CS 61A: Solutions for Homework 10

Due by 11:59pm on Wednesday, 12/3

Solutions: You can find the Python file with solutions for all questions here.

Readings: You might find the following references useful:

- Section 4.3

Table of Contents

To complete this homework assignment, you will need to use Sqlite version 3.8.3 or greater. You can

- Download Sqlite3 and install it on your own computer
- Run `sqlite3` from an instructional machine using your course account
- Use this online Sqlite interpreter

After installing Sqlite, you can run the starter file using the command:

```
sqlite3 -init hw10.sql
```

Data

In each question below, you will define a select statement that processes 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 select statement should still perform correctly even if the values in these tables are changed. 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`.

Question 1

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 a `size`.

Select the names of all dogs that are either `toy` or `mini` sized.

```
-- The names of all "toy" and "mini" dogs
select name from dog_size
  where size="toy" or size="mini";
-- Expected output:
--   abraham
--   eisenhower
--   fillmore
--   grover
--   herbert
```

Hint: First create a table with two columns, one for each dog's `name` and another for its `size`. You can use this table in future questions as well.

Question 2

Select the names of all dogs that have a `parent`, ordered by the height of the parent from tallest parent to shortest parent.

```
-- All dogs with parents ordered by decreasing height of their p
select child from parents, dogs where name = parent order by -he
-- Expected output:
--   herbert
--   fillmore
--   abraham
--   delano
--   grover
--   barack
--   clinton
```

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.

Question 3

Select a single string for every pair of siblings that have the same size. Each value should be a sentence describing the siblings by their size, as shown in the expected output below.

```
-- Sentences about siblings that are the same size
with
  siblings(first, second) as (
    select a.child, b.child from parents as a, parents as b
      where a.parent = b.parent and a.child < b.child
  )
select first || " and " || second || " are " || a.size || " sibl
  from siblings, dog_size as a, dog_size as b
  where a.size = b.size and a.name = first and b.name = second;
-- Expected output:
--   barack and clinton are standard siblings
--   abraham and grover are toy siblings
```

Each sibling pair should appear only once in alphabetical order.

Hint: First use a `with` clause to create a local table of siblings. Comparing the size of siblings will then be simplified.

Hint: If you join a table with itself, use `as` within the `from` clause to give each table an alias.

Question 4

When dogs are stacked on top of one another, the total height of the stack is the sum of the heights of the dogs.

Select a two-column table describing all stacks of dogs at least 170 cm high. The first column should contains a comma-separated list of dogs in the stack, and the second column should contain the total height of the stack. Order the stacks in increasing order of total height.

```
-- Ways to stack 4 dogs to a height of at least 170, ordered by
with
  sums(names, total, n, max) as (
    select name, height, 1, height from dogs union
    select names || ", " || name, total+height, n+1, height
      from sums, dogs
      where n < 4 and max < height
  )
select names, total from sums where n=4 and total>=170 order by
-- Expected output:
--   abraham, delano, clinton, barack|171
--   grover, delano, clinton, barack|173
--   herbert, delano, clinton, barack|176
--   fillmore, delano, clinton, barack|177
--   eisenhower, delano, clinton, barack|180
```

A valid stack of dogs includes each dog only once, and the dogs should be listed in increasing order of height within the stack. **Assume that no two dogs have the same height**.

Hint: Use a `with` clause to create a recursive table with additional columns, such as the number of dogs that have been stacked and information about the last dog added (to control the dog order). Then, select the rows and columns from this larger table to generate the final solution.

Hint: Use height comparisons to ensure that dogs are not repeated in a stack.

Hint: Generating the comma-separated list of dogs is easier if your base case includes the name of one dog without any commas before or after it, rather than no dogs at all.

Question 5

This question is optional but recommended for practice. You can receive full credit for the homework without attempting this problem.

A non-parent relation is either an ancestor that is not a parent (such as a grandparent or great-grandparent) or a descendent that is not a child (such as a grandchild or great-grandchild). Siblings are *not* relations under this definition.

Select all pairs that form non-parent relations ordered by the difference in height between one dog and the other.

```
-- All non-parent relations ordered by height difference
with
  grandparents(grandog, grandpup) as (
    select a.parent, b.child from parents as a, parents as b
      where a.child = b.parent
  ),
  ancestors(ancestor, descendent) as (
    select grandog, grandpup from grandparents union
    select ancestor, child from ancestors, parents
      where parent = descendent
  ),
  relations(first, second) as (
    select ancestor, descendent from ancestors union
    select descendent, ancestor from ancestors
  )
select first, second from relations, dogs as f, dogs as s
  where first = f.name and second = s.name order by f.height-s.h
-- Expected output:
--   fillmore|barack
--   eisenhower|barack
--   fillmore|clinton
--   eisenhower|clinton
--   eisenhower|delano
--   abraham|eisenhower
--   grover|eisenhower
--   herbert|eisenhower
--   herbert|fillmore
--   fillmore|herbert
--   eisenhower|herbert
--   eisenhower|grover
--   eisenhower|abraham
--   delano|eisenhower
--   clinton|eisenhower
--   clinton|fillmore
--   barack|eisenhower
--   barack|fillmore
```

The shortest paired with the tallest should appear first, and the tallest paired with the shortest should appear last. If two pairs have the same height difference, they may appear

together in any order.