Assignment #1: Declarative SQL Programming

The goal of this assignment is to write several SQL queries that will answer questions over the database used by the imaginary Southern Sierra Wildflower Club (SSWC), an organization whose members are interested in observing wildflowers in their native habitat in the southern part of the Sierra Nevada mountains of California.

The database maintained by the club has four tables:

```
SIGHTINGS (SIGHT_ID, NAME, PERSON, LOCATION, SIGHTED)
FEATURES (LOC_ID, LOCATION, CLASS, LATITUDE, LONGITUDE, MAP, ELEV)
FLOWERS (FLOW_ID, GENUS, SPECIES, COMNAME)
PEOPLE (PERSON_ID, PERSON)
```

- `SIGHTINGS` gives information that describes every time that a member of the club observes one of the wildflowers described in the table `FLOWERS`. `NAME` tells the name of the flower observed; `PERSON` describes who saw the flower; `LOCATION` tells the name of a nearby geographical feature where the flower was seen, and `SIGHTED` tells the day when the flower was seen.

- `FEATURES` lists the various locations where flowers have been observed. `LOCATION` is the name of the place; `CLASS` is the type of place (there are several types, such as Summit, Mine, Locale, etc.); `LATITUDE` and `LONGITUDE` describe where on the surface of the earth the locations are found (if you are not familiar with the concepts of latitude and longitude, you might want to do a web search on them; the first is like an x-coordinate on the Earth's surface, and the second is like a y-coordinate); MAP tells the name of the topographic map where the feature can be found, and `ELEV` tells the height of the feature.

- `FLOWERS` lists all of the flowers that the members of the SSWC try to find. `GENUS` and SPECIES give the scientific name for the flower, and `COMNAME` gives the non-scientific name (`SIGHTING.NAME` is a foreign key into `FLOWER.COMNAME`).

- Finally, PEOPLE lists all of the people in the club.

Getting Started

To begin on the assignment, download and unzip the SSWC.zip archive. You will then want to log onto the class SQLServer instance, create your database using the contents of the zip archive, and then you can start writing your queries.

You can find the instructions on how to connect to the database here. Once you can connect and you create your own database (**make sure to create and use your own!**) you will first run createtables.sql, and then run sightings.sql, flowers.sql, people.sql, and features.sql.

If you ever want to destroy those tables, you can just run droptables.sql. You will then need to run those scripts again to recreate them.

Note that in SQLServer, all queries are followed by a ";". To compile and execute the query, you need to type "go".

So, once you create the SIGHTINGS table, you might type:

```
SELECT *
FROM SIGHTINGS;
GO
```

Queries

Your assignment is to write SQL queries that answer the following questions, somewhat sorted in order of difficulty from easiest to most difficult.

1. Who has seen a flower at Alaska Flat?
2. Who has seen the same flower at both Moreland Mill and at Steve Spring?
3. What is the scientific name for each of the different flowers that have been sighted by either Michael or Robert below 7250 feet in elevation?
4. Which maps hold a location where someone has seen Alpine penstemon in June?
5. Which genus have more than one species recorded in the SSWC database?
6. How many mines are on the Claraville map?
7. What is the furthest north location that James has seen a flower? "Furthest north" means highest latitude.
8. Who has not seen a flower at a location of class Spring?
9. Who has seen flowers at the least distinct locations, and how many distinct flowers was that?
10. For those people who have seen all of the flowers in the SSWC database, what was the date at which they saw their last unseen flower? In other words, at which date did they finish observing all of the flowers in the database?

11. For Tim, compute the fraction of his sightings on a per-month basis. For example, we might get {(September, .12), (October, .74), (November, .14)}. The fractions should add up to one across all months.

12. Whose set of flower sightings is most similar to Michael's? Set similarity is here defined in terms of the Jaccard Index, where JI (A, B) for two sets A and B is (size of the intersection of A and B) / (size of the union of A and B). A larger Jaccard Index means more similar.

What To Turn In

Submit to Canvas **a single *netid_assignment1.zip*** file containing the following:

1. A "*netid_queries.sql*" file with all your queries in it. We will use this file to run your queries.
2. A *"netid_results.pdf"* file with both your queries and the results you got when running them.

You should use your **actual Rice NetID** instead of *netid* in the file names.

Grading

Each question is worth 10 points. Points will be assigned as follows:

- 0 points: for a query not attempted, the query does not give any results, or it does not compile. You will also get 0 points if the results we get from running your query do not match the results in our *results.pdf* file.
- 5 points: the query compiles and runs successfully but the answer is incorrect.
- 10 points: the query compiles, runs successfully, and gives the right answer.

A Word of Caution

Start early! The average student might need **6 to 10 hours** to get all 12 queries to work. Furthermore, if SQLServer goes down for a couple hours right before the assignment is due, you will not be in trouble since you finished it early. A couple of hours of downtime will not affect the due date of the assignment, so again, start early!

Two Final Notes

Several of these queries will be nearly impossible to write without breaking them up into pieces. The way to do this is with the SQL `CREATE VIEW` command.

Here's a big tip. If you are not using an IDE and are instead using some other software to connect and interact with the database via a command prompt, do not type your queries directly into the SQLServer prompt.

Instead, type them into your favorite text editor or word processor, and then copy and paste your query into the prompt. That way, when your query does not work (it usually won’t the first time!), you don't have to retype it. You can simply edit your query and then re-paste it. This may seem obvious, but many people would not do this without first being advised to do it!