**Objectives**

* Apply computing tools and techniques to solve problems at multiple levels of abstraction.
* Collaborate with others to gain insight, interpret data, and solve problems using computation.

**Part 0: Connecting to SQLite**

Please install[[](https://sqlitebrowser.org/) DB Browser for SQLite](https://sqlitebrowser.org/) on your computer.

In order to complete today's lab, you will need to open SQLite databases found in your Firefox profile folder. The location of your profile folder depends on your operating system:

* **Linux:** ~/.mozilla/firefox/<profile folder>
* **macOS:** ~/Library/Application Support/Firefox/Profiles/<profile folder>
* **Windows:** %APPDATA%\Mozilla\Firefox\Profiles\<profile folder>

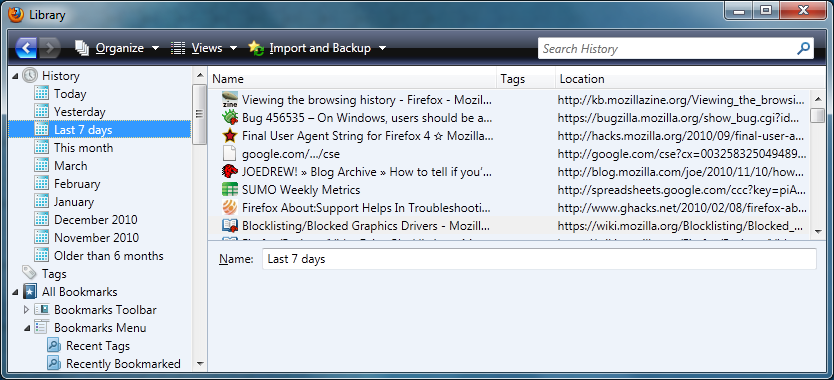
If you can not find the folder at first, search for firefox within your finder type window.

The name of your <profile folder> will look something like xxxxxxxx.default or xxxxxxxx.default-release, where xxxxxxxx are random letters and numbers. See the support article "[where firefox stores user data](https://support.mozilla.org/en-US/kb/profiles-where-firefox-stores-user-data)" for more details.

**Part 1: Your Firefox Profile**

**PRE-WORK.**

If you are not a firefox user, make sure you download it. Go to some sites and navigate and make some bookmarks first. If you are a firefox user…you can skip this.

1. Run the DB Browser app, and click the "Open Database" button on the toolbar. Navigate to your profile folder (see Part 0), and look for files like cookies.sqlite and places.sqlite. Use the "Database Structure" and "Browse Data" tabs to view the contents of each table stored in these files. Explore the other databases using this same process. *Answer #1 and #2 in your worksheet.*
2. 

Sometimes you will need to query two or more tables that have columns with the same name (e.g., id). You need to use the table name in that case so that SQL can tell which column you want. For example, in places.sqlite:

SELECT \*

FROM moz\_historyvisits, moz\_places

WHERE moz\_historyvisits.place\_id = moz\_places.id

Because the table.column syntax makes the code harder to read, SQL allows you to rename tables within the query. Often developers use a single letter name for this purpose, like this:

SELECT h.visit\_type, h.visit\_date, p.url

FROM moz\_historyvisits h, moz\_places p

WHERE h.place\_id = p.id

Note how p.url in the SELECT clause makes it easy to tell what table that column comes from. *Answer #3 in your worksheet.*

1. In addition to SELECT, FROM, and WHERE, there are many other clauses in SQL. For example, you can sort the query results:
2. SELECT \*
3. FROM moz\_places
4. ORDER BY visit\_count DESC, url

In this query, the output is first sorted by visit\_count in descending order and then sorted by url in ascending order (for records that have the same visit\_count). *Answer #4 in your worksheet.*

1. One of the most interesting features of SQL is *grouping and aggregation*, which allows you to do more than just filter data. For example, this query shows which servers you visit most:
2. SELECT rev\_host, sum(visit\_count)
3. FROM moz\_places
4. GROUP BY rev\_host
5. ORDER BY sum(visit\_count) DESC

GROUP BY essentially means "for each." This query finds all rows with the same rev\_host, and then calls the sum function to add up all their visit\_count values. The query then outputs a single row for each rev\_host with its total number of visits. *Answer #5 in your worksheet.*

**Part 2: Other Data Sets**



1. Download the [LAB 2 Data](https://github.com/jasonwnc/ds3002/raw/gh-pages/assets/Lab2.sqlite)database and open it in DB Browser. Explore the tables via the "Structure" and "Browse" tabs.
2. Write SQL statements to *answer the remaining questions* in your worksheet. For each one, include both your SQL code and the result.

To learn more about SQL, check out some of the many free [tutorials online](https://www.google.com/#q=sql+tutorial). The one at [w3schools.com](http://www.w3schools.com/sql/) is a great place to start.

**Part 3: Using Python**

1. Access the Lab 🡪 <https://towardsdatascience.com/python-sqlite-tutorial-the-ultimate-guide-fdcb8d7a4f30>
2. Complete this lab and upload a link to your Py file in Git

**Submission Instructions**

* Submit your GIT LINK to me and Dylan by Sunday Evening @11:00PM with your Worksheet answers.

WORKSHEET

1. What types of information does Firefox manage using SQLite? Briefly explain the contents of two or three databases.

Firefox manages the persistent and current session data within SQLite using the places database - including things like previous browsing history and bookmarks - everything that consists of the state of sites/”places” the user has visited. It includes all the relevant data about those sites - URL, title, description, etc.

2. In what database and tables are your bookmarks stored? Why do you think

Firefox stores the title and url separately?

In the “places.sqlite” database, in the “moz\_bookmarks” table. Firefox stores the title and URL separately to allow the user to change the title, and because these attributes should be stored atomicly for easy retrieval. The User can change the “title” in the bookmarks table to change how their bookmark is displayed - the “fk” or foreign key connects to the “moz\_places” table which contains information about the sites URL and actual title. This latter table is universal information that should have its own table for a well formed database.

3. Write an SQL statement that selects your bookmarks. For each one, display

only the title, url, date added (as an integer), and visit count.

SELECT moz\_bookmarks.title, url, dateAdded, visit\_count

FROM moz\_bookmarks, moz\_places

WHERE moz\_bookmarks.fk = moz\_places.id

4. Write an SQL statement that lists the base domain, name, and value for each of your cookies in order of expiration date.

SELECT name, host, value

FROM moz\_cookies

ORDER BY expiry DESC

5. Write an SQL statement to show how many cookies you have for each domain.

Note that you will need to use the "count" function instead of sum.

SELECT host, COUNT(\*)

FROM moz\_cookies

GROUP BY host

ORDER BY COUNT(\*) DESC

6. What are the top five movies since the year 2000, in terms of adjusted

gross income?

SELECT \*

FROM movie

WHERE year > 2000

ORDER BY adjusted DESC

LIMIT 5

|  |
| --- |
| Avatar |
| Marvel's The Avengers |
| The Dark Knight |
| Shrek 2 |
| Spider-Man |

7. What are the title, author, and year of books from Russia in the top 100?

(For convenience, you can right-click the results and "Copy Rows as CSV.")

SELECT title, year, author

FROM book

WHERE country = 'Russia'

|  |  |  |
| --- | --- | --- |
| Stories | 1886 | Anton Chekhov |
| Crime and Punishment | 1866 | Fyodor Dostoevsky |
| The Idiot | 1869 | Fyodor Dostoevsky |
| The Possessed | 1872 | Fyodor Dostoevsky |
| The Brothers Karamazov | 1880 | Fyodor Dostoevsky |
| Dead Souls | 1842 | Nikolai Gogol |
| War and Peace | 1865?1869 | Leo Tolstoy |
| Anna Karenina | 1877 | Leo Tolstoy |
| The Death of Ivan Ilyich | 1886 | Leo Tolstoy |

8. What are the names and symbols of the periodic elements with an atomic mass of more than 280? Display the results from lightest to heaviest.

SELECT name, symbol

FROM periodic

WHERE mass > 280

ORDER BY mass ASC

|  |  |
| --- | --- |
| Ununtrium | Uut |
| Ununbium | Uub |
| Ununpentium | Uup |
| Ununquadium | Uuq |
| Ununhexium | Uuh |
| Ununoctium | Uuo |

9. Which of the top 15 songs were featured in the top 100 movies? (Hint: Figure out how to join the two tables, and then use AND clauses to filter by rank.)

SELECT \*

FROM song, movie

WHERE song.film = movie.title AND song.rank <= 15 AND movie.rank <= 100

|  |
| --- |
| Mrs. Robinson |
| When You Wish upon a Star |
| The Sound of Music |
| My Heart Will Go On |

10. Do you Python Lab

\*\*Used under the open commons from JMU CS101 Labs