CSE 344 Section 3 Worksheet Solutions

SQL Aggregates Practice

```
CREATE TABLE Movies (id int PRIMARY KEY,
                     name varchar(30),
                     budget int,
                     gross int,
                     rating int,
                     year int);

CREATE TABLE Actors (id int PRIMARY KEY,
                     name varchar(30),
                     age int);

CREATE TABLE ActsIn (mid int REFERENCES Movies(id),
                     aid int REFERENCES Actors(id));
```

What is the minimum age of an actor who has appeared in a movie where the gross of the movie has been over $1,000,000,000?

```
SELECT MIN(A.age)
  FROM Movies as M, ActsIn as AI, Actors as A
 WHERE M.id = AI.mid AND A.id = AI.aid
       AND M.gross > 1000000000;

Notes:
M.id = AI.mid links movies to ActsIn
A.id = AI.aid links actors to ActsIn
"actor A acted in Movie M"
```

What is the name and budget of each movie released in 2017 whose oldest actor is less than 30?

```
SELECT M.name, M.budget
  FROM Movies as M, ActsIn as AI, Actors as A
 WHERE M.id = AI.mid AND A.id = AI.aid AND M.year = 2017
 GROUP BY M.id, M.name, M.budget
HAVING MAX(A.age) < 30;
```

More Challenging

```
  CREATE TABLE Class (
   dept VARCHAR(50),
   number INT,
   title VARCHAR(50),
   PRIMARY KEY (dept, number));

  CREATE TABLE Instructor (
   username VARCHAR(50) PRIMARY KEY,
   fname VARCHAR(50),
   lname VARCHAR(50),
   started_on CHAR(10));

  CREATE TABLE Teaches (
   username VARCHAR(50),  -- alternately, can use the REFERENCES syntax here
   dept VARCHAR(50),
number INT,
PRIMARY KEY (username, dept, number),
FOREIGN KEY username REFERENCES Instructor,
FOREIGN KEY (dept, number) REFERENCES Class);
```

1. For each instructor, return their username and the number of classes they teach.

```
SELECT I.username, COUNT(T.number) AS count
  FROM Instructor AS I LEFT OUTER JOIN Teaches AS T
           ON I.username = T.username
 GROUP BY I.username;
```

```
Notes:
COUNT(column) ignores NULL values
   -  Instructors with no classes
   -  Inside T.number, have NULL
   -  Their count would show as 0 which is great!
COUNT(*) does count the NULL row. We don't want that
   -  Here, in same scenario, where inside T.number we have null
   -  Their count would show 1. This is bad since a NULL here
      means instructor doesn't teach a class yet output shows 1.
      Contradiction!
```

2. Return the first and last name of the newest instructor(s) (who started on the latest date). Assume Instructor.started_on uses yyyy-mm-dd format. If there are multiple instructors, list all of them.
   - Example of the witnessing problem that doesn't require subqueries.

```
SELECT I.fname, I.lname
  FROM Instructor AS I, Instructor AS I2
 GROUP BY I.username, I.fname, I.lname, I.started_on
HAVING I.started_on = MAX(I2.started_on);
```

Or using a subquery:

```
SELECT fname, lname
FROM Instructor
WHERE started_on = (
  SELECT MAX(started_on)
  FROM Instructor
);
```

3. How many classes are currently being taught by at least one instructor?

By the nature of our data, we know that any class that appears in Teaches must be taught by at least 1 teacher. Thus, if we categorize the tuples in Teaches by dept and coursenum (the primary key), we can get our answer by counting the number of groups. The sticking point of this query is how to count the number of groups. The easy solution is to wrap the grouping query in a count(*) query.

```
SELECT COUNT(*)
  FROM (SELECT DISTINCT dept, coursenum
          FROM Teaches);
```

4. Return the first name and last name of the instructors who teach the most number of classes. If there are multiple instructors, list all of them.

   This is another example of the witnessing problem. There are multiple approaches - see below.

   SOLUTION 1:

```
WITH (
   SELECT username, COUNT(*) AS count
     FROM Teaches
    GROUP BY username
) AS ClassCounts
SELECT I.fname, I.lname
  FROM ClassCounts AS C1, ClassCounts AS C2, Instructor AS I
 WHERE C1.username = I.username
 GROUP BY I.username, I.fname, I.lname, C1.count
HAVING C1.count = MAX(C2.count);
```

   SOLUTION 2:

```
WITH (
   SELECT username, COUNT(*) AS count
     FROM Teaches
    GROUP BY username
) AS ClassCounts,
(
   SELECT MAX(count) AS max
     FROM ClassCounts
) AS MaxCounts
SELECT I.fname, I.lname
  FROM ClassCounts AS C, MaxCounts AS M, Instructor AS I
 WHERE C.username = I.username AND C.count = M.max;
```

   SOLUTION 3:

```
SELECT I.fname, I.lname
  FROM Instructor AS I, Teaches AS T
 WHERE I.username = T.username
 GROUP BY I.username, I.fname, I.lname
HAVING COUNT(*) >= ALL (
                    SELECT COUNT(*)
                      FROM Teaches
                     GROUP BY username
               );
```