HW 2

CS 216, Everything Data, Spring 2020

DUE: Monday Feb. 3 by 4:40 pm (class time)

This assignment is in two parts. In the first part, you will initialize a local relational database using sqlite3 with information about the United States Congress. In the second part, you will write sql queries to answer a number of questions. You will include all of your answers for this assignment within this notebook. You will then convert your notebook to a .pdf and a .py file to submit to gradescope (submission instructions are included at the bottom).

Please take note of the course collaboration policy (https://sites.duke.edu/compsci216s2020/policies/). You may work alone or with a single partner. If you work with a partner, you may not split up the assignment; you should work together in-person or complete parts independently and come together to discuss your solutions. In either case, you are individually responsible for your work, and should understand everything in your submission.

Name and NetID

Nathan Kim (njk24) Eugene Kim (yk171)

Part 1: Getting Started

We will use the `SQLite` command line interface for working with the database in this assignment. First we will need to set up the necessary tools.

Terminal

You will need access to a unix style terminal (or bash shell), the basic non-graphical interface with which all computer scientists (and likely all data scientists) need to be familiar. If your operating system is a Linux distribution or any MacOS distribution, this part should be easy for you: simply open the terminal application and you are looking at a bash shell.

If your device uses Windows 10, a unix style terminal is not included by default (the Windows Command Prompt is not the same). You have some options. The easiest is likely to use sqlite3.exe to open a command line as discussed at https://www.sqlite.org/cli.html (https://www.sqlite.org/cli.html); you can find the relevant file at https://www.sqlite.org/download.html (https://www.sqlite.org/download.html) under the tools download for Precompiled Binaries for Windows. If you wish, you can also explore tools for obtaining a unix style terminal (among other features) on Windows 10 such as https://www.microsoft.com/en-us/p/ubuntu-1804-lts/9n9tngvndl3q?activetab=pivot:overviewtab (https://www.microsoft.com/en-us/p/ubuntu-1804-lts/9n9tngvndl3q?activetab=pivot:overviewtab) or https://www.microsoft.com/en-us/p/windows-terminal-preview/9n0dx20hk701?activetab=pivot:overviewtab (https://www.microsoft.com/en-us/p/windows-terminal-preview/9n0dx20hk701?activetab=pivot:overviewtab).

If you have never used a unix style terminal before, we strongly reccomend that you look at this very short tutorial (created by a previous year's CS 216 student) to familiarize yourself with the basic look and function: https://www2.cs.duke.edu/courses/spring17/compsci216/help/shell-tutorial.pdf (https://www2.cs.duke.edu/courses/spring17/compsci216/help/shell-tutorial.pdf).

SQLite

Now we need to confirm that you have SQLite set up on your device. SQLite is included in most distributions of Python, so most of you shouldn't need to do anything. Simply open your terminal and type "sqlite3" then press enter (or just run sqlite3.exe if you are using the sqlite command line tool for Windows). You should see sqlite3 load as in the getting started section of https://sqlite.org/cli.html (https://sqlite.org/cli.html). You can quit the SQLite command line interface and return to the standard unix terminal at any time by pressing ctrl+d (that is, hold down the ctrl button and then press d).

Note that `SQLite` is one of many implementations of relational databases using the language `SQL`. There are abundant resources and references for learning SQL, but you might consider viewing these resources for an introduction to `SQL` from a `SQLite` perspective:

- Getting started guide for the `SQLite` command line interface: https://sqlite.org/cli.html (https://sqlite.org/cli.html)
- Tutorial and reference: https://www.w3resource.com/sqlite/index.php (https://www.w3resource.com/sqlite/index.php)
- Another tutorial and reference: https://www.tutorialspoint.com/sqlite/index.htm (https://www.tutorialspoint.com/sqlite/index.htm)

Once you have successfully opened the `SQLite` command line interface, you can move on and start loading the database. Otherwise you will need to go to https://www.sqlite.org/download.html (https://www.sqlite.org/download.html) and install sqlite3 on your device.

Loading the Congress Database

First, you will need to download the folder from box containing hw 2 and all of the related files to a working directory (folder) on your device (make sure to unzip the folder if it downloads in a compressed format). Once you have done that, open up your terminal, and navigate to the same working directory; you should be able to see `load.sql` and a folder titled `load`. Now you can open your SQLite command line tool (simply type "sqlite3" and press enter). To create and load the database, simply enter ".read load.sql" into the SQLite command line tool (should take just a couple of seconds to load).

Congratulations, you now have a local (on your device only) relational database containing the history of the US Congress through the 115th Congress (which ended in Jan. 2019). You see a visual representation of this database in the `database_visual_schema.pdf` document included on box. You can query this database using sql commands, and there are two ways to do so. First, you can enter commands directly into the command line interface. Second, you can execute a query stored in a local file using the `.read` command. We have included a few files with SQL queries from class for you to try on your own. For instance,

```
.read price-pelosi.sql
```

would result in the output:

```
agree   total   percent
------- ------- --------------------
1411    2649    53.2653831634579086
```

What to Turn in

Nothing for Part 1

Part 2: SQL Queries on the Congress Database

Write SQL queries to answer the following questions. Note that some queries may be long enough to make it cumbersome to type them all directly into the command line tool, in case you need to edit them multiple times. We reccomend that you work with a two window setup: one window with the SQLite command line tool open and connected to the database, and another window open with a plain text editor where you write and edit your queries. You can either execute those queries by saving them as plain text files and using the .read command (e.g., save queryA.sql and then enter ".read queryA.sql" into the SQLite command line tool) or simply copying them into the SQLite command line tool. When you are satisfied that you have answered the question, simply write your answer and the sql query you used to arrive at that answer, along with any other explanation if needed.

Problem A

List all past and present female members of the Congress who were born in the 1970s. Give their first name, last name, and birthday.

In [ ]:

```
SELECT first_name, last_name, birthday
FROM persons
WHERE gender = 'F' AND (birthday>='1970-01-01' AND birthday<'1980-01-01'
);
```

```
first_name  last_name        birthday
----------  ---------------  ----------
Jaime       Herrera Beutler  1978-11-03
Kristi      Noem             1971-11-30
Martha      Roby             1976-07-27
Kyrsten     Sinema           1976-07-12
Grace       Meng             1975-10-01
Mia         Love             1975-12-06
Joni        Ernst            1970-07-01
Nanette     Barragán         1976-09-15
Stephanie   Murphy           1978-09-16
Jenniffer   González-Colón   1976-08-05
Stephanie   Herseth Sandlin  1970-12-03
Gabrielle   Giffords         1970-06-08
```

In [ ]:

```
### Problem B
Which state is represented by the most of the 25 youngest current members of congress
(senators/representatives)? How many of those 25 members represent that state?

SELECT first_name, last_name, birthday, state
FROM cur_members
ORDER BY birthday DESC LIMIT 25;
```

first_name	last_name	birthday	state
Elise	Stefanik	1984-07-02	NY
Mike	Gallagher	1984-03-03	WI
Trey	Hollingswo	1983-09-12	IN
Matt	Gaetz	1982-05-07	FL
Tulsi	Gabbard	1981-04-12	HI
Eric	Swalwell	1980-11-16	CA
Joseph	Kennedy	1980-10-04	MA
Brian	Mast	1980-07-10	FL
Jason	Smith	1980-06-16	MO
Ruben	Kihuen	1980-04-25	NV
Justin	Amash	1980-04-18	MI
Carlos	Curbelo	1980-03-01	FL
Lee	Zeldin	1980-01-30	NY
Ruben	Gallego	1979-11-20	AZ
Jim	Banks	1979-07-16	IN
Scott	Taylor	1979-06-27	VA
Pete	Aguilar	1979-06-19	CA
Jaime	Herrera Be	1978-11-03	WA
Seth	Moulton	1978-10-24	MA
Stephanie	Murphy	1978-09-16	FL
Ron	DeSantis	1978-09-14	FL
Adam	Kinzinger	1978-02-27	IL
Darren	Soto	1978-02-25	FL
Will	Hurd	1977-08-19	TX
Markwayne	Mullin	1977-07-26	OK

Got the 25 youngest current members. Last birthday was 1977-07-26/

```
SELECT birthday, state, COUNT(state)
FROM cur_members
WHERE birthday >= '1977-07-26'
GROUP BY state;
```

birthday	state	COUNT(state)
1979-11-20	AZ	1
1980-11-16	CA	2
1978-09-14	FL	6
1981-04-12	HI	1
1978-02-27	IL	1
1979-07-16	IN	2
1980-10-04	MA	2
1980-04-18	MI	1
1980-06-16	MO	1

```
1980-04-25  NV          1
1980-01-30  NY          2
1977-07-26  OK          1
1977-08-19  TX          1
1979-06-27  VA          1
1978-11-03  WA          1
1984-03-03  WI          1
```

Florida is the most represented.

In [ ]:

```
### Problem C
List all North Carolina Democratic senators (past and present), together
with their terms as
a North Carolina senator.  Output only first name, last name, birthday,
as well as the start
and end dates of the term.  Order the results by start dates, most recen
t first.  Here are some
example output rows (the answer has more):

SELECT p.first_name, p.last_name, p.birthday, r.start_date, r.end_date
FROM persons p, person_roles r
WHERE r.party = 'Democrat' AND r.state= 'NC' AND r.type = 'sen' AND p.id
= r.person_id
ORDER BY r.start_date DESC;
```

first_name	last_name	birthday	start_date	end_date
Kay	Hagan	1953-05-26	2009-01-06	2015-01-03
John	Edwards	1953-06-10	1999-01-06	2005-01-03
James	Sanford	1917-08-20	1987-01-06	1993-01-03
James	Sanford	1917-08-20	1986-11-05	1987-01-03
Robert	Morgan	1925-10-05	1975-01-14	1981-01-03
Samuel	Ervin	1896-09-27	1969-01-03	1975-01-03
Benjamin	Jordan	1896-09-08	1967-01-10	1973-01-03
Samuel	Ervin	1896-09-27	1963-01-09	1969-01-03
Benjamin	Jordan	1896-09-08	1961-01-03	1967-01-03
Benjamin	Jordan	1896-09-08	1958-01-01	1961-01-03
Samuel	Ervin	1896-09-27	1957-01-03	1963-01-03
William	Scott	1896-04-17	1954-01-01	1959-01-03
Samuel	Ervin	1896-09-27	1954-01-01	1957-01-03
Alton	Lennon	1906-08-17	1953-01-03	1955-01-03
Clyde	Hoey	1877-12-11	1951-01-03	1955-01-03
Willis	Smith	1887-12-19	1950-01-01	1953-12-31
Joseph	Broughton	1888-11-17	1949-01-03	1949-03-06
Frank	Graham	1886-10-14	1949-01-03	1951-01-03
Joseph	Broughton	1888-11-17	1948-12-31	1949-01-03
William	Umstead	1895-05-13	1946-01-01	1949-01-03
Clyde	Hoey	1877-12-11	1945-01-03	1951-01-03
Josiah	Bailey	1873-09-14	1943-01-06	1947-01-03
Robert	Reynolds	1884-06-18	1939-01-03	1945-01-03
Josiah	Bailey	1873-09-14	1937-01-05	1943-01-03
Robert	Reynolds	1884-06-18	1933-01-03	1939-01-03
Robert	Reynolds	1884-06-18	1932-12-05	1933-01-03
Josiah	Bailey	1873-09-14	1931-12-07	1937-01-03
Cameron	Morrison	1869-10-05	1930-01-01	1933-03-03
Lee	Overman	1854-01-03	1927-12-05	1930-12-12
Furnifold	Simmons	1854-01-20	1925-12-07	1931-03-03
Lee	Overman	1854-01-03	1921-04-11	1927-03-04
Furnifold	Simmons	1854-01-20	1919-05-19	1925-03-03
Lee	Overman	1854-01-03	1915-12-06	1921-03-03
Furnifold	Simmons	1854-01-20	1913-04-07	1919-03-03
Lee	Overman	1854-01-03	1909-03-15	1915-03-03
Furnifold	Simmons	1854-01-20	1907-12-02	1913-03-03
Lee	Overman	1854-01-03	1903-11-09	1909-03-03
Furnifold	Simmons	1854-01-20	1901-12-02	1907-03-03
Thomas	Jarvis	1836-01-18	1894-01-01	1895-12-31

```
Zebulon     Vance       1830-05-13  1891-12-07  1895-03-03
Matt        Ransom      1826-10-08  1889-12-02  1895-03-03
Zebulon     Vance       1830-05-13  1885-12-07  1891-03-03
Matt        Ransom      1826-10-08  1883-12-03  1889-03-03
Zebulon     Vance       1830-05-13  1879-03-18  1885-03-03
Matt        Ransom      1826-10-08  1877-10-15  1883-03-03
Augustus    Merrimon    1830-09-15  1873-12-01  1879-03-03
Matt        Ransom      1826-10-08  1872-01-01  1877-03-03
Thomas      Bragg       1810-11-09  1859-12-05  1861-12-31
Thomas      Clingman    1812-07-27  1858-01-01  1861-03-03
Asa         Biggs       1811-02-04  1855-12-03  1859-03-03
David       Reid        1813-04-19  1854-01-01  1859-03-03
William     Haywood     1801-10-23  1843-12-04  1847-03-03
Robert      Strange     1796-09-20  1836-01-01  1841-03-03
Bedford     Brown       1795-06-06  1835-12-07  1841-03-03
```

In [ ]:

```
### Problem D
Find the past and present members of congress from North Carolina who ha
ve served both in the
House and the Senate. Output the id, first_name and last_name.

SELECT p.id, p.first_name, p.last_name
FROM persons p, person_roles r
WHERE r.state='NC' AND r.person_id=p.id AND r.type = 'sen'
INTERSECT
SELECT p.id, p.first_name, p.last_name
FROM persons p, person_roles r
WHERE r.state='NC' AND r.person_id=p.id AND r.type = 'rep';

id          first_name  last_name
----------  ----------  ----------
B000456     Asa         Biggs
B000563     Timothy     Bloodworth
B000763     John        Branch
B000966     James       Broyhill
B001135     Richard     Burr
C000524     Thomas      Clingman
E000211     Samuel      Ervin
F000344     Jesse       Franklin
H000679     Clyde       Hoey
L000240     Alton       Lennon
M000034     Nathaniel   Macon
M000096     Willie      Mangum
M000993     Cameron     Morrison
R000144     David       Reid
S000415     Furnifold   Simmons
S000955     David       Stone
U000005     William     Umstead
V000021     Zebulon     Vance
```

In [ ]:

```
### Problem E
One of the important votes cast is for electing the Speaker of the House
. You can find these votes
by looking for value 'Election of the Speaker' in the 'question' column
of 'votes' table. For the
most recent such vote, who won the election? Output the vote_id, date, r
esult and count.

SELECT p.vote_id, v.date, v.result, COUNT(p.vote)
FROM votes v, person_votes p
WHERE v.question = 'Election of the Speaker' AND p.vote_id =v.id
GROUP BY p.vote
ORDER BY v.date DESC LIMIT 1;
```

```
vote_id         date                        result      COUNT(p.vote)
--------------  --------------------------  ----------  -------------
h581-114.2015   2015-10-29T10:46:00-04:00   Ryan (WI)   236
```

In [ ]:

```
### Problem F
For the election from problem E, can you identify who were the people wh
o recieved votes for
this position? (you may directly use useful values from previous answer)
.

SELECT DISTINCT v.result
FROM votes v, person_votes p
WHERE v.question = 'Election of the Speaker' AND p.vote_id =v.id
ORDER BY v.date;
```

```
result
----------
Boehner
Ryan (WI)
```

In [ ]:

```
### Problem G
How many members in each state voted for Paul Ryan for this position?

SELECT r.state, COUNT(DISTINCT v.person_id)
FROM person_votes v, person_roles r
WHERE v.vote='Ryan (WI)' AND v.vote_id = 'h581-114.2015' AND v.person_id
= r.person_id
GROUP BY r.state;
```

state	COUNT(DISTINCT v.person_id)
AK	1
AL	6
AR	4
AZ	4
CA	14
CO	4
FL	13
GA	10
IA	3
ID	2
IL	8
IN	7
KS	4
KY	4
LA	5
MD	1
ME	1
MI	9
MN	3
MO	6
MS	3
MT	1
NC	9
ND	1
NE	2
NH	1
NJ	6
NM	1
NV	3
NY	9
OH	12
OK	5
OR	1
PA	13
SC	6
SD	1
TN	7
TX	23
UT	4
VA	7
WA	4
WI	4
WV	3
WY	1

Problem H

In Lecture 2, we showed how queries computing vote correlations for David Price (D-NC) with Nancy Pelosi (D-CA) revealed unexpected results. Can you explain why it seems that Price votes so infrequently with Pelosi (when compared with another Represetative from NC, Butterfield)? What is the correct percentage of votes on which Price and Pelosi agreed?

For your convenience, the file 'price-pelosi.sql' contains the SQL query used in the class. You may edit this file to run variations of this query to help you debug.

In [ ]:
```
WITH votes_compare(vote_id, vote1, vote2) AS
(SELECT v1.vote_id, v1.vote, v2.vote
 FROM votes v, persons p1, persons p2, person_votes v1, person_votes v2
 WHERE v.chamber = 'h' AND (v.session = 2015 or v.session = 2016)
   AND p1.last_name = 'Price' AND p2.last_name = 'Pelosi'
   AND v1.person_id = p1.id AND v2.person_id = p2.id
   AND v1.vote_id = v2.vote_id AND v.id = v1.vote_id
   AND p1.first_name = 'David')
SELECT COUNT(*) AS agree,
       (SELECT COUNT(vote1=vote2) FROM votes_compare) AS total,
       COUNT(*)*100.00 / (SELECT COUNT(*) FROM votes_compare) AS percent
FROM votes_compare
WHERE vote1 = vote2;

agree       total       percent
----------  ----------  ----------------
1157        1324        87.3867069486405

Price's percentage was so low because it was taking into account the vot
es of multiple people with the last name
'Price'. This effectively skewed the data so that the total number of vo
tes under the last name 'Price' would be
much larger than what was intended.
```

Submitting HW 2

1. Double check that you have written all of your answers along with your supporting work in this notebook. Make sure you save the complete notebook.
2. Double check that your entire notebook runs correctly and generates the expected output. To do so, you can simply select Kernel -> Restart and Run All.
3. You will download two versions of your notebook to submit, a .pdf and a .py. To create a PDF, we reccomend that you select File --> Download as --> HTML (.html). Open the downloaded .html file; it should open in your web broser. Double check that it looks like your notebook, then print a .pdf using your web browser (you should be able to select to print to a pdf on most major web browsers and operating systems). Check your .pdf for readability: If some long cells are being cut off, go back to your notebook and split them into multiple smaller cells. To get the .py file from your notebook, simply select File -> Download as -> Python (.py) (note, we recognize that you may not have written any Python code for this assignment, but will continue the usual workflow for consistency).
4. Upload the .pdf to gradescope under hw2 report and the .py to gradescope under hw2 code. If you work with a partner, only submit one document for both of you, but be sure to add your partner using the group feature on gradescope (https://www.gradescope.com/help#help-center-item-student-group-members).

In [ ]: