CS345 - Final - Fall 2017 - 11/26/2017

1. Do not open the exam until you are directed to do so. Read all of the instructions first.
2. Please print your name on every page of the exam (in the upper right corner). Additionally, sign your name on the front page.
3. The exam contains 8 pages (including this one). Verify that you do indeed have 8 pages.
4. Please turn cell phones **off**.
5. The exam is closed book and closed notes. Please place books, bags, notes and personal items **under** your desk.
6. The exam is closed neighbor. Refrain from talking to your neighbors during the exam.
7. Computers, headsets, hats or sunglasses are **not** permitted.
8. You may bring a calculator.
9. Write your solutions in the space provided. If you need extra space or scratch space, you may use pp. 8. If you need additional pages, raise your hand and additional paper will be provided. Any additional paper you use must be turned in with your exam and have your name printed on the upper right corner.
10. Do not spend too much time on one problem. Read them all and attack them in the order that allows you to make the most progress.
11. Show your work. Partial credit will be given. You are graded not only on correctness of outcome but in **correctness** and **clarity** of approach.
12. Questions during the exam should only be of an administrative nature (I'm missing page 2, I can't read this, etc.)
13. Good luck!

Name (printed) Name (signed)

Comparisons and Boolean operators

1. Return the trainee ID and full name for all trainees who are older than 20.

```
SELECT
      trainee.ID, trainee.first_name, trainee.last_name
FROM
      trainee
WHERE
      trainee.age > 20
;
```

2. Return the ids and names of trainees whose last name starts with the letter 'D'.

```
SELECT
      trainee.ID, trainee.first_name, trainee.last_name
FROM
      trainee
WHERE
      SUBSTRING(trainee.last_name, 1, 1) = 'D'
;
```

3. Assume that you have a view available, that returns all data, called 'all_data'. See the attachment.

Return the ids and names of trainees, along with the course ID and short name of courses they take, whose first name starts with the letter 'M' OR whose last name starts with the letter 'R'. Of that group of trainees we only want to return those who are between 20 and 30 years old (exclusively).

```
SELECT
      trainee_id, trainee_first_name, trainee_last_name,
course_id, course_short_name
FROM
      all_data
WHERE
      (SUBSTRING(trainee_first_name,1,1) = 'M' OR
SUBSTRING(trainee_last_name,1,1) = 'R') AND trainee_age > 20 AND
trainee_age < 30
;
```

4. Assume that you have a view available, that returns all data, called 'all_data', see the attachment.
   Who are the trainees who received an A in CS345? Return the full name and ID.

```
SELECT
      trainee_first_name, trainee_last_name, trainee_id
FROM
      all_data
WHERE
      trainee_grade = 'A'
;
```

5. Assume that you have a view available, that returns all data, called 'all_data', see the attachment.

   See the last question: who failed that course? Trainees fail a class if they receive either an 'F' or a 'D'.

```
SELECT
      trainee_first_name, trainee_last_name, trainee_id
FROM
      all_data
WHERE
      trainee_grade = 'F' OR trainee_grade = 'D'
;
```

6. Assume that you have a view available, that returns all data, called 'all_data', see the attachment.

   Which courses are popular among older trainees (course ID and full name)? Return the courses that are taken by trainees who are older than 40. Return each course only once.

```
SELECT
      course_id, course_long_name
FROM
      all_data
WHERE
      trainee_age > 40
GROUP BY
      course_id
;
```

Joins and order by

For all join problems, you are not allowed to use view 'all_data'.

7. Write a query that returns the instructor IDs, their full name and the course short names that they teach. Use as small a number of tables as possible. Order the result by instructor last name, first name.

```
SELECT
      instructor.id, instructor.last_name, instructor.first_name,
course.short_name
FROM
      course
            INNER JOIN
      class ON class.course_id = course.id
            INNER JOIN
      instructor ON class.instructor_id = instructor.id
;
```

8. Return all the IDs and full names of trainees who have an A in any class. Make sure to return any name only once. Also, use only tables that are necessary.

```
SELECT
      trainee.id, trainee.first_name, trainee.last_name
FROM
      trainee
            INNER JOIN
      trainee_class ON trainee_class.student_id = student.id
WHERE
      trainee_class.grade = 'A'
;
```

9. Write a query that returns the complete data set over all tables, ordered by: trainee last name, first name, grade. Hint: to join in instructor, the intermediate table structure needs to already contain table class. That said, you might as well join instructor as the very last table.

```
SELECT
      trainee.last_name, trainee.first_name, trainee_class.grade
FROM
      course
            INNER JOIN
      class ON class.course_id = course.id
            INNER JOIN
      trainee_class ON trainee_class.course_id = class.course_id
            AND trainee_class.section_id = class.section_id
            INNER JOIN
      instructor ON class.instructor_id = instructor.id
            INNER JOIN
      trainee ON trainee.id = trainee_class.trainee_id
;
```

10. Write a query that returns all trainee IDs and full names of trainees who have not taken any class.

```
SELECT
      trainee.id, trainee.last_name, trainee.first_name
FROM
      trainee
            LEFT JOIN
      trainee_class ON trainee_class.trainee_id = trainee.id
WHERE
      trainee_class.trainee_id IS NULL
;
```

11. Write a query that returns the course ID as well as short and long name of courses that have not been offered at all.

```
SELECT
      course.id, course.short_name, course.long_name
FROM
      course
            LEFT JOIN
      class ON course.id = class.course_id
WHERE
      class.id IS NULL
;
```

Group by

12. Return each course (not class), i.e. its ID and short name, along with the age of the oldest trainee that takes the course (in some class).

```
SELECT
      course.id, course.short_name, MAX(trainee.age)
FROM
      course
            LEFT JOIN
      class ON class.course_id = course.id
            LEFT JOIN
      trainee_class ON trainee_class.course_id = course.id AND
trainee_class.section_id = class.section_id
            LEFT JOIN
      trainee ON trainee.id = trainee_class.trainee_id
GROUP BY course.id
```

13. Return the number of classes each trainee took. Return the trainee ID and full name.

```
SELECT
      COUNT(trainee_class.trainee_id), trainee.id,
trainee.first_name, trainee.last_name
FROM
      trainee
            INNER JOIN
      trainee_class ON trainee_class.trainee_id = trainee_id
GROUP BY
      trainee.id
;
```

14. Return the trainee ID, along with first/last name and the best grade that trainee achieved in any class.

```
SELECT
      trainee.id, trainee.first_name, trainee.last_name,
MIN(trainee_class.grade)
FROM
      trainee
            INNER JOIN
      trainee_class ON trainee_class.trainee_id = trainee_id
GROUP BY
      trainee.id
;
```

Subqueries

15. Return the course (not class!) ID and short/long names of all courses, along with the oldest trainee in the course, plus the trainee's ID and full name. Write a correlated subquery in order to achieve this.

```
SELECT
      outer_table.course_id, course_short_name, course_long_name,
trainee_age, trainee_id, trainee_first_name, trainee_last_name
FROM
      all_data AS outer_table
WHERE trainee_age = (
      SELECT
            max(trainee_age)
      FROM
            all_data AS inner_table
      WHERE
            outer_table.course_id = inner_table.course_id )
GROUP BY
      course_id
;
```

16. See the last question: write the same query, but use an INNER JOIN, instead of a correlated subquery.

```
SELECT
      outer_table.course_id, course_short_name, course_long_name,
trainee_age, trainee_id, trainee_first_name, trainee_last_name
FROM
      all_data AS outer_table
INNER JOIN
(
      SELECT
            course_id, max(trainee_age) AS max_age
      FROM
            all_data
      GROUP BY course_id
      ) AS inner_table
      ON outer_table.course_id = inner_table.course_id AND
outer_table.trainee_age = inner_table.max_age
;
```

```
CREATE VIEW all_data AS
  (SELECT
     course.id AS course_id,
     class.section_id AS section_id,
     course.short_name AS course_short_name,
     course.long_name AS course_long_name,
     instructor.id AS instructor_id,
     instructor.first_name AS instructor_first_name,
     instructor.last_name AS instructor_last_name,
     trainee_class.grade AS trainee_grade,
     trainee.id AS trainee_id,
     trainee.first_name AS trainee_first_name,
     trainee.last_name AS trainee_last_name,
     trainee.age AS trainee_age
   FROM
     course
       INNER JOIN
     class ON course.id = class.course_id
       INNER JOIN
     trainee_class ON trainee_class.course_id = class.course_id
       AND trainee_class.section_id = class.section_id
       INNER JOIN
     instructor ON class.instructor_id = instructor.id
       INNER JOIN
     trainee ON trainee.id = trainee_class.trainee_id);
```