CSE 180, Final Exam, Fall 2019, Shel Finkelstein

Multiple Choice Questions (Part I)
Answered on Scantron Sheet

Test Form letter: A

This first Section (Part I) of the Fall 2019 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 this 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.

Name and Student ID must also be filled in by shading letters/numbers on the form. You must also mark the **version** ("Test Form letter **A**") of the Multiple Choice section that you took. The box for Test Form letter is at the top of the Scantron sheet, just to the left of the Multiple Choice questions.

The 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 the 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 **both your Scantron sheet for this first Section of the Exam** and also the separate second Section of the Exam. You must also show your UCSC id when you hand them in.

Part I: (40 points, 2 points each)

Answer the questions in Part I on your Scantron sheets, which should have your name, email and UCSC id on them. Select the **best answer** for each of the following. For some questions, a choice is **"All of the Above"** or "**None of the Above"**, so be sure to read answer choices carefully.

Question 1: Which operation can be performed using an ALTER statement?

a) Create a table
b) Drop a table
c) Delete all rows from a table
d) Add additional columns to a table
e) None of the Above

Question 2: If an instance of relation R1(A,B,C) has 10 different tuples in it, and an instance of relation R2(B,C,D) has 8 different tuples in it, then how many tuples will there be in the result if the following SQL query is executed on those instances?

```
SELECT *
FROM R1, R2
WHERE R1.B = R2.B AND R1.C = R2.C;
```

a) 0
b) Exactly 18
c) Exactly 80
d) Between 0 and 18. Could be 0 or 18, but could be values other than 0 and 18.
e) Between 0 and 80. Could be 0 or 80, but could be values other than 0 and 80.

Question 3: We discussed the ACID properties of transactions. What does Atomicity (the "A" in ACID) mean for transactions?

a) Transaction execution is as if transactions were executed one at a time.
b) Transactions happen completely or not-at-all.
c) If a transaction commits, its changes are permanent, even if there are failures, although they can be changed by later transactions.
d) Business rules are always maintained by the database system.
e) Uncommitted (dirty) values from one transaction are never read by any other transaction.

Question 4: Assume that r is a relation, *attribs* is some attributes of r, and *cond* is a condition on r. When will the following equation be true?

$$\sigma_{cond} (\pi_{attribs} (r)) = \pi_{attribs} (\sigma_{cond} (r))$$

a) It will always be true.
b) It will be true only when r is a binary relation, with two attributes.
c) It will be true only when there are two attributes in *attribs*.
d) It will be true only when the conditions in *cond* refer just to attributes that are in *attribs*.
e) None of the Above.

Question 5: An instance of relation *r(A,B)* has m tuples in it, all exactly the same, and an instance of relation *s(A,B)* has n tuples in it, and all of those tuples are exactly the same as the tuples in *r(A,B)*. If r and s are Union-Compatible, then how many tuples are there in the result of the following query?

```
( SELECT * FROM r )
    INTERSECT ALL
( SELECT * FROM s );
```

a) 1
b) m + n
c) m – n, as long as m ≥ n, otherwise 0
d) min(m, n)
e) max(m, n)

Question 6: Let R(A,B,C,D) be a relation, where (A, B) is the Primary Key, C can be NULL, and D is UNIQUE, and D also can't be NULL.

Assume that A's domain has 4 different values, B's domain has 5 different values, C's domain has 7 different values, and D's domain has 11 different values. What is the maximum number of different tuples that can be in any instance of R?

a) 7
b) 8
c) 11
d) 12
e) 20

Question 7: Which statement is FALSE for Relational Algebra Operations, where R is a relation and C1 and C2 are conditions on the attributes of R?

a) $\sigma_{C1}(\sigma_{C2} (R)) = \sigma_{C1 \text{ AND } C2}(R)$
b) Intersection is Commutative (for union-compatible relations): $R \cap S = S \cap R$
c) Union is Associative (for union-compatible relations): $(R \cup S) \cup T = R \cup (S \cup T)$
d) Minus is Associative (for union-compatible relations): $(R - S) - T = R - (S - T)$
e) Product distributes over union, assuming that S and T are union-compatible relations: $R \times (S \cup T) = (R \times S) \cup (R \times T)$

Question 8: Which of these is/are advantages of Stored Procedures?

a) Stored Procedures can make performance better by enabling operations to be performed on data without moving that data from the database to the client.
b) Stored Procedures can make programming easier by allowing code to be written once in a procedure, and then reused by anyone authorized to execute the procedure.
c) Stored Procedures can improve security by allowing people to be authorized to execute a procedure without authorizing them to access all the data used by the procedure.
d) All of the Above
e) None of the Above

Question 9: What difference between FALSE and UNKNOWN in the three-valued logic used by SQL is the major justification for the need for UNKNOWN?

(The correct answer must both be correct, and also be the major justification for the need for UNKNOWN.)

a) FALSE is the same as NULL, but UNKNOWN isn't.
b) When you execute a SQL query, if the result for some tuples is FALSE, those tuples don't contribute to the answer, but if the result is UNKNOWN, then they do contribute to the answer.
c) The negation of FALSE is TRUE, but the negation of UNKNOWN is UNKNOWN.
d) TRUE OR UNKNOWN is UNKNOWN, but TRUE or FALSE is TRUE.
e) FALSE AND UNKNOWN is FALSE, but FALSE OR UNKNOWN is UNKNOWN.

Question 10: Here's a query Q on the relation Employees(name, age, salary):

```
Q:      SELECT E1.name
        FROM Employees E1
        WHERE EXISTS
                ( SELECT E2.salary
                  FROM Employees E2
                  WHERE E1.salary = E2.salary AND E2.age >= 65 );
```

And here are 3 other queries, which are different only on the line shown in **bold**:

```
Q1:     SELECT E1.name
        FROM employee E1
        WHERE E1.salary =
                ( SELECT E2.salary
                  FROM employee E2
                  WHERE E2.age >= 65 );
```

```
Q2:     SELECT E1.name
        FROM employee E1
        WHERE E1.salary IN
                ( SELECT E2.salary
                  FROM employee E2
                  WHERE E2.age >= 65 );
```

```
Q3:     SELECT E1.name
        FROM employee E1
        WHERE E1.salary = ANY
                ( SELECT E2.salary
                  FROM employee E2
                  WHERE E2.age >= 65 );
```

Which query (or queries) are equivalent to Q?

a) Just Q1
b) Just Q2
c) Just Q3
d) Just Q2 and Q3
e) Q1, Q2 and Q3 are all equivalent to Q

Question 11: student(student_id, name, address, age, major) is a table where student_id is the Primary Key. If (5678, 'Kylo Ren', 'Solo Place', 21, 'CMPS') is a tuple in that table, and the following is executed, with no other work going on:

```
BEGIN TRANSACTION;

UPDATE Students
SET age = age + 1
WHERE name = 'Kylo Ren';

UPDATE Students
SET major = 'CSE'
WHERE age = 21;

ROLLBACK TRANSACTION;
```

then afterwards, what will be in the tuple that has student_id 5678?

a) (5678, 'Kylo Ren', 'Solo Place', 21, 'CMPS')
b) (5678, 'Kylo Ren', 'Solo Place', 22, 'CMPS')
c) (5678, 'Kylo Ren', 'Solo Place', 21, 'CSE')
d) (5678, 'Kylo Ren', 'Solo Place', 22, 'CSE')
e) None of the Above

Question 12: For a very large table student(<u>ID</u>, name, dept_name, tot_cred), which index will probably <u>help the most</u> for executing the following query?

```
SELECT ID, name, tot_cred
FROM student
WHERE dept_name = 'BIO' AND tot_cred > 60
```

a) An index on dept_name
b) An index on tot_cred
c) An index on (dept_name, tot_cred) in that order
d) An index on (tot_cred, dept_name) in that order
e) Indexes on (dept_name, tot_cred) and (tot_cred, dept_name) are <u>both</u> the most helpful, and they're equally good.

Question 13: Assume that the employee table has been created as follow:

```
CREATE TABLE employee (
        name   CHAR(20) PRIMARY KEY,
        age    INTEGER NOT NULL DEFAULT 18,
        salary INTEGER NOT NULL,
        CHECK (age < 21 OR salary >= 8000)
 );
```

Which INSERT statement (or statements) will result in an error? Be sure to give the best answer.

a) INSERT INTO *employee*(*name*, *salary*) VALUES ('Alpha', 9000);
b) INSERT INTO *employee*(*name*) VALUES ('Beta');
c) INSERT INTO *employee* VALUES ('Gamma', 30, 6500);
d) Answers a), b) and c) will all result in errors.
e) Answers b) and c) will result in errors, but answer a) will not.

Question 14: *sells(bar, beer, price)* and *beers(name, manufacturer)* are tables in a database schema, with Primary Key underlined. You look at a million different instances of the database. Here are two statements:

i. *sells.beer* must be a Foreign Key corresponding to *beers.name* if (for all million instances) every beer in *sells.beer* also appears in *beers.name*
ii. *beers.name* must be a Foreign Key corresponding to *sells.beer* if (for all million instances) every beer in *beers.name* also appears in *sells.beer*

a) Both statements are True.
b) The first statement is True and the second statement is False.
c) The first statement is False and the second statement is True.
d) Both statements are False.
e) All of the Above.

Question 15: Identify the statement about OLAP that is FALSE.
[Choose e if choices a, b, c and d are all correct.]

a) There must be a row in the Fact table for every combination of Dimension table values.
b) Fact tables may have multiple Measure attributes.
c) In a Fact table, Dimension attributes are Foreign Keys referring to the Dimension tables.
d) Roll-up of a Fact table corresponds to performing SQL GROUP BY and aggregation.
e) None of the above statements are False.

Question 16: What is a difference between JSON and the Relational Model?

a) JSON documents are not in First Normal Form, because JSON allows arrays and nesting, but the Relational Model requires First Normal Form.
b) Relational tables are not in First Normal Form, because relational tables allow arbitrarily many appearances of elements in a document, but JSON requires First Normal Form.
c) Relational requires that data be in Boyce-Codd Normal Form, but JSON does not.
d) Relational requires that data be in Third Normal Form, but JSON does not.
e) None of the Above.

Question 17: Why is a reason (or reasons) that having a relational schema in Boyce-Codd Normal Form (BCNF) may be better than <u>not</u> having a schema in BCNF?

a) BCNF schemas reduce redundancy.
b) BCNF schemas avoid the Update Anomaly.
c) BCNF schemas avoid the Insert Anomaly.
d) BCNF schemas avoid the Delete Anomaly.
e) All of the Above.

Question 18: *r(A,B)* and *s(A,B)* are Union-Compatible tables which may have duplicates. Here are two SQL queries, Q1 (on the left) and Q2 (on the right) on those tables. Which statement is always correct for the results of those queries?

```
( SELECT DISTINCT *
  FROM r
  WHERE A > 17 )

 UNION

( SELECT DISTINCT *
  FROM s
  WHERE B < 500 );
```

```
( SELECT *
  FROM r
  WHERE A > 17 )

 UNION ALL

( SELECT *
  FROM s
  WHERE B < 500 );
```

a) The same tuples will appear in the results of Q1 and Q2, but Q1 may have duplicates.
b) The same tuples will appear in the results of Q1 and Q2, but Q2 may have duplicates.
c) There may be some tuples in the result of Q1 that do not appear even once in the result of Q2.
d) There may be some tuples in the result of Q2 that do not appear even once in the result of Q1.
e) None of the Above.

Question 19: What are Armstrong's Axioms?

a) Transformations which relational systems use to optimize a query.
b) The 5 relational operators that Ted Codd used to define Relational Algebra.
c) Rules for generating functional dependencies from other functional dependencies.
d) An algorithm to determine whether a decomposition is a Lossless Join decomposition.
e) None of the Above.

Question 20: *sells(bar, beer, price)* is a table, where *bar* and *beer* are CHAR(20) and *price* is FLOAT. Assume that myCon is a JDBC connection. What's the error in the following JDBC, which is supposed to print the beers and prices for Joe's Bar?

```
Statement stmt = myCon.createStatement();
ResultSet Menu =
      stmt.executeQuery(" SELECT beer, price FROM sells WHERE bar = 'Joe''s Bar' ");
while (Menu.fetch()) {
  // For each value in result, get values of beer and price, and print them
  System.out.println(Menu.getString(1), Menu.getFloat(2));
```

a) Should use *PreparedStatement*, not *Statement*
b) Should use *Menu.next*, not *Menu.fetch*
c) Should use *System.out.println(ResultSet.Float(1), ResultSet.getString(2));* not *System.out.println(Menu.getString(1), Menu.getFloat(2));*
d) Should use *stmt.updateQuery,* not *stmt.executeQuery*
e) In the SELECT statement, 'Joe''s Bar' has the single quote symbol appearing twice (between Joe and s) in 'Joe''s Bar', but the quote symbol should only appear once, replaced by 'Joe's Bar'.

Question 21: *sells(bar, beer, price)* is a table, and *ripoffBars(bar)* is another table. What does this Trigger do, assuming that *price* represents a dollar amount?

```
CREATE TRIGGER PriceTrig
AFTER UPDATE OF price ON sells
REFERENCING
        OLD ROW AS old_row
        NEW ROW AS new_row
FOR EACH ROW
WHEN(new_row.price – old_row.price > 6.00)
INSERT INTO ripoffBars
        VALUES(new_row.bar);
```

a) If any row in *sells* has a beer whose *price* is more than 6 dollars, then the *bar* in that row is inserted into *ripoffBars*.
b) If a row is inserted into *sells* that has a beer whose *price* is more than 6 dollars, then the *bar* in that inserted row is inserted into *ripoffBars*.
c) If *price* is updated for a row in *sells*, and the new price is more than 6 dollars higher than the old price, then the *bar* in that updated row is inserted into *ripoffBars*.
d) If *price* is updated for a row in *sells*, and the old price is more than 6 dollars higher than the new price, then the *bar* in that updated row is inserted into *ripoffBars*.
e) If *price* is updated for a row in *sells*, and the new price is more than 6 dollars higher than the old price, then the update returns an error.