Question Booklet

W4111 Introduction to Databases
Spring 2019
Midterm Exam Solutions
Instructor: Eugene Wu

Closed Book, 1 page notes: 8.5x11" letter paper, both sides
Duration: 75 minutes
Havemeyer 309

Instructions

This is the question booklet, which contains questions for the exam.
There is a separate answer booklet for your answers.

1. You are supposed to write your answers on the answer sheets.
2. The staff will ignore text written on the question sheets.
3. You will submit the Quesion AND Answer booklets at the end of the exam. If we do not receive both booklets with your UNI, **you will receive a zero**.

Your Name: Alice H. Acker
Your UNI: aa0000

ALL regrade requests due by 3/15 10AM

If we added your score incorrectly, submit regrade request on Gradescope.
If there is an error in the solutions, please let us know with a **private piazza message**.
If you want a regrade of a question, submit a regrade request on Gradescope—we will *regrade it carefully and deduct points for syntactic errors that we ignored.*

1 (16 points) Equivalences

Consider the following schema. You may assume that no primary keys have been explicitly defined by the database administrator, and that there are no null values. Otherwise, do not make any assumptions about the database.

```
A(a int, x int, c int);
B(b int, x int, d int)
```

1.1 (4 Points)

Consider the following relational algebra statement:

$$\pi_{a,b}(\sigma_{a>10}(A \bowtie_x B))$$

For each statement below, fill in (T) if it is equivalent to the above statement, and (F) otherwise.

(A) $\pi_{a,b}(\sigma_{a>10}(A \bowtie_{A.x=B.x} B))$

(B) $\sigma_{a>10}(\pi_{a,b}(A) \bowtie_x B))$

(C) $\pi_{a,b}(A \bowtie_a \sigma_{a>10}(B))$

(D) $\pi_{a,b}(\sigma_{a>10}(A) \times B)$

T,F,F,F

1.2 (4 Points)

Consider the following statement:

```
SELECT *
  FROM A
 WHERE NOT EXISTS (SELECT COUNT(*)
                   FROM B
                   WHERE B.x = A.x) AND
       A.a = A.x
```

For each statement below, fill in (T) if it is equivalent to the above statement, and (F) otherwise.

(A) $\sigma_{a=x}(A)$

(B)
```
SELECT a, b, x
  FROM A, B
GROUP BY B.x
  HAVING B.x = A.x AND A.a = A.x
```

(C) $\sigma_{A.a=A.x}(A \bowtie_{B.x=A.x} B)$

(D)
```
SELECT *
  FROM A
  WHERE A.x IN (SELECT B.b FROM B) AND
        A.a = A.x
```

F,F,F,F

1.3 (4 Points)

Consider the following statement:

```
SELECT * FROM A
INTERSECT
SELECT * FROM B
```

For each statement below, fill in (T) if it is equivalent to the above statement, and (F) otherwise.

(A)
```
SELECT * FROM A
WHERE (a,x,c) IN (SELECT * FROM B)
```

(B) $A \cap B$

(C) $A - (B - A)$

(D)
```
SELECT *
  FROM A
  WHERE NOT EXISTS (SELECT 1 FROM B
                    WHERE NOT EXISTS (SELECT 1 WHERE A.x = B.x))
```

F,T,F,F

1.4 (4 Points)

Consider the following schema and SQL query:

```
CREATE TABLE S(
  s int PRIMARY KEY
)
CREATE TABLE T(
  s int REFERENCES S NOT NULL,
  t int PRIMARY KEY
)

  SELECT S.s
    FROM S, T
   WHERE S.s = T.s AND t > 10
GROUP BY S.s
  HAVING COUNT(*) > 2
```

Write a relational algebra statement that is equivalent to the SQL query, or "NOT POSSIBLE" if there does not exist a relational algebra equivalent.

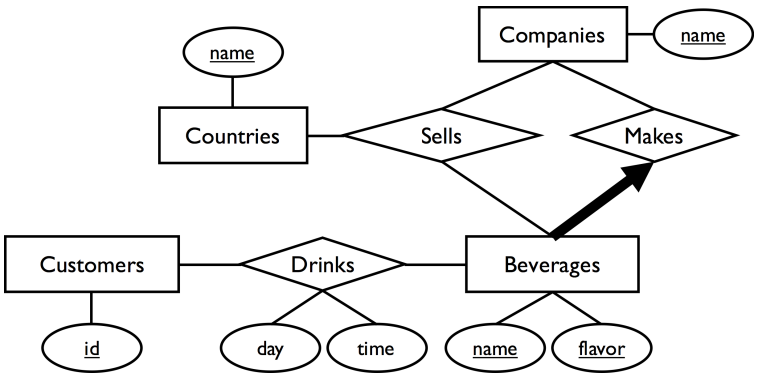
$\rho(Z1, \sigma_{t>10}(S \bowtie_{S.s=T.s} T))$
$\rho(Z2, \sigma_{t>10}(S \bowtie_{S.s=T.s} T))$
$\rho(Z3, \sigma_{t>10}(S \bowtie_{S.s=T.s} T))$
$\pi_{Z1.s}((Z1 \bowtie_{Z1.s=Z2.s \wedge Z1.t!=Z2.t} Z2) \bowtie_{Z2.s=Z3.s \wedge Z1.t!=Z3.t \wedge Z2.t!=Z3.t} Z3)$

The key parts are

1. the use of temporary tables,
2. using the join between temporary tables to express the `COUNT`,
3. to filter using `Z1.t != Z2.t`,
4. and the final projection.

2 (18 points) Entity-Relationship Models

The non-alcoholic beverage market was estimated at $967 Billion in 2016. With so much at stake, it is difficult for these companies to keep track of who makes which beverage, and who owns the rights to sell beverages in each country. It is also hard to know which beverages customers like. To this end, companies have decided to make cans and bottles IoT devices, so that they can track every time a customer enjoys a delicious beverage. To help manage all of this information, they have hired database experts to design the following ER diagram:



2.1 ER Diagrams (2 Points Each)

For each of the following statements, specify True if the statement is consistent with the constraints in the ER diagram, and False otherwise (or if it disallowed).

(A) The companies *Pepsico* and *Coca Cola* can both make the beverage *Limade* (this is both the name and the flavor).

(B) The *Pepsico* company can make *Lipton Tea* but not have sales rights in any country.

(C) *Professor Wu* is a customer. He can drink *Grapefruit-flavored Bubly* every Tuesday and Thursday.

(D) The beverages *Le Croix* and *Bubly* can be sold by the same company in the *United States*.

(E) The *Suntory* company makes the *Peach Chuhai* beverage. Unfortunately, when *Suntory* went bankrupt and its compary record was deleted, *Peach Chuhai* had to be deleted as well. Now customers are unable to drink it anymore.

1. F: Beverages has exactly one relationship with companies,
2. T: The makes relationship is distinct from the sells relationship,

3. F: The key for the drinks relationship is the customer and beverage key,

4. T: sells is a many to many relationship,

5. F: Beverage is not a weak entity, thus the beverage need not be deleted. As an aside, Suntory cannot be deleted without more information. Peach Chuhai (and all other beverages made by Suntory) should first be deleted or given a new manufacturer, before Suntory is deleted. This is referential integrity.

2.2 Schemas (2 Points Each)

Consider the following SQL schema, which translates of the ER diagram above into tables in the relational model.

```
CREATE TABLE Customers(id PRIMARY KEY);
CREATE TABLE Countries(name text PRIMARY KEY);
CREATE TABLE Companies(name text PRIMARY KEY);
CREATE TABLE Beverages(
  name text,
  flavor text,
  PRIMARY KEY(name, flavor)
);

CREATE TABLE drinks(
  cid int references Customers(cid),
  name text references Beverages(name),
  flavor text references Beverages,
  day date,
  time time,
  PRIMARY KEY(day, time, cid, name, flavor)
);

CREATE TABLE makes(
  name_bev text REFERENCES Beverages(name),
  flavor text REFERENCES Beverages(flavor),
  name_comp text REFERENCES Companies(name),
  PRIMARY KEY(name_comp, flavor, name_bev),
  UNIQUE (name_bev, flavor)
)

CREATE TABLE sells(
  name_country text REFERENCES Countries(name),
  name_company text REFERENCES Company(name),
  name_bev text REFERENCES Beverages(name),
  flavor text REFERENCES Beverages(flavor),
  PRIMARY KEY (name_country, name_company, name, flavor)
)
```

List and briefly explain the **four most important problems** that you find with this schema, in terms of how well it models the E/R diagram above. For each problem, provide a very brief example of data that illustrates the problem. Do not base your answer on comparing this schema with other possible schemas. Instead, just compare the schema against the E/R diagram to identify what is not captured from the diagram, etc. in the schema.

You will be graded on the importance of the problems that you identify and the extent that the problems you describe overlap. For example, if you say "table Foo is wrong" as one problem, and "relationship with

Foo is a problem because Foo is wrong", it is clear the second statement is implied by the first. Thus, we will grade the second "problem" lower.

1. Customers.id missing type constraint
2. drinks primary key should only be cid, name, flavor
3. drinks.cid should reference Customers(id)
4. makes translation expresses at-most-one rather than exactly one.
5. name_bev should be in primary key of sells/sells.name in its primary key is not defined
6. (sales.name_bev, sales.flavor) should be a composite reference of Beverages(name, flavor). Same for other relations as well.

3 (8 points) Triggers

Consider the following database schema, and each table is empty:

```
CREATE TABLE A(a int);
CREATE TABLE B(b int);
```

We have installed the following trigger in the database.

```
CREATE FUNCTION UDF1() RETURNS trigger AS $$
BEGIN
 IF NEW.a < 2 THEN
   INSERT INTO A VALUES (NEW.a + 1);
 END IF;
 INSERT INTO B VALUES (NEW.a);
 RETURN (NEW.a + 4);
END;
$$ language plpgsql;
```

```
CREATE TRIGGER T1
BEFORE INSERT ON A
FOR EACH ROW
  EXECUTE PROCEDURE UDF1()
```

The user then runs the following query:

```
INSERT INTO A VALUES (1), (2);
```

3.1 Table A (4 Points)

Specify the contents in tables `A`. Since each row is a single attribute, simply write a list of value(s). For instance, 1, 1, 2, 3, 5.

A: 6,5,6

1 point for each correct value, 1 extra point for fully correct table. Incorrect values lost points (e.g., if you wrote 1,2,3,4,5,6,7).

3.2 Table B (4 Points)

Specify the contents in tables `B`. Since each row is a single attribute, simply write a list of value(s). For instance, 3, 14, 15, 9.

B: 2,1,2

1 point for each correct value, 1 extra point for fully correct table. Incorrect values lost points (e.g., if you wrote 1,2,3,4,5,6,7).

4 (8 points) Misc. Questions

4.1 (2 Points)

In at most 2 sentences, describe the main difference between constraints and triggers.

Contraints restrict what database instances are allowed but do not modify the database contents, trigger may modify the contents.

4.2 (2 Points)

Which relational algebra operators can generate more result records than are in their inputs.

$\bowtie$, $\times$

4.3 (2 Points)

In at most 2 short sentences, describe what union compatible means.

Two schemas are union compatible if they contain the same sequence of attribute types.

4.4 (2 Points)

Assume input relation $T(a)$. Susan runs a query $T \times T \times T \times T$ and wants only keep results where the second T relation's a value is equal to 1. What concept can Susan use to unambiguously specify the filtering condition? Write "NOT POSSIBLE" if it is not possible in relational algebra.

Positional notation. $\sigma_{\$2=1}$. Renaming the tables so that each T is unambiguous was also acceptable if clearly stated.

5 (13 points) Intro to Databases

An anonymous database course (certainly not W4111) is having trouble managing assignment grades for the class. Consultants were hired to design the database schema but forgot to write any SQL queries to analyze the data. It's up to you to help save the day.

Consider the following database schema. `Students` contains information about each student, including the program the student is part of. `ug` means undergraduate, `ms` means masters of CS (Counter Strike), `dsi`

means masters of DSI (DotA Science Institute). `Hws` contains information about each homework assignment, including the maximum score possible, its weight when computing the total class score, and whether or not the assignment is extra credit. `Submissions` contains the student and the homework, the submission grade, and optionally the id of the teammate if it was a team assignment.

We can assume that every student has submitted each homework exactly once. If a student was the teammate (sid2) of a submission, that counts as submitting the homework.

```
Students(
  sid int primary key,
  name text not null,
  program text,
  check (program IN ('ug', 'ms', 'dsi'))
)

Hws (
  hid int primary key,
  name text not null,
  maxscore float not null,
  weight float not null,
  extracred bool not null
)

Submissions(
  sid1 int references Students(sid),
  hid int references Hws,
  sid2 int references Students(sid),
  score float,
  primary key (sid1, hid)
)
```

5.1 Submissions in Relational Algebra (3 Points)

Write a **relational algebra** statement that computes the score for every student's submission. If two students were teammates for a submission, there should be two rows in `Submits`: one for each teammate. Return the student's `sid`, the homework id `hid`, and the submission grade. We will call this table `Submits(sid, hid, score)`. You may assume that nulls are not present in the relations, and that relations are sets.

$$\rho_{\$1 \to sid,hid,score}(\pi_{sid1,hid,score}(Submissions) \cup \pi_{sid2,hid,score}(Submissions))$$

5.2 Extra Extra, Read All About It! (4 Points)

Write a **SQL query** to find the students that submitted all of the extra credit homeworks (`extracred` is true). The result should have the schema `(sid)`. You may assume `Submits` has been correctly defined, if that helps.

```
SELECT sid
FROM Students
WHERE NOT EXISTS (
  SELECT *
  FROM Hws
  WHERE extracred = true AND
```

```
            NOT EXISTS(
              SELECT *
              FROM Submits
              WHERE Submits.sid = Students.sid AND
                    Submits.hid = Hws.hid)
    )
```

5.3 Final Grades (6 Points)

Let `PreCurve(sid, is_ec, total)` be a predefined view:

```
CREATE TABLE PreCurve(sid, is_ec, total) AS
   SELECT sid, H.extracred, SUM(score / maxscore * weight)
     FROM Submits S, Hws H
    WHERE S.hid = H.hid
 GROUP BY sid, H.extracred;
```

The cutoff for receiving an `A` is defined as the average total homework score (from the `PreCurve` table) plus two times the standard deviation of total homework score. A student's final score is the sum of their homework and extra credit scores from the `PerCurve` table. If the final score is greater than or equal to the cutoff, then the student should receive an `A`.

Write a query that computes the `sid` of students that should receive an `A` grade. You may assume that `avg()`, `sum()`, `count()`, `std()` are aggregation functions that compute average, sum, count, and standard deviation for a group.

(Note that this is a fictional class. Not W4111)

```
WITH final(sid, final) AS (
  SELECT sid, sum(total)
  FROM PreCurve
  GROUP BY sid
)
SELECT sid
FROm final
WHERE final.final >= (
  SELECT avg(total) + 2*std(total)
  FROm PreCurve
  WHERE PreCurve.is_ec = false);
```

6 JOIN In! (2 Points)

6.1 (2 Points)

Write a creative example of joins In Real Life by filling in the sentence in the answer sheet. Most creative answer (subjectively judged by the staff) gets 2 extra credit points.

Accepted any answer that is sensible and can actually be viewed as a join. A common mistake was writing a non-join-related filter as the join condition. Another was to fill the blanks with words that seem similar but did not correspond to a join. Another was to join individual entities rather than actual SETS of entities.

6.2 (Up to 2 Points Extra Credit)

Draw a picture of your "Joins in Real Life" answer above. Points awarded subjectively based on whether staff can decipher a connection between the drawing and the above answer, and creativity.