Do not turn this page until you have received the signal to start.
(In the meantime, please fill out the identification section above, and read the instructions below.)
Question 1.  [11 marks]

Recall this schema, which we have used many times in class. Here we are adding one more relation called Program. It records the subject POSs that students are enrolled in. (“POSt” is short for “program of study”, by the way.)

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>Program(sID, POSt)</td>
<td></td>
</tr>
</tbody>
</table>

Integrity constraints

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

Part (a)  [7 marks]

Write a query to find the sID of every student who either (a) has a POSt that no student from the UTM campus has or (b) has at least two POSts. Use only the basic operators Π, σ, ⋈, ×, ∩, ∪, −, ρ, and assignment.
**Part (b) [4 marks]**

Consider the following query:

\[
\text{One}((\text{dept}, \text{cNum}, \text{term})) := \\
\Pi_{\text{O}_1.\text{dept}, \text{O}_1.\text{cNum}, \text{O}_1.\text{term}} \sigma_{\text{O}_1.\text{dept} = \text{O}_2.\text{dept} \land \text{O}_1.\text{cNum} = \text{O}_2.\text{cNum} \land \text{O}_1.\text{term} < \text{O}_2.\text{term}} (\rho_{\text{O}_1 \text{Offering}} \times \rho_{\text{O}_2 \text{Offering}})
\]

\[
\text{Two}((\text{dept}, \text{cNum}, \text{term})) := (\Pi_{\text{dept}, \text{cNum}, \text{term}} \text{Offering}) - \text{One}
\]

\[
\text{Answer}((\text{instructor})) := \Pi_{\text{instructor}} (\text{Two} \bowtie \text{Offering})
\]

1. Below is an instance of the only relation that is relevant to this query, \text{Offering}. Add the fewest possible rows to \text{Offering} so that professors Able and Bland will not appear in the result of the query, but professors Cranky and Devlish will.

<table>
<thead>
<tr>
<th>oID</th>
<th>dept</th>
<th>cNum</th>
<th>term</th>
<th>instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>O3</td>
<td>CSC</td>
<td>324</td>
<td>1</td>
<td>Bland</td>
</tr>
<tr>
<td>O6</td>
<td>CSC</td>
<td>324</td>
<td>4</td>
<td>Able</td>
</tr>
<tr>
<td>O1</td>
<td>CSC</td>
<td>443</td>
<td>2</td>
<td>Eager</td>
</tr>
<tr>
<td>O2</td>
<td>CSC</td>
<td>324</td>
<td>2</td>
<td>Bland</td>
</tr>
<tr>
<td>O7</td>
<td>CSC</td>
<td>148</td>
<td>3</td>
<td>Devlish</td>
</tr>
<tr>
<td>O9</td>
<td>CSC</td>
<td>148</td>
<td>2</td>
<td>Cranky</td>
</tr>
</tbody>
</table>

2. What does this query compute? Do not describe the steps it takes, only what is in the result, and make your answer general to any instance of the schema.

All instructors who ...
Question 2. [6 marks]

Part (a) [2 marks]

In the previous question, we introduced a new relation called *Program* to record information about students’ POSIs. Does our schema enforce the following constraint:

Every student has at least one POSI.

Circle one answer. If the statement is enforced, say what part of the schema enforces it. If it is not enforced, write an integrity constraint that would enforce it, using the form \( R = \emptyset \).

Enforced This part of the schema enforces it:

Not enforced This new integrity constraint would enforce it:

---

Part (b) [4 marks]

Consider this schema:

\[
\begin{align*}
R(\text{one}, \text{two}, \text{three}) & \quad R[\text{three}] \subseteq T[\text{seven}] \\
S(\text{four}, \text{five}, \text{six}) & \quad S[\text{four}] \subseteq T[\text{seven}] \\
T(\text{seven}, \text{eight}) &
\end{align*}
\]

Suppose relation \( R \) has 100 tuples. How many tuples could \( T \) have? Circle all answers that do not violate the schema.

0 1 82 100 101

Suppose relation \( S \) has 100 tuples. How many tuples could \( T \) have? Circle all answers that do not violate the schema.

0 1 82 100 101
Question 3. [5 marks]

The question refers to the schema from Question 1. Write a query in SQL to find the maximum grade given in every course, across all offerings. For each, report the name of the department, the course number, and the maximum grade given in that course in any offering of that course.
Question 4. [8 marks]

Part (a) [3 marks]

Consider the same schema from the Question 1. Suppose we wrote the query

```
SELECT ________________________
FROM Student, Took
WHERE Student.sID = Took.sID
GROUP BY Took.sID;
```

Which of the following could go in the SELECT clause? Circle all that apply.

- sID
- oID
- count(oID)-2
- Took.sID
- avg(grade)
- max(cgpa)
- cgpa

Part (b) [3 marks]

We discussed in lecture how the SQL subquery operators could possibly be implemented using other SQL operations. Suppose we have two tables \( R(a, b) \) and \( S(b, c) \). Note that their keys are underlined.

Consider the following two queries:

```
-- Query 1
SELECT a AS answer
FROM R
WHERE b > SOME (SELECT b FROM S);
```

```
-- Query 2
SELECT R.a AS answer
FROM R, S
WHERE R.b > S.b;
```

On the next page, give a database instance where these two queries produce different results, and the results of the two queries.
Instance of the database where the two queries produce different results:

Result of Query 1 on this instance:  
Result of Query 2 on this instance:

Part (c)  [2 MARKS]
Without changing the FROM clause, fix Query 2 so that it produces the same results as Query 1 for any valid dataset. Do not use subqueries or views.
This page is left (mainly) blank for things that don’t fit elsewhere.

# 1: _____/11
# 2: _____/ 6
# 3: _____/ 5
# 4: _____/ 8

TOTAL: _____/30