CSE 182 Midterm Exam, Spring 2022, Shel Finkelstein

Student Name: _____

Student ID: _____

UCSC Email: _____

ANSWERS

Midterm Points:

Part	Max Points
I	30
II	20
III	24
IV	27
Total	101

Closed book, but it's okay to bring a single two-sided 8.5" x 11" sheet of paper with as much info written on it as you can fit and read unassisted. Please hand in Midterm and your sheet of paper (with name on top right) when you finish the Midterm. You must also show your UCSC ID when you hand in the Midterm.

Be sure to answer each question readably in the space provided.

Part I (30 Points, 6 each): Questions 1-5 are about the instance of the table Scores that you have been given as a separate piece of paper.

Do not turn in that page when you hand in your Midterm!

What is the result of each of the following SQL queries?

Be sure to show attribute names at the top of all SQL outputs!

Question 1:

```
SELECT DISTINCT Team
FROM Scores
WHERE Runs <= 4;
```

Answer 1:

```
Team
----------
Dragons
Carp
Bay Stars
Giants
Swallows
```

Tuples may appear in any order, since there's no ORDER BY clause. But no tuple can appear twice, since it's SELECT DISTINCT.

The underline below Team is not required.

Question 2:

```
SELECT Day, Team
FROM Scores
WHERE Team LIKE '%ar%'
ORDER BY Day, Team DESC;
```

Answer 2:

```
Day       | Team
--------------------------
Monday |  Carp
Monday |  Bay Stars
Sunday   |  Carp
Sunday   |  Bay Stars
```

Tuples must appear in this order because of the ORDER BY.

The underline below the attributes and the "|" separator are not required.

Question 3:

```
SELECT S1.Day, S1.Team, S1.Runs
FROM Scores S1
WHERE S1.Runs <= ALL ( SELECT S2.Runs
                       FROM Scores S2
                       WHERE S1.Day = S2.Day );
```

Answer 3:

Day	Team	Runs
Sunday	Carp	1
Sunday	Bay Stars	1
Monday	Swallows	0

Tuples may appear in any order, since there's no ORDER BY clause.

Query finds the Day, Team and Runs where that team had the smallest number of runs of that day. Note that there are two tuples for Sunday.

Okay to have S1.Day, S1.Team and S1.Runs as the attributes.

The underline below the attributes and the "|" separator are not required.

Question 4:

```
SELECT Team, SUM(Runs) AS RunTotal
FROM Scores
GROUP BY Team;
```

Answer 4:

```
Team        | RunTotal
---------------------
Dragons     | 10
Tigers      | 13
Carp        | 4
Swallows    | 7
Bay Stars   | 10
Giants      | 9
```

Tuples may appear in any order, since there's no ORDER BY clause.

Query finds the total number of runs for each team.

The underline below the attributes and the "|" separator are not required.

Question 5:

Write a SQL statement that changes the Scores table so that all tuples in which the Opponent is 'Swallows' are deleted.

This statement should work for any instance of the Scores table, not for the instance that you've been given on the last page of the Midterm.

Answer 5:

```
DELETE FROM Scores
WHERE Opponent = 'Swallows';
```

Okay to use different legal spacing. For example, you could have entire statement on one line.

No deduction for different capitalization, except for the value 'Swallows', which needed to be capitalized that way.

No deduction if semi-colon is omitted.

Part II (20 points, 4 each): The questions in PART II are all about a table Employees that was created as follows:

```
CREATE TABLE Employees (
        name        CHAR(30) PRIMARY KEY,
        age         INTEGER NOT NULL DEFAULT(21),
        salary      INTEGER,
        department  CHAR(20) NOT NULL
    );
```

Answer **YES** or **NO** to each question in Part II, writing out the full word.

Question 6: In an instance of Employees there can't be two different tuples that have identical values for both name and salary.

Answer 6: ___**YES**___ **name is the Primary Key.**

Question 7: Two SQL queries are Equivalent if they always have the same result on any legal database.

Are the following two SQL queries Equivalent?

```
SELECT COUNT(*)
FROM Employees;
```

```
SELECT COUNT(age)
FROM Employees;
```

Answer 7: ____**YES**___ **age can't be NULL.**

Question 8: Is the following a legal SQL query?

```
SELECT department, MAX(age), MAX(salary)
FROM Employees
WHERE salary > 8000
GROUP BY department;
```

Answer 8: ___**YES**___

Question 9: Does the following SQL query output the names of all Employees whose age is 32 and whose salary is NULL?

```
SELECT e.name
FROM Employees e
WHERE e.age = 32
    AND e.salary = NULL;
```

Answer 9: ___**NO**____ Should be e.salary IS NULL

Question 10: Are the following two SQL queries Equivalent?

```
SELECT e.name
FROM Employees e
WHERE e.age NOT IN ( SELECT e2.age
                     FROM Employees e2
                     WHERE department = 'Sales' );

SELECT e.name
FROM Employees e
WHERE e.age != ANY ( SELECT e2.age
                     FROM Employees e2
                     WHERE department = 'Sales' );
```

Answer 10: ___**NO**____ "NOT IN" can be replaced by "!= ALL"

Part III (24 points, 6 each): Answer questions 11-14.

Question 11: If R(A,B) is a relation where A's domain is (a1, a2, a3) and B's domain is (b1, b2, b3, b4, b5), what the maximum number of different tuples that can be in an instance of R, assuming that A can be NULL, but B can't be NULL?

Just supply your answer; no part credit.

Answer 11: ___**20**___ 4 * 5, since A can be NULL and B can't be NULL

Question 12: Let S(A,B,C) be a relation where A is the primary key for S, and no attribute can be NULL. Suppose that A's domain has 5 different values, B's domain has 2 different values, and C's domain has 6 different values. What is the maximum number of different tuples that can be in an instance of S?

Just supply your answer; no part credit.

Answer 12: ___**5**___ Values of Primary Key can't appear twice

Question 13: We discussed the ACID properties for transactions. The letter "A" in ACID stands for Atomicity. Briefly explain what Atomicity means. Your answer should provide an explanation, not just a phrase.

Answer 13:

Atomicity for a transaction means that all of the database operations in that transaction are performed, or none of the database operations are performed. This is often referred to as the "all-or-nothing" property, but you have to say what that means, not just write that phrase.

Question 14: SQL uses 3-valued logic, with values TRUE, FALSE and UNKNOWN. What is the value of each of the following three expressions?

14a): FALSE AND UNKNOWN

Answer 14a): ____**FALSE**___

14b): UNKNOWN OR TRUE

Answer 14b): ____**TRUE**____

14c): TRUE AND NOT UNKNOWN

Answer 14c): __**UNKNOWN**___

Part IV (27 points, 9 each):

The questions in Part IV ask you to write SQL statements using the tables shown below, which are 3 of the tables in our Lab Assignments.

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

RacingPersons(personID, personName, registryDate, canBeJockey, canBeTrainer)

Stables(stableID, stableName, address, stableOwnerID)

Horses(horseID, horseName, horseBreed, birthDate, stableID, trainerID, horseOwnerID)

You should assume the following Foreign Keys:

- Every stableOwnerID in Stables appears as a personID in RacingPersons.
- Every stableID in Horses appears as a stableID in Stables.
- Every trainerID in Horses appears as a personID in RacingPersons.
- Every horseOwnerID in Horses appears as a personID in RacingPersons.

Tables and attributes are repeated at the top of each question, with Primary Keys underlined.

RacingPersons(personID, personName, registryDate, canBeJockey, canBeTrainer)
Stables(stableID, stableName, address, stableOwnerID)
Horses(horseID, horseName, horseBreed, birthDate, stableID, trainerID, horseOwnerID)

Question 15: For each horse whose name isn't NULL and who does not have 'Mr' (with that capitalization) as the first two characters in its name, find its horseID, its horseName, and the name of its trainer (which should appear in your result as trainerName).

No duplicates should appear in your result.

Answer 15:

```
SELECT h.horseID, h.horseName, p.personName AS trainerName
FROM Horses h, RacingPersons p
WHERE h.horseName IS NOT NULL
    AND h.horseName NOT LIKE 'Mr%'
    AND h.trainerID = p.personID;
```

Okay to have SELECT DISTINCT in CSE 182, although there can't be any duplicates even if you don't write SELECT DISTINCT.

Okay to have different tuple variables, or use names of tables instead of tuple variable, or omit tuple variable when there is no ambiguity (and there is no ambiguity for any attributes in this question).

Okay to use different legal spacing.

Okay to write NOT (h.horseName LIKE 'Mr%')

No deduction if semi-colon is omitted.

RacingPersons(personID, personName, registryDate, canBeJockey, canBeTrainer)
Stables(stableID, stableName, address, stableOwnerID)
Horses(horseID, horseName, horseBreed, birthDate, stableID, trainerID, horseOwnerID)

Question 16: horseBreed and stableID are attributes in the Horses table. stableID identifies the stable that the horse is in. A horse is a Quarterhorse if its horseBreed value equals 'Q'.

For each stable that has no Quarterhorses in it, output the name of the stable, the name of the owner of the stable, and the address of the stable. These attributes should appear as stableName, stableOwner and stableAddress

The tuples in your result should be in reverse alphabetical order based on stableName. No duplicates should appear in your result.

Answer 16:

Here are multiple correct solutions. Comments appear after them.

```
SELECT DISTINCT s.stableName, p.personName AS stableOwner,
                        s.address AS stableAddress
FROM Stables s, RacingPersons p
WHERE p.personID = s.stableOwnerID
    AND NOT EXISTS ( SELECT *
                            FROM Horses h
                            WHERE h.horseBreed = 'Q'
                                AND h.stableID = s.stableID )
ORDER BY s.stableName DESC;
```

Q16 Answer (continued):

```
SELECT DISTINCT s.stableName, p.personName AS stableOwner,
                s.address AS stableAddress
FROM Stables s, RacingPersons p
WHERE p.personID = s.stableOwnerID
    AND s.stableID NOT IN ( SELECT h.stableID
                            FROM Horses h
                            WHERE h.horseBreed = 'Q' )
ORDER BY s.stableName DESC;
```

"NOT IN" in above solution could be replaced by "!= ALL"

DISTINCT is needed, since there could potentially be multiple stables with the same name that are owned by people with the same name and which have the same address..

Okay to have different tuple variables, or use names of tables instead of tuple variable, or omit tuple variable when there is no ambiguity (and there is no ambiguity for any attributes in this question).

Okay to use different legal spacing.

Okay to write NOT (h.horseName LIKE 'Mr%')

DESC is needed in the ORDER BY for reverse alphabetical order.

Writing horseBreed LIKE 'Q' instead of horseBreed = 'Q"is correct, but will get a minor deduction since use of LIKE is over complicated when there no pattern.

No deduction if semi-colon is omitted.

RacingPersons(personID, personName, registryDate, canBeJockey, canBeTrainer)
Stables(stableID, stableName, address, stableOwnerID)
Horses(horseID, horseName, horseBreed, birthDate, stableID, trainerID, horseOwnerID)

Question 17: horseBreed is an attribute in the Horses table that identifies the breed of the horse.

Find the breeds for which at least 5 horses of that breed were born on or after the date December 29, 2018. For each such breed, output the horseBreed, (which should appear as breed) and the number of horses who were born on or after that date (which should appear as breedTotal).

No duplicates should appear in your result.

Answer 17:

```
SELECT h.horseBreed, COUNT(*) AS breedTotal
FROM Horses h
WHERE h.birthDate >= DATE '2018-12-29'
GROUP BY h.horseBreed
HAVING COUNT(*) >= 5;
```

DISTINCT is not needed, since there can't be more than on group for a horseBreed. But there won't be a deduction in CSE 182 if you use DISTINCT unnecessarily.

Okay to have different tuple variables, or use names of tables instead of tuple variable, or omit tuple variable when there is no ambiguity (and there is no ambiguity for any attributes in this question).

Okay to use different legal spacing.

Instead of DATE '2018-12-29' it's also okay to write:
DATE '12/29/2018' or DATE '12/29/18'

Instead of COUNT(*) in the HAVING clause, it would be okay to have COUNT(a), where a is any attribute that can't be NULL. horseID can't be NULL, since it's the Primary Key of Horses, and birthDate can't be NULL for tuples that satisfy the WHERE clause predicate. But all other attributes of Horses, including horseBreed, could be NULL.

Can't say HAVING breedTotal >= 3 in the HAVING clause, because breedTotal is defined in the SELECT clause.