CSE 344 Section 3 Worksheet Solutions

1)"Complicated" SQL Practice

```
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 (as
 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);
```

a. Return the first and last name of all instructors who are teaching at least one class.
   - This problem does not require any subqueries. However, we must group by the Instructor's first name and last name along with the username in order to select those attributes.

```
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(*) >= 1;
```

b. For each instructor, return their username and the number of classes they teach.
   - This problem does not require any subqueries:

```
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;
```

c. Unnest the following SQL queries so they do not use subqueries:

```
SELECT I.username, (
        SELECT COUNT(*)
          FROM Teaches AS T
         WHERE T.username = I.username
       )
  FROM Instructor AS I;
```

```
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;

SELECT C1.dept, C1.number
FROM Class AS C1
WHERE C1.number >= ALL (
         SELECT C2.number
           FROM Class AS C2
          WHERE C1.dept = C2.dept
      );

SELECT C.dept, MAX(C.number)
  FROM Class AS C
 GROUP BY C.dept;
```

d. Return the number of instructors who are teaching at least one class. (1 row)

- By the nature of our data, we know that any instructor that appears in Teaches must teach at least 1 class. Thus, if we categorize the tuples in Teaches by username (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 1
          FROM Teaches
         GROUP BY username);


SELECT COUNT(*)
  FROM (SELECT username
          FROM Teaches
         GROUP BY username);
```

- You can actually do this without a subquery with the following subquery:

```
SELECT COUNT(DISTINCT username)
  FROM Teaches;
```

e. 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. Here is one solution.

```
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);
```

- Another solution using subqueries is

```
SELECT I1.fname, I1.lname
FROM Instructors AS I1
WHERE I.started_on >= ALL(
      SELECT I2.started_on
```

```
      FROM Instructors AS I2
)
```

f. 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. We can utilize the query from above to help solve this problem as a subquery. From there, there are multiple ways to return the intended result.

```
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);
```

```
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;
```

```
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
                  );
```

g. Which CSE courses do neither Dr. Levy (username 'levy') nor Dr. Wetherall (username 'djw') teach? Give the department, number, and title of these courses.

- The framing of this question is a negated existential. This hints that a simple SELECT-FROM-WHERE query (monotonic query) will not work.
- A gut reaction, if you think of filtering out tuples with levy or djw, might lead to the query below.

This query is **wrong**! Imagine we have a course taught by levy. You can see that if we have a course taught by suciu and levy, that tuple will end up in the answer even though it shouldn't

```
SELECT C.dept, C.number, C.title
  FROM Class C, Teaches T
 WHERE C.dept = 'CSE' AND C.dept = T.dept AND C.number = T.number
        AND T.username != 'levy' AND T.username != 'djw';
```

- The tricky part of this problem is that more than one instructor may teach a single course. But this problem can be solved with subqueries easily. A negated existential problem can be translated directly into SQL via the NOT IN keywords.

```
SELECT C.dept, C.number, C.title
  FROM Class C
 WHERE C.dept = 'CSE'
       AND C.number NOT IN
                 (SELECT C1.number
                    FROM Class C1, Teaches T
                   WHERE C1.dept = T.dept AND C1.number = T.number
                       AND C1.dept = 'CSE'
                     AND (T.username = 'levy' OR T.username = 'djw'));
```

- Alternatively, you might take a different approach: to compute classes in CSE, then subtract those taught by levy or djw. This decorrelated version uses *set difference*. This is decorrelated since it uses two separate queries.

```
SELECT C.dept, C.number, C.title
FROM Class C
WHERE C.dept = 'CSE'
EXCEPT
SELECT C.dept, C.number, C.title
FROM Class C, Teaches T
WHERE C.dept = 'CSE' AND C.dept = T.dept AND C.number = T.number AND
(T.username = 'djw' OR T.username = 'levy');
```