Midterm Exam #2 (Version C)  
CS 122A  
Spring 2018  
Max. Points: 100  
(Please read the instructions carefully)

Instructions:  
- The total time for the exam is 50 minutes; be sure to budget your time accordingly.  
- The exam is closed book and closed notes but “open cheat sheet”.  
- Read each question first, in its entirety, and then carefully answer each part of the question.  
- If you don’t understand something, ask one of the exam proctors for clarification.  
- If you still find ambiguities in a question, note the interpretation you are taking.

<table>
<thead>
<tr>
<th>QUESTION</th>
<th>TOPIC</th>
<th>POINTS</th>
<th>SCORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Short Answers</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Name That Query</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>True, False, or Null?</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>How Queryous</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Never Ending SQL</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>All</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
Question 1: Short Answers (10 points)

(a) (1 pt each) Given two relations P and Q, where P has \( N_p \) tuples, Q has \( N_q \) tuples, and \( N_p > N_q > 0 \), what are the minimum and maximum possible result cardinalities for the following relational algebra queries expressed in terms of \( N_p \) and \( N_q \)?

\[
\begin{align*}
\text{Q - P:} & \quad \text{min size:} & \quad \text{max size:} \\
\pi_A P: & \quad \text{min size:} & \quad \text{max size:} \\
Q \bowtie P: & \quad \text{min size:} & \quad \text{max size:}
\end{align*}
\]

(b) (4 points) Suppose that the table Orders(uid, oname, price, shipping, tax) contains three tuples:

\[
\begin{align*}
(454, \ 'Whatzit', 50.00, 5.00, \text{NULL}) \\
(300, \ 'Gadget', 30.00, 4.00, 1.50) \\
(72, \ 'Widget', \text{NULL}, \text{NULL}, 1.20)
\end{align*}
\]

For each of the following SQL queries, show the output that it would produce if run against Orders:

(i) (2 pts) SELECT SUM(price + shipping) AS taxfreecost FROM Orders;

(ii) (2 pts) SELECT uid, (price * tax) + shipping AS totalcost FROM Orders;

SCORE: _________
Question 2: Name That Query (30 points)

(30 pts) Consider the Hoofers Diving Club database:

Divers(did, dname, rating, age)  Reserves(did, vid, rdate)  Vehicles(vid, vname, color)

What follows is a list of queries in various languages. For the first five queries, Q1-Q5, list all of the equivalent queries from the remainder of the list (Q6-Q16). Note that there may be several, or perhaps none, and that two queries are equivalent if their results are identical under all database states.

Q1 (6 pts) – \[ \pi_{\text{vid,vname,\textcolor{red}{color}}}((\sigma_{\textcolor{red}{age}=22} \text{ Divers}) \bowtie (\text{Reserves} \bowtie \text{Vehicles})) \]

Equivalent queries:

Q2 (6 pts) – \{ t(\text{age}) | \exists \ d \in \text{Divers} \ (t.\text{age} = d.\text{age} \land (d.\text{rating} < 3)) \} \}

Equivalent queries:

Q3 (6 pts) – SELECT vid FROM Vehicles WHERE color = 'red' AND color = 'purple';

Equivalent queries:

Q4 (6 pts) – Select the ids and names of divers to whom alcohol can’t be legally served at a UCI party; (Proof of age 21 or over required)

Equivalent queries:

Q5 (6 pts) – SELECT MAX(age) FROM Divers;

Equivalent queries:

Q6 – \{ t(\text{age}) | \exists \ d \in \text{Divers} \ (t.\text{age} = d.\text{age} \land \exists d2 \in \text{Divers} \ (d2.\text{age} > d.\text{age})} \}

Q7 – SELECT * FROM Vehicles V WHERE V.vid IN
  (SELECT R.vid FROM Reserves R WHERE R.did IN
  (SELECT D.did FROM Divers D WHERE D.age = 22));

Q8 – SELECT R.vid FROM Reserves R WHERE FALSE;

Q9 – \pi_{\text{age}}(\sigma_{\text{rating} < 3} \text{ Divers})

Q10 – SELECT D.age FROM Drivers D ORDER BY D.age \textit{ASC LIMIT 1};

Q11 – SELECT did, dname FROM Divers WHERE age < 21 OR age IS NULL;

Q12 – SELECT age FROM Divers WHERE rating < 3;

Q13 – SELECT did, dname FROM Divers WHERE age < 21;

Q14 – SELECT DISTINCT V.vid, V.vname, V.color FROM Reserves R, Divers D, Vehicles v
  WHERE D.age = 22 AND D.did = R.did AND R.vid = V.vid;

Q15 – \{ t(\text{age}) | \exists d \in \text{Divers} \ (t.\text{age} = d.\text{age} \land (d.\text{rating} >= 3)) \} \}

Q16 – SELECT R.vid FROM Reserves R, Vehicles V1, Vehicles V2
  WHERE (V1.color = 'red' OR R.vid = V1.vid ) AND (V2.color = 'purple' OR R.vid = V2.vid);

SCORE: _________
Question 3: True, False, or Null? (20 points)

(2 pts each) For each of the following statements, indicate whether the statement is true (circle TRUE) or false (circle FALSE):

- The use of triggers and stored procedures is advisable when writing database applications that need to be easily run on different relational DBMS choices (e.g., MySQL, Oracle, DB2, …).
  
  TRUE  FALSE

- If “CREATE TABLE Shirts (name VARCHAR(40), size ENUM('small', 'medium', 'large'))” is the DDL for the table Shirts, “SELECT size FROM Shirts” can have at most 3 result rows.
  
  TRUE  FALSE

- Most real SQL implementations disallow CHECK constraints that involve correlated subqueries.
  
  TRUE  FALSE

- Enforcing the uniqueness of a table’s column (e.g., User.email) would be a good use of a trigger.
  
  TRUE  FALSE

- Consider the query “SELECT * FROM Orders o INNER JOIN Lineitems l ON (o.oid = l.oid)”. The result tuple count for this query could be larger if “INNER” were “LEFT OUTER” instead.
  
  TRUE  FALSE

- Two tables (or queries) are considered by SQL to be UNION-compatible if they have the same number of columns and the same column names in the same order.
  
  TRUE  FALSE

- The query “(SELECT * FROM Boat B WHERE B.color = ‘yellow’) UNION ALL (SELECT * FROM Boat B WHERE B.color != ‘yellow’)” is a way to (inefficiently ⊖) return all Boat tuples.
  
  TRUE  FALSE

- The SQL set operation EXCEPT is not actually necessary because it is always possible to express an equivalent query by appropriately using a subquery instead.
  
  TRUE  FALSE

- It’s possible to write a tuple relational calculus query involving Sailor, Boat, and Reservation relations containing less than 20 tuples each whose result tuple count is infinite.
  
  TRUE  FALSE

- It’s possible to write a relational algebra query involving small Sailor, Boat, and Reservation relations whose result tuple count is infinite.
  
  TRUE  FALSE

SCORE: _________
Question 4: How Queryous (20 points)

(20 pts) Consider again the Hoofers Diving Club database, and suppose (if needed or helpful) that the tables contain the data shown below.

Divers(did, dname, rating, age)
Reserves(did, vid, date)
Vehicle(vid, vname, color)

<table>
<thead>
<tr>
<th>Divers</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>did</td>
<td>dname</td>
<td>rating</td>
<td>age</td>
</tr>
<tr>
<td>2</td>
<td>Sally</td>
<td>9</td>
<td>22</td>
</tr>
<tr>
<td>3</td>
<td>Zack</td>
<td>10</td>
<td>19</td>
</tr>
<tr>
<td>5</td>
<td>Joe</td>
<td>null</td>
<td>35</td>
</tr>
<tr>
<td>1</td>
<td>Bob</td>
<td>5</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td>Abby</td>
<td>3</td>
<td>null</td>
</tr>
</tbody>
</table>

| Reserves | | | |
|---|---|---|
| did | vid | date |
| 1 | 4 | 2017-03-15 |
| 4 | 4 | 2018-01-01 |
| 5 | 1 | 2017-12-25 |
| 1 | 5 | 2017-04-15 |
| 3 | 2 | 2017-04-15 |

<table>
<thead>
<tr>
<th>Vehicles</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>vid</td>
<td>vname</td>
<td>color</td>
</tr>
<tr>
<td>1</td>
<td>Scooter</td>
<td>red</td>
</tr>
<tr>
<td>2</td>
<td>Submarine</td>
<td>yellow</td>
</tr>
<tr>
<td>6</td>
<td>Semi-wet sub</td>
<td>red</td>
</tr>
<tr>
<td>3</td>
<td>Mini-sub</td>
<td>green</td>
</tr>
<tr>
<td>4</td>
<td>Hyper-sub</td>
<td>green</td>
</tr>
<tr>
<td>5</td>
<td>Wet-sub</td>
<td>yellow</td>
</tr>
</tbody>
</table>

Write the following queries in the indicated language:

(10 pts) **Relational Algebra:**
Print the vids of vehicles that are reserved by a diver with a rating of at least 5 as well as a diver named Abby.

(10 pts) **SQL:**
For each diver, print their did, name, rating, and – if they have any red vehicle reserved – the reservation details (date, vid, and vname).

SCORE: _________
Question 5: Never Ending SQL (20 points)

And again! Just when you thought it was safe. For this problem, you are to assume that the Reserves table has been created with did and vid each being defined as FOREIGN KEYs with the ON DELETE CASCADE option for their respective tables (Divers and Vehicles). Note that a new table has just been added to the mix – VehiclePrefs – which has no primary or foreign keys. VehiclePrefs is initially empty and will soon be used by the club to assist them in advertising vehicles and other services to their club members.

<table>
<thead>
<tr>
<th>Divers(did, dname, rating, age)</th>
<th>Reserves(did, vid, date)</th>
<th>Vehicles(vid, vname, color)</th>
</tr>
</thead>
<tbody>
<tr>
<td>did</td>
<td>dname</td>
<td>rating</td>
</tr>
<tr>
<td>2</td>
<td>Sally</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>Zack</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>Joe</td>
<td>null</td>
</tr>
<tr>
<td>1</td>
<td>Bob</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Abby</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In addition, assume that the following additional SQL goodies have already been defined:

```
DELIMITER $$
CREATE PROCEDURE DropVehicle(oldvehicle INT(11))
BEGIN
  DELETE FROM Vehicles WHERE vid = oldvehicle;
END $$
CREATE TRIGGER TrackActivity
AFTER INSERT ON Reserves FOR EACH ROW
BEGIN
  INSERT INTO VehiclePrefs(did, date, vname, color)
  VALUES( NEW.did, NEW.date,
          (SELECT vname
           FROM Vehicles WHERE vid = NEW.vid),
          (SELECT color
           FROM Vehicles WHERE vid = NEW.vid));
END; $$
DELIMITER ;
```

(a) (10 pts) Show below the result of running the following statement against the initial four tables above. (I.e., show the contents of the tables after execution. You may skip any tables that are unaffected.)

```
CALL DropVehicle(1);
```
Question 5: SQL Never Ending SQL (continued)

Here’s all of the database information once again. You are to assume that Reserves table was created with did and vid each being defined as FOREIGN KEYs with the ON DELETE CASCADE option for their respective tables (Divers and Vehicles). Note the new table – VehiclePrefs – which has no primary or foreign keys. This new table, which they just added, is currently empty and will be used by the club to assist them in advertising vehicles and other services to their club members.

Divers(did, dname, rating, age)  Reserves(did, vid, date)  Vehicles(vid, vname, color)
VehiclePrefs(did, date, vname, color)

(b) (5 pts) Show below the result of running the following statement starting from the initial four tables above. (I.e., show the contents of the tables after execution. You may skip any tables that are unaffected.)

```
INSERT INTO Reserves(did, vid, date) VALUES (3,2,'2018-05-30'), (5,4,'2018-05-31');
```

(c) (5 pts) Show below the results of executing the following statement against the initial tables above.

```
CREATE VIEW AgeSummary(age, numdivers) AS
    SELECT age, COUNT(*) as numdivers FROM Divers GROUP BY age;
SELECT * FROM AgeSummary A WHERE A.age >= 21;
```