CS 157A Spring 2020 **Homework 4**

Point: 10 (each question is 1 point)

Part 1 (Listen to the posted Video lecture on Nested/Hash Join and answer following questions):

1. What is the time complexity of nested join algorithm?
   (a) O(m+n)
   (b) O(m*n)
   (c) O(m)
   *m and n are number of rows of tables being joined.
2. If we use hash join algorithm to join takes and section tables, then which table should be used to build hash table?
   (a) takes and section (both)
   (b) either of them (takes or section)
   (c) pick table with fewer number of rows to build hash table

Part 2: SQL questions

The following queries are based on the campaign data (campaign-ca-2016.sql)

3. how many contributions are contained in the data?

   select count(*) from campaign;

4. show the min and maximum amounts of all contributions?

   select min(contb_receipt_amt), max(contb_receipt_amt) from campaign;

5. list the (distinct) ids and names of all the candidates in the data. order by name?

   select distinct cand_id, cand_nm from campaign order by cand_nm;

6. show the candidate name and number of contributions, for each candidate? order by number of contributions in descending order?

   select cand_nm, count(*) as cnt from campaign group by cand_nm order by cnt desc;

7. show the candidate name and average contribution amount for each candidate, looking at positive contributions only? Order by average amount in descending order?

   ```
   select cand_nm, round(avg(contb_receipt_amt)) as avg_amt

   from campaign

   where contb_receipt_amt > 0

   group by cand_nm

   order by avg_amt desc;
   ```

8. show the candidate name and the total amount received by each candidate. Order the output by total amount received.

   ```
   select cand_nm, sum(contb_receipt_amt) as tot from campaign group by cand_nm
   order by tot desc;
   ```

9. how do you set the SQL output so that it no longer goes to a file?
   ```
   .out stdout
   ```

Part 3: Use the courses data (read the files `courses-ddl.sql` and `courses-small.sql`).

10. write an SQL query that gives the number of courses taken for every student in the student table.

    ```
    select ID, count(course_id) as course_count

      from student natural left outer join takes

      group by ID;
    ```