solution: 0, 65, 80
Question 3.  [6 marks]

Recall this schema, which we have used many times in class. Here we are adding a few more relations:

- **Residence:** all the student residences from all 3 campuses.
- **LivesInRes:** which students live in these residences.
- **AppliedToRes:** the current new applications to live in residence, either from students who do not yet have a spot in a residence, or from those who wish to switch residences. For example, if a student applies for the ‘New College’ residence, then a new application entry will be recorded in ‘AppliedInRes’, mapping the student’s sID to the rID for the residence with rname = ‘New College’.

Relations

<table>
<thead>
<tr>
<th>Relation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student(sID, surName, firstName, campus, email, cgpa)</td>
<td></td>
</tr>
<tr>
<td>Course(dept, cNum, name, breadth)</td>
<td></td>
</tr>
<tr>
<td>Offering(oID, dept, cNum, term, instructor)</td>
<td></td>
</tr>
<tr>
<td>Took(sID, oID, grade)</td>
<td></td>
</tr>
<tr>
<td>Residence(rID, rname, address, campus)</td>
<td></td>
</tr>
<tr>
<td>LivesInRes(sID, rID)</td>
<td></td>
</tr>
<tr>
<td>AppliedToRes(aID, rID, sID)</td>
<td></td>
</tr>
</tbody>
</table>

Please note that:
- It is possible that some students do not live in any residence at all, in any campus (e.g., if they rent their own place off-campus, or live with parents).
- In the Student relation, the ‘campus’ field represents the campus where the student is registered. A student can be registered in one campus, but live in a residence located on a different campus (see the ‘campus’ field from the Residence relation).

**What do you need to do?** Write a query to find all students (sID and surname), who are registered at the ‘UTM’ campus, but who live in a residence on the ‘StG’ campus.

**Solution:**

SELECT * FROM Student
WHERE campus = ‘UTM’
AND sID IN
(SELECT sID FROM LivesInRes L, Residence R
WHERE L.rID = R.rID
AND R.campus = ‘StG’);

Note that multiple other possible solutions exist to this question.
Question 4. [15 marks]

For this question, we will use the same schema from Question 3.

Part (a) [3 marks]

Which of these fields can go in the select list of the query below? Circle only the legal ones.

count(*) avg(cgpa) surName rID sID email

SELECT ---------------
FROM Student S, LivesInRes L
WHERE S.sID = L.sID
GROUP BY rID
ORDER BY rID;

Solution:
Only aggregates or rID are valid.

Part (b) [4 marks]

For this part and all the following parts of Question 4, please assume that no student lives off-campus (each student lives in some residence on campus).

Create a view called CommutingStudent(sID, regCampus, liveCampus), which contains all students who are registered in a different campus than the one they live in (the one where their home residence is located). The fields for this View represent the student’s sID, the campus she is registered in, and the campus she lives in.

Solution:
CREATE VIEW CommutingStudent(sID, regCampus, liveCampus) AS
SELECT S.sID, S.Campus, R.Campus
FROM Student S, Residence R, LiveInResidence L
WHERE S.sID = L.sID AND R.rID = L.rID
AND S.campus <> R.campus;
Part (c) [4 marks]

Create a view called WantsToRelocate(sID), that takes all students from the CommutingStudent view, and finds those among them who have applied to switch residences so they don’t have to commute anymore. To be clear: You must consider those students who have applied only to residences on the campus where they are registered. For example, if Justin lives in a residence at UTSC and is registered on the StG campus, Justin must have applied for residences only on StG, and no applications to any residences at UTSC or UTM.

Solution:

CREATE VIEW WantsToRelocate(sID) AS
SELECT sID
FROM CommutingStudent C
WHERE NOT EXISTS
(SELECT R.campus
FROM CommutingStudent C1, Residence R1, AppliedToResidence A1
WHERE C1.sID = A1.sID AND R1.rID = A1.rID
AND C1.regCampus <> R.campus);

Part (d) [4 marks]

Assume that you have the views in parts b) and c) correctly produced. Using these views, find the sIDs for those commuting students who are happy to continue to commute. To be clear, this means either they did not apply to switch residences at all, or do not have a strong preference to only get a spot on their registered campus.

Solution:

(SELECT sID FROM CommutingStudent) EXCEPT (SELECT * FROM WantsToRelocate);