Homework 10: SQL hw10.zip (hw10.zip)

Due by 11:59pm on Thursday, November 30

Instructions

Download hw10.zip (hw10.zip).

Submission: When you are done, submit the assignment by uploading all code files you've edited to Gradescope. You may submit more than once before the deadline; only the final submission will be scored. Check that you have successfully submitted your code on Gradescope. See Lab 0 (/lab/lab00#submitting-the-assignment) for more instructions on submitting assignments.

Using Ok: If you have any questions about using Ok, please refer to this guide. (/articles/using-ok)

Grading: Homework is graded based on correctness. Each incorrect problem will decrease the total score by one point. There is a homework recovery policy as stated in the syllabus. **This homework is out of 2 points.**

To check your progress, you can run `sqlite3` directly by running:

```
python3 sqlite_shell.py --init hw10.sql
```

You should also check your work using `ok`:

```
python3 ok
```

Required Questions

Getting Started Videos

SQL

Dog Data

In each question below, you will define a new table based on the following tables.

```
CREATE TABLE parents AS
  SELECT "abraham" AS parent, "barack" AS child UNION
  SELECT "abraham"          , "clinton"         UNION
  SELECT "delano"           , "herbert"         UNION
  SELECT "fillmore"         , "abraham"         UNION
  SELECT "fillmore"         , "delano"          UNION
  SELECT "fillmore"         , "grover"          UNION
  SELECT "eisenhower"       , "fillmore";

CREATE TABLE dogs AS
  SELECT "abraham" AS name, "long" AS fur, 26 AS height UNION
  SELECT "barack"         , "short"      , 52           UNION
  SELECT "clinton"        , "long"       , 47           UNION
  SELECT "delano"         , "long"       , 46           UNION
  SELECT "eisenhower"     , "short"      , 35           UNION
  SELECT "fillmore"       , "curly"      , 32           UNION
  SELECT "grover"         , "short"      , 28           UNION
  SELECT "herbert"        , "curly"      , 31;

CREATE TABLE sizes AS
  SELECT "toy" AS size, 24 AS min, 28 AS max UNION
  SELECT "mini"       , 28       , 35        UNION
  SELECT "medium"     , 35       , 45        UNION
  SELECT "standard"   , 45       , 60;
```

Your tables should still perform correctly even if the values in these tables change. For example, if you are asked to list all dogs with a name that starts with h, you should write:

```
SELECT name FROM dogs WHERE "h" <= name AND name < "i";
```

Instead of assuming that the `dogs` table has only the data above and writing

```
SELECT "herbert";
```

The former query would still be correct if the name `grover` were changed to `hoover` or a row was added with the name `harry`.

Q1: By Parent Height

Create a table `by_parent_height` that has a column of the names of all dogs that have a `parent`, ordered by the height of the parent dog from tallest parent to shortest parent.

```
-- All dogs with parents ordered by decreasing height of their parent
CREATE TABLE by_parent_height AS
  SELECT "REPLACE THIS LINE WITH YOUR SOLUTION";
```

For example, `fillmore` has a parent `eisenhower` with height 35, and so should appear before `grover` who has a parent `fillmore` with height 32. The names of dogs with parents of the same height should appear together in any order. For example, `barack` and `clinton` should both appear at the end, but either one can come before the other.

```
sqlite> select * from by_parent_height;
herbert
fillmore
abraham
delano
grover
barack
clinton
```

Use Ok to test your code:

```
python3 ok -q by_parent_height
```

Q2: Size of Dogs

The Fédération Cynologique Internationale classifies a standard poodle as over 45 cm and up to 60 cm. The `sizes` table describes this and other such classifications, where a dog must be over the `min` and less than or equal to the `max` in `height` to qualify as `size`.

Create a `size_of_dogs` table with two columns, one for each dog's `name` and another for its `size`.

```
-- The size of each dog
CREATE TABLE size_of_dogs AS
  SELECT "REPLACE THIS LINE WITH YOUR SOLUTION";
```

The output should look like the following:

```
sqlite> select * from size_of_dogs;
abraham|toy
barack|standard
clinton|standard
delano|standard
eisenhower|mini
fillmore|mini
grover|toy
herbert|mini
```

Use Ok to test your code:

```
python3 ok -q size_of_dogs
```

Q3: Sentences

There are two pairs of siblings that have the same size. Create a table that contains a row with a string for each of these pairs. Each string should be a sentence describing the siblings by their size.

```
-- Filling out this helper table is optional
CREATE TABLE siblings AS
  SELECT "REPLACE THIS LINE WITH YOUR SOLUTION";

-- Sentences about siblings that are the same size
CREATE TABLE sentences AS
  SELECT "REPLACE THIS LINE WITH YOUR SOLUTION";
```

Each sibling pair should appear only once in the output, and siblings should be listed in alphabetical order (e.g. `"barack and clinton..."` instead of `"clinton and barack..."`), as follows:

```
sqlite> select * from sentences;
The two siblings, barack and clinton, have the same size: standard
The two siblings, abraham and grover, have the same size: toy
```

> **Hint**: First, create a helper table containing each pair of siblings. This will make comparing the sizes of siblings when constructing the main table easier.
>
> **Hint**: If you join a table with itself, use `AS` within the `FROM` clause to give each table an alias.
>
> **Hint**: In order to concatenate two strings into one, use the `||` operator.

Use Ok to test your code:

```
python3 ok -q sentences
```

Q4: Low Variance

We want to create a table that contains the height range (difference between maximum and minimum height) of all dogs that share a fur type. However, we'll only consider fur types where each dog with that fur type is within 30% of the average height of all dogs with that fur type.

For example, if the average height for short-haired dogs is 10, then in order to be included in our output, all dogs with short hair must have a height of at most 13 and at least 7.

To achieve this, we can use `MIN`, `MAX`, and `AVG`. For this problem, we'll want to find the average height and make sure that:

- There are no heights smaller than 0.7 of the average.
- There are no heights greater than 1.3 of the average.

Your output should first include the fur type and then the height range for the fur types that meet this criteria.

```
-- Height range for each fur type where all of the heights differ by no more than 30% fror
CREATE TABLE low_variance AS
  SELECT "REPLACE THIS LINE WITH YOUR SOLUTION";

-- Example:
SELECT * FROM low_variance;
-- Expected output:
-- curly|1
```

Explanation: The average height of long-haired dogs is 39.7, so the low variance criterion requires the height of each long-haired dog to be between 27.8 and 51.6. However, `abraham` is a long-haired dog with height 26, which is outside this range. For short-haired dogs, `barack` falls outside the valid range (check!). Thus, neither short nor long haired dogs are included in the output. There are two curly haired dogs: `fillmore` with height 32 and `herbert` with height 31. This gives a height range of 1.

Use Ok to test your code:

```
python3 ok -q low_variance
```

Submit

Make sure to submit this assignment by uploading any files you've edited **to the appropriate Gradescope assignment.** For a refresher on how to do this, refer to Lab 00 (https://cs61a.org/lab/lab00/#submit-with-gradescope).

Make sure to submit `hw10.sql` to the autograder!

Exam Practice

The following are some SQL exam problems from previous semesters that you may find useful as additional exam practice.

1. Fall 2019 Final, Question 10: Big Game (https://cs61a.org/exam/fa19/final/61a-fa19-final.pdf#page=11)
2. Summer 2019 Final, Question 8: The Big SQL (https://cs61a.org/exam/su19/final/61a-su19-final.pdf#page=11)
3. Fall 2018 Final, Question 7: SQL of Course (https://inst.eecs.berkeley.edu/~cs61a/fa18/assets/pdfs/61a-fa18-final.pdf#page=9)