CSE 180, Final Exam, Fall 2021, Shel Finkelstein

Student Name: _____

Student ID: _____

UCSC Email: _____

Part	Max Points	Points
I	40	
II	24	
III	36	
Total	100	

The first Section (Part I) of the Fall 2021 CSE 180 Final is multiple choice and is double-sided. Answer all multiple choice questions on your Scantron sheet. You do not have to hand in the first Section of the Exam, but you must hand in the Scantron sheet, with your name, email and student id on that Scantron sheet. Please be sure to use a #2 pencil to mark your choices on this Section of the Final.

This separate second Section (Parts II and III) of the Final is not multiple choice and is single-sided, so that you have extra space to write your answers. If you use that extra space, please be sure to write the number of the problem that you're solving next to your answer. Please write your name, email and student id on this second Section of the Exam, which you must hand in. You may use any writing implement on this Section of the Exam.

At the end of the Final, please be sure to hand in your **Scantron sheet** for the first Section of the Exam and also **this second Section of the Exam**, and show your **UCSC id** when you hand them in. Also hand in your **8.5 x 11 "cheat sheet"**, with your name written in the upper right corner of the sheet.

Part II: (24 points, 6 points each)

Question 21: Here are statements creating two table S and T:

```
CREATE TABLE S (
   c INT PRIMARY KEY,
   d INT
   );
```

```
CREATE TABLE T (
   a INT PRIMARY KEY,
   b INT,
   FOREIGN KEY(b)
         REFERENCES S(c)
            ON UPDATE CASCADE
   );
```

Relation S(c,d) currently contains the four tuples, (2,60), (3,61), (4,62), and (5,63).
Relation T(a,b) currently contains the four tuples, (0,4), (1,5), (2,4), and (3,5).

Indicate all changes that happen to both S and T when each of the following SQL statements is executed upon the S and T instances that are shown above. That is, you should ignore changes made by earlier parts of this question when you answer later parts.

If a statement changes existing tuples in a relation, show both old and new values of those tuples. If a statement makes no changes to a relation, say that.

21a): UPDATE S set c=8 WHERE d=62;
Answer 21a):
Changes made to the original S:

Changes made to the original T:

21b): DELETE FROM T WHERE a=2;
Answer 21b):
Changes made to the original S:

Changes made to the original T:

21c): DELETE FROM S WHERE c=3;
Answer 21c):
Changes made to the original S:

Changes made to the original T:

Question 22: This question has two parts; be sure to answer both of them.

Assume that we have the following table in our database:

Employees(empNumber, empName, dept, salary)

In which empNumber is the Primary Key.

22a): Write a SQL statement that updates the salaries of all the employees who make more than the average salary, giving them a 12% raise. For example, the salary for an employee who makes more than the average salary and whose salary is 200 would become 224 after the update.

Answer 22a):

22b): Do you agree or disagree with the following statement? Justify your answer.

If the statement in part a) of this question is executed, the result depends on the order in which the tuples in Employees are processed, because average salary changes each time a tuple is updated.

Answer 22b):

Question 23: We have the following relations:

Sailors(sid, sname, rating, age)
    // sailor id, sailor name, rating, age
Boats(bid, bname, color)
    // boat id, boat name, color of boat
Reserves(sid, bid, day)
    // sailor id, boat id, and date that sailor sid reserved that boat.

Codd's Relational Algebra included only 5 operators, (σ , π , x , U , and -). Write the following query using Codd's Relational Algebra (not SQL):

> *Find the sname of Sailors whose age is 21 and who have a reserved a red boat and have also reserved a blue boat.*

If you'd like, you may also use Rename (ρ) and Assignment ($\leftarrow$) in your answer.

To simplify notation, you may write SIGMA for σ and PI for π. You may also use <-- for Assignment and RHO for ρ (Rename). Also, although you can use subscripts, you can also use square brackets, for example writing:

PI[Boats.bid] (SIGMA[Boats.bname ='Titanic'] (Boats))

instead of writing:

$\pi_{\text{Boats.bid}} (\sigma_{\text{Boats.bname ='Titanic'}} (\text{Boats}))$

Answer 23:

Question 24: This question has two parts; be sure to answer both of them.

24a): You have a relation Departments(deptName, building, floor, manager), which has just the following non-trivial Functional Dependencies, no others:

deptName → building
manager, building → floor
building, floor → deptName
building, floor → manager

Is the Departments relation in 3NF? Justify your answer clearly.

Answer 24a)

24b): In Design Theory, what is the Update Anomaly? In your answer, explain why it is a problem by giving a clear example.

Answer 24b)

Part III: (36 points, 9 points each)

Some familiar tables appear below, with Primary Keys underlined. **These tables also appear on the last page of the Final, which you can tear off to help you do questions in Part III of the Final.** You don't have to turn in that last page at the end of the Exam.

Persons(personID, lastName, firstName, email, affiliation, isStudent)

Conferences(conferenceName, year, conferenceDate, regularAttendeeCost, studentAttendeeCost, submissionDueDate, reviewDueDate, importance)

Submissions(conferenceName, year, submissionID, numPages, submitDate, wasAccepted, submissionTitle, dateAccepted, datePublished)

Authors(authorID, conferenceName, year, submissionID, authorPosition)

Reviewers(reviewerID, conferenceName, year, reliability)

Reviews(reviewerID, conferenceName, year, submissionID, reviewDate, rating)

Attendees(attendeeID, conferenceName, year)

The Primary Key in each table is shown underlined. Assume that there aren't any UNIQUE or NOT NULL constraints specified for these tables. Data types aren't shown to keep thing simple. There aren't any trick questions about data types.

You should assume Foreign Keys as follows:

- Every (conferenceName, year) in Submissions appears as a Primary Key in Conferences.
- Every authorID in Authors appears as a Primary Key in Persons.
- Every (conferenceName, year) in Authors appears as a Primary Key in Conferences.
- Every reviewerID in Reviewers appears as a Primary Key in Persons.
- Every (conferenceName, year) in Reviewers appears as a Primary Key in Conferences.
- Every (reviewerID, conferenceName, year) in Reviews appears as a Primary Key in Reviewers.
- Every (conferenceName, year, submissionID) in Reviews appears as a Primary Key in Submissions.
- Every attendee in Attendees appears as a personID in Persons.
- Every (conferenceName, year) in Attendees appears as a Primary Key in Conferences.

Write legal SQL queries for Questions 25-28. If you want to create and then use views to answer these questions, that's okay, but views are not required unless the question asks for them.

Don't use DISTINCT in your queries unless it's necessary, 1 point will be deducted if you use DISTINCT when you don't have to do so. And some points may be deducted for queries that are very complicated, even if they are correct.

Question 25: Find all the reviewers who are students, who wrote a review whose review date is December 7, 2021 and whose reliability **(associated with the conference for which they wrote that December 7, 2021 review)** isn't NULL. (isStudent is a Boolean attribute in the Persons table which indicates whether the person is a student.)

The attributes in the result should be the first name of the Reviewer, the last name of the reviewer and the reliability of the reviewer. These attributes should appear as theFirstName, theLastName and theReliability.

No duplicates should appear in your result.

Answer 25:

Question 26: Find the authors of submissions whose datePublished is after August 26, 1990 and whose highest review rating **for that submission** is greater or equal to 3. The attributes in your result should be authorID, datePublished, submissionTitle and authorPosition.

The tuples in your result should be in alphabetical order based on submissionTitle; tuples that have the same submissionTitle should appear with bigger authorPosition values appearing before smaller authorPosition values.

No duplicates should appear in your result.

Answer 26:

Question 27: Find all the conferences whose conferenceName has the string SIG anywhere in it (uppercase), and for which no submissions were accepted. Use the Boolean attribute wasAccepted to determine if a submission was accepted.

The attributes in your result should be conferenceName, year and the number of submissions to the conference, which should appear as theConferenceName, theConferenceYear and theSubmissionCount. (Yes, you should count all the submissions, even though none of them was accepted.)

No duplicates should appear in your result.

Answer 27:

Question 28: This question has two parts; be sure to answer both.

Sometimes a person can be an author, a reviewer and also an attendee, all for the same conference. We'll say that such a person is a "triply-involved" person for that conference. A person might be triply-involved in more than one conference.

28a): Create a view called TriplyInvolved that finds all triply-involved persons. The attributes in your result should be the ID of a triply involved person and conferenceName and year for which that person is triply involved.

No duplicates should appear in your view.

28b): Write a query that uses the TriplyInvolved view to determine triplyInvolvedCount for each conference, the number of people who are triply-involved in that conference. The attributes in your result should be conferenceName, year and triplyInvolvedCount. In your result, include only the conferences for which there is at least one triply involved person.

Be sure to use the TriplyInvolved view to do this; you may also use the original tables, if necessary.

No duplicates should appear in your result.

Answers 28a) and 28b):