MP1: Setting Up Database Environments

Note: All updates, FAQ, and bug fixes, if any, will be announced in this post. Please check it frequently.

Objectives

1. Set up three database systems– MySQL, MongoDB, and Neo4j– which we will learn and use throughout this class.
2. Create a database in each system from our toy dataset “FridayNight”.
3. Run a query to test that your setup is working.

Submission by Week 2 (May 28, 2023)

You will be provided with an online work document on which you can write your answers and communicate with the course staff for questions.
Please complete your work on the work document by the submission deadline, after which you will not have access to the document. **We will grade and provide feedback on the work doc.**

To help check the correctness of your results, please refer to the expected output.

Grading Rubric

You will receive your grades on your work doc and Coursera **in a week** after MP1 is due.

Problem	**Submission**	**Weight**	**Criteria**
Problem 1	1.1	10	Results are correct as expected.
	1.2	10	Results are correct as expected.
	1.3	10	Results are correct as expected.
	1.4	10	Results are correct as expected.
Problem 2	2.1	10	Results are correct as expected.
	2.2	10	Results are correct as expected.
	2.3	10	Results are correct as expected.
Problem 3	3.1	10	Results are correct as expected.
	3.2	10	Results are correct as expected.
	3.3	10	Results are correct as expected.

Problem 1: MySQL

Expected Time: 40 minutes

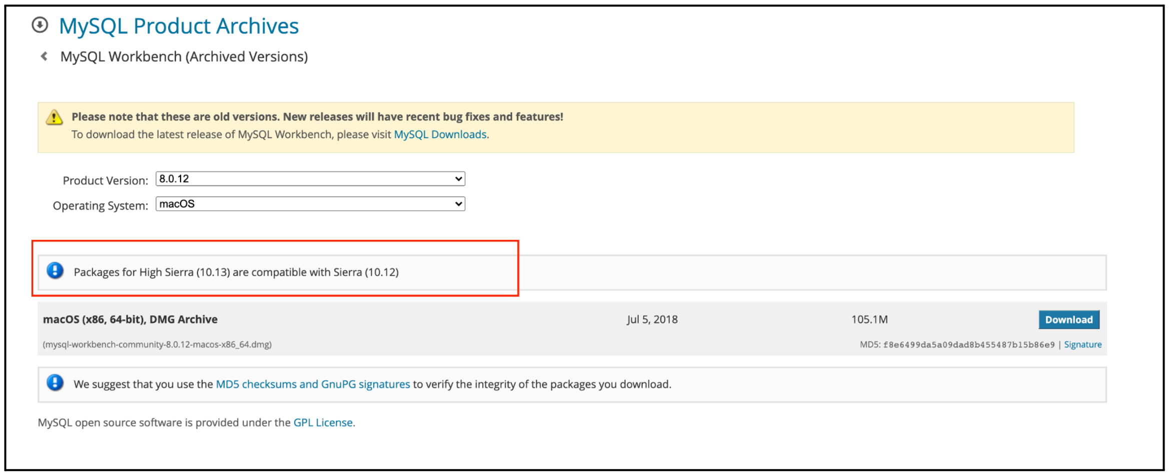
1. Install MySQL Server

1. Download MySQL server package.

   a. Download the server for MySQL.

      - For MacOS - download from here.
      - For Windows - download from here.

      Make sure to check if the product version is compatible with your current OS version. A simple way to do this is to select the product version and to check for a prompt that shows which version of the OS is compatible with the product selected.



MySQL Product Archives

MySQL Workbench (Archived Versions)

Please note that these are old versions. New releases will have recent bug fixes and features!
To download the latest release of MySQL Workbench, please visit MySQL Downloads.

Product Version: 8.0.12
Operating System: macOS

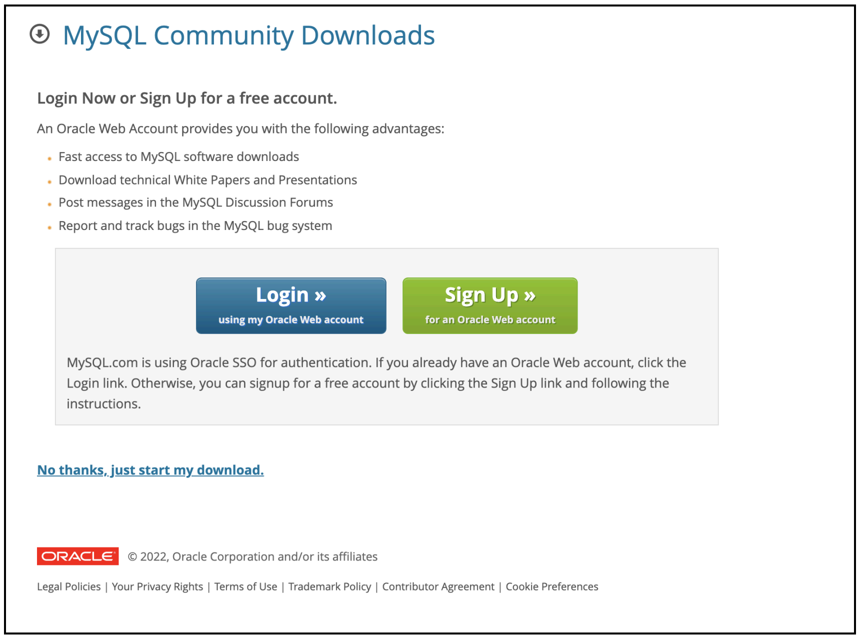
Packages for High Sierra (10.13) are compatible with Sierra (10.12)

macOS (x86, 64-bit), DMG Archive	Jul 5, 2018	105.1M	Download
(mysql-workbench-community-8.0.12-macos-x86_64.dmg)			MD5: f8e6499da5a09dad8b455487b15b86e9 | Signature

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

MySQL open source software is provided under the GPL License.

   b. Clicking download will take you to a new page. Please **Click** on "**No thanks, just start my download**".


MySQL Community Downloads

Login Now or Sign Up for a free account.

An Oracle Web Account provides you with the following advantages:

- Fast access to MySQL software downloads
- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system

Login » using my Oracle Web account
Sign Up » for an Oracle Web account

MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

No thanks, just start my download.

© 2022, Oracle Corporation and/or its affiliates

Legal Policies | Your Privacy Rights | Terms of Use | Trademark Policy | Contributor Agreement | Cookie Preferences

2. Install the package on your computer.

   Please follow the tutorials linked below as installation varies depending on the package and the OS version:

   - For MacOS: follow the instructions here.
   - For Windows: follow the instructions here.
   - For Ubuntu 20.04 - follow the instructions here under the section **Installing MySQL Server**.

   The following are some things to keep in mind when installing:

   a. When choosing a Setup type , you may choose to install all MySQL products or install only the Server. For the course purpose **we only need to install the Server and Workbench.**

   b. When the prompt for root password shows up, **set the password as *test_root*.** Note - the password can be of your choice but please make sure to save it as it cannot be recovered.

   c. During the installation ,you can choose to **Add a new User**. We highly recommend this.

2. Install MySQL Workbench

1. Download MySQL Workbench.

   a. If you already have MySQL Workbench installed, skip to step 3. Otherwise, download the MySQL Workbench for your OS version from here. **Make sure to check if the Workbench version** is compatible with your current OS version. We don't require you to install software with a specified version. You could choose the latest version, regardless of the version in this instruction.

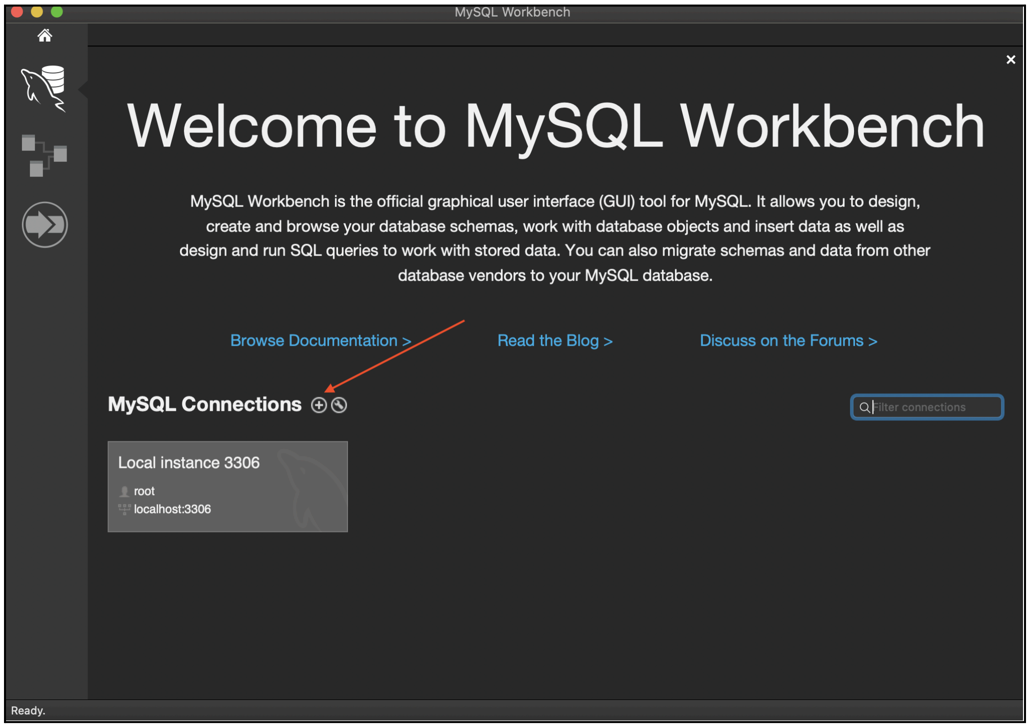
   b. Click on the package to install.

- For MacOS- follow the instructions here.
- For Windows - follow the instructions here.
- For Ubuntu 20.04 - follow the instructions here under the section **Installing MySQL WorkBench.**

2. Add connection.

    a. Open MySQL Workbench.

    b. Click on the + icon.The interface will vary depending on the package you have downloaded. Either way you should be able to find "Add connections" or simply a "+" icon as shown below.

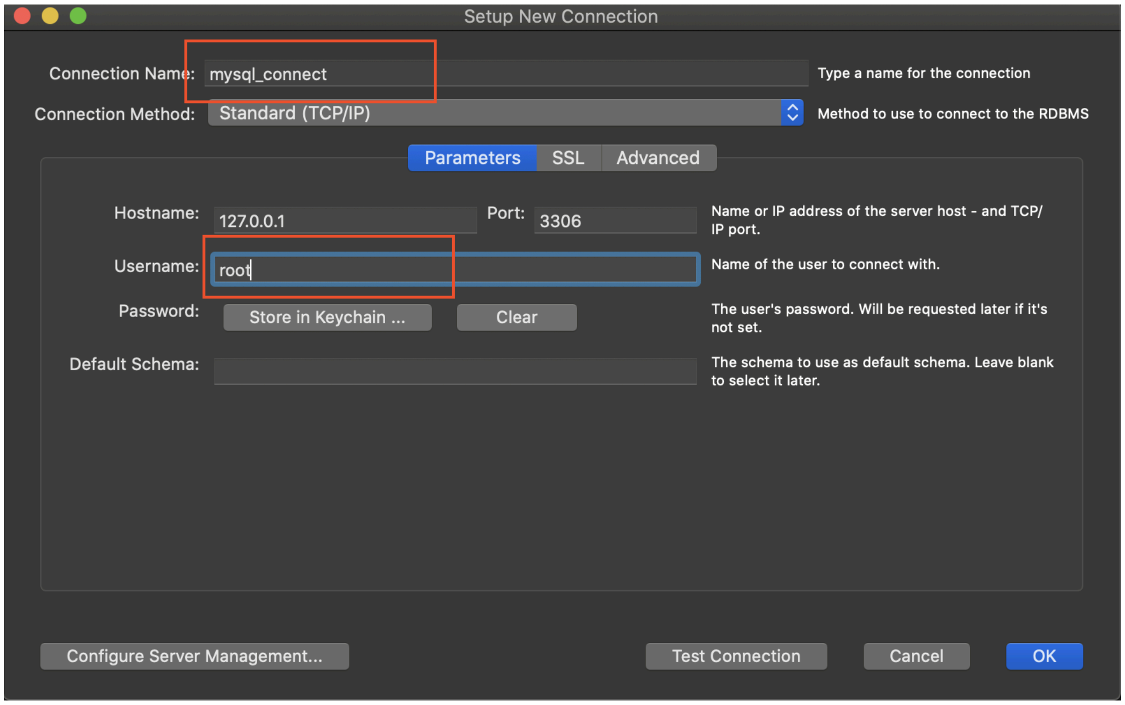


MySQL Workbench

Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

    c. This should open up a **Setup New Connection** box like this



Setup New Connection

Connection Name: mysql_connect — Type a name for the connection

Connection Method: Standard (TCP/IP) — Method to use to connect to the RDBMS

Parameters | SSL | Advanced

Hostname: 127.0.0.1 Port: 3306 — Name or IP address of the server host - and TCP/IP port.

Username: root — Name of the user to connect with.

Password: Store in Keychain ... Clear — The user's password. Will be requested later if it's not set.

Default Schema: — The schema to use as default schema. Leave blank to select it later.

d. Enter the username **root** or user from the server setup and connection name **mysql_connect**.

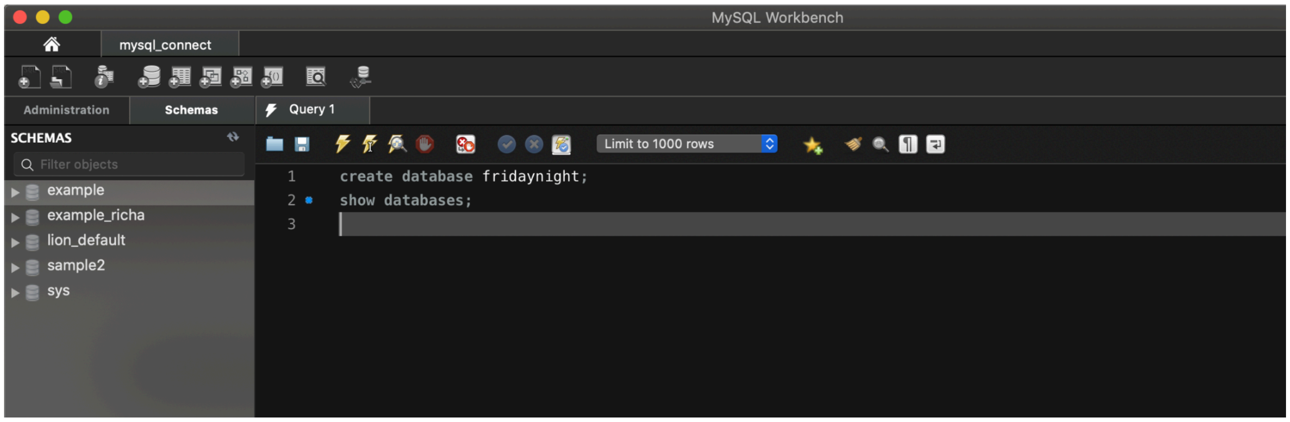
e. Click on **Test Connection** and enter the **password** you setup for the root user or new user to validate the details and then Click **OK**.
(The newly created connection should pop up under **MySQL Connections)**

3. Create Sample Database

1. Click on the connection you just created.

2. A query editor window will open. Now type the following commands in the query editor.

```
create database fridaynight;
show databases;
```

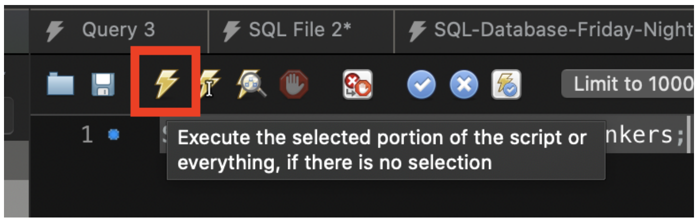


MySQL Workbench — mysql_connect — Query 1

SCHEMAS: example, example_richa, lion_default, sample2, sys

```
create database fridaynight;
show databases;
```

3. Click on the “thunder” icon in the top panel to execute the statements. You should then see the results displaying the list of databases in the system, including fridaynight.

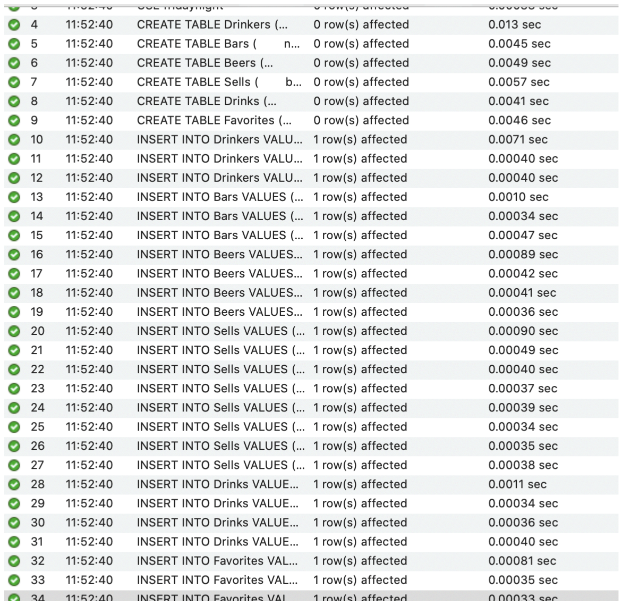


Execute the selected portion of the script or everything, if there is no selection

4. **(Submission 1.1)** Take a screenshot of the results. Paste it into your work document.

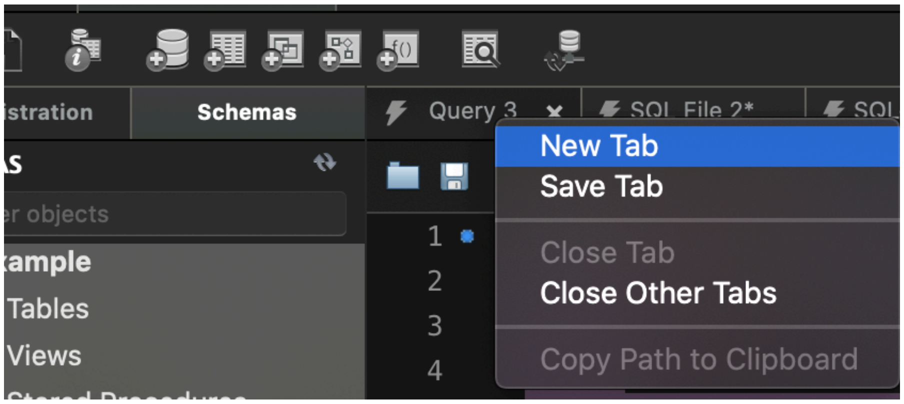
4. Populate Sample Database

1. Simply **paste** ALL the commands in [Appendix A](#) in the editor and then **execute**. This should create 6 tables and insert values into each table.



```
4   11:52:40  CREATE TABLE Drinkers (...       0 row(s) affected   0.013 sec
5   11:52:40  CREATE TABLE Bars (        n...  0 row(s) affected   0.0045 sec
6   11:52:40  CREATE TABLE Beers (...          0 row(s) affected   0.0049 sec
7   11:52:40  CREATE TABLE Sells (        b... 0 row(s) affected   0.0057 sec
8   11:52:40  CREATE TABLE Drinks (...         0 row(s) affected   0.0041 sec
9   11:52:40  CREATE TABLE Favorites (...      0 row(s) affected   0.0046 sec
10  11:52:40  INSERT INTO Drinkers VALU...     1 row(s) affected   0.0071 sec
11  11:52:40  INSERT INTO Drinkers VALU...     1 row(s) affected   0.00040 sec
12  11:52:40  INSERT INTO Drinkers VALU...     1 row(s) affected   0.00040 sec
13  11:52:40  INSERT INTO Bars VALUES (...     1 row(s) affected   0.0010 sec
14  11:52:40  INSERT INTO Bars VALUES (...     1 row(s) affected   0.00034 sec
15  11:52:40  INSERT INTO Bars VALUES (...     1 row(s) affected   0.00047 sec
16  11:52:40  INSERT INTO Beers VALUES...      1 row(s) affected   0.00089 sec
17  11:52:40  INSERT INTO Beers VALUES...      1 row(s) affected   0.00042 sec
18  11:52:40  INSERT INTO Beers VALUES...      1 row(s) affected   0.00041 sec
19  11:52:40  INSERT INTO Beers VALUES...      1 row(s) affected   0.00036 sec
20  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00090 sec
21  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00049 sec
22  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00040 sec
23  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00037 sec
24  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00039 sec
25  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00034 sec
26  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00035 sec
27  11:52:40  INSERT INTO Sells VALUES (...    1 row(s) affected   0.00038 sec
28  11:52:40  INSERT INTO Drinks VALUE...      1 row(s) affected   0.0011 sec
29  11:52:40  INSERT INTO Drinks VALUE...      1 row(s) affected   0.00034 sec
30  11:52:40  INSERT INTO Drinks VALUE...      1 row(s) affected   0.00036 sec
31  11:52:40  INSERT INTO Drinks VALUE...      1 row(s) affected   0.00040 sec
32  11:52:40  INSERT INTO Favorites VAL...     1 row(s) affected   0.00081 sec
33  11:52:40  INSERT INTO Favorites VAL...     1 row(s) affected   0.00035 sec
34  11:52:40  INSERT INTO Favorites VAL...     1 row(s) affected   0.00033 sec
```

2. Open a new editor by **clicking New Tab**.



3. Run the command

```
use fridaynight;
show tables;
```

4. **(Submission 1.2)** Take a screenshot of the results. Paste it into your work document.

5. Query Sample Database

1. Run the following query.

```
SELECT * FROM fridaynight.Drinkers;
```

2. **(Submission 1.3)** Take a screenshot of the results. Paste it into your work document.

3. Run the following query.

```
USE fridaynight;

SELECT "Drinkers" as TABLENAME,count(*) as COUNT from Drinkers
UNION
SELECT "Bars" as TABLENAME,count(*) as COUNT from Bars
UNION
SELECT "Beers" as TABLENAME,count(*) as COUNT from Beers
UNION
SELECT "Sells" as TABLENAME,count(*) as COUNT from Sells
UNION
SELECT "Drinks" as TABLENAME,count(*) as COUNT from Drinks
UNION
SELECT "Favorites" as TABLENAME,count(*) as COUNT from Favorites;
```

4. **(Submission 1.4)** Take a screenshot of the results. Paste it into your work document.

Problem 2: MongoDB

Expected Time: 30 minutes

1. Install MongoDB

1. Please find the instructions for installing **MongoDB Comunity Edition** on your system in the following link: https://www.mongodb.com/docs/manual/administration/install-community/

2. For Windows, you need to install **MongoDB Shell** separately because it is not included in the MongoDB Server.

2. Create Sample Database

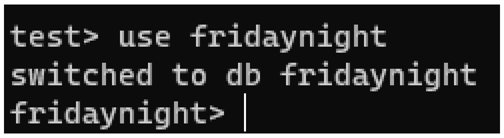
1. Open the **mongo shell** by running the following command in the terminal.

   ```
   mongosh
   ```

   If there is an error about command not found, check and add the command path to your environment "path" variables in your OS.

2. Create a database for our toy dataset by running the following command in the **mongo shell**.

   ```
   use fridaynight
   ```



```
test> use fridaynight
switched to db fridaynight
fridaynight>
```

3. Populate Sample Database

1. Paste all the commands in Appendix B directly into the mongo shell to create collections in the database.

2. List databases in the system. In mongo shell, run the following command:

   ```
   show dbs
   ```

3. **(Submission 2.1)** Take a screenshot of the results. Paste it into your work document.

4. Run the command

   ```
   show collections
   ```

5. **(Submission 2.2)** Take a screenshot of the results. Paste it into your work document.

4. Query Sample Database

1. Run the following query in mongo shell for the dataset we have just loaded in the database. You will be able to see the elements in the collection “Bars”.

   ```
   db.Bars.find().sort({name: 1})
   ```

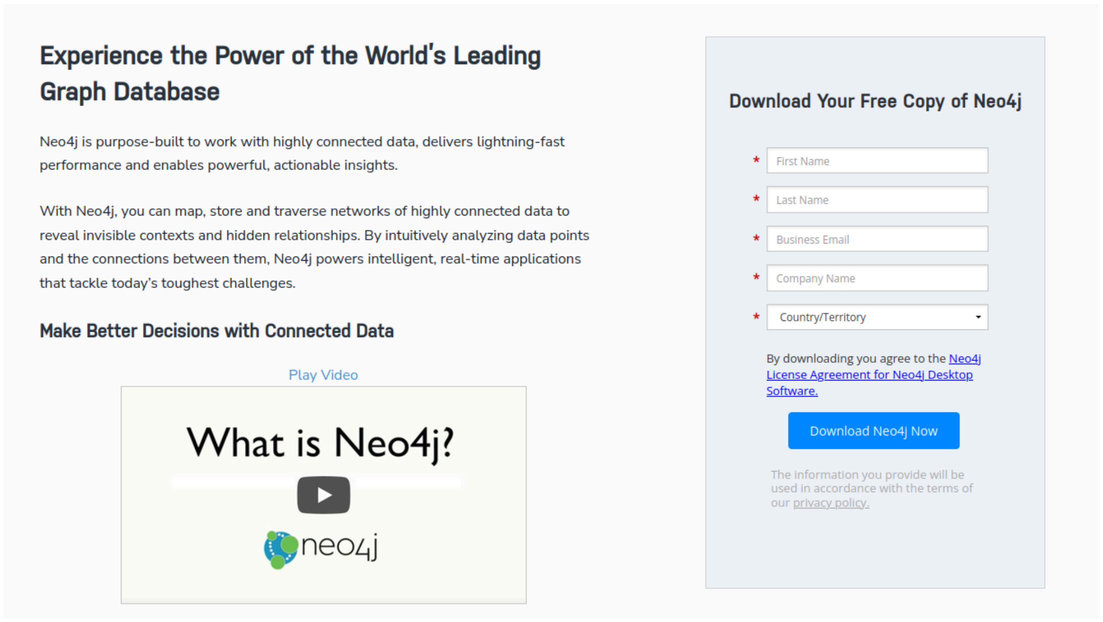
2. **(Submission 2.3)** Take a screenshot of the results. Paste it into your work document.

Problem 3: Neo4j

Expected Time: 30 minutes

1. Install Neo4j

1. Install Neo4j on your own operating system
   a. Go to https://neo4j.com/download-neo4j-now , **fill in the information box** on the right and then click **Download Neo4j Now**, this will jump to the download page.



Experience the Power of the World's Leading Graph Database

Neo4j is purpose-built to work with highly connected data, delivers lightning-fast performance and enables powerful, actionable insights.

With Neo4j, you can map, store and traverse networks of highly connected data to reveal invisible contexts and hidden relationships. By intuitively analyzing data points and the connections between them, Neo4j powers intelligent, real-time applications that tackle today's toughest challenges.

Make Better Decisions with Connected Data

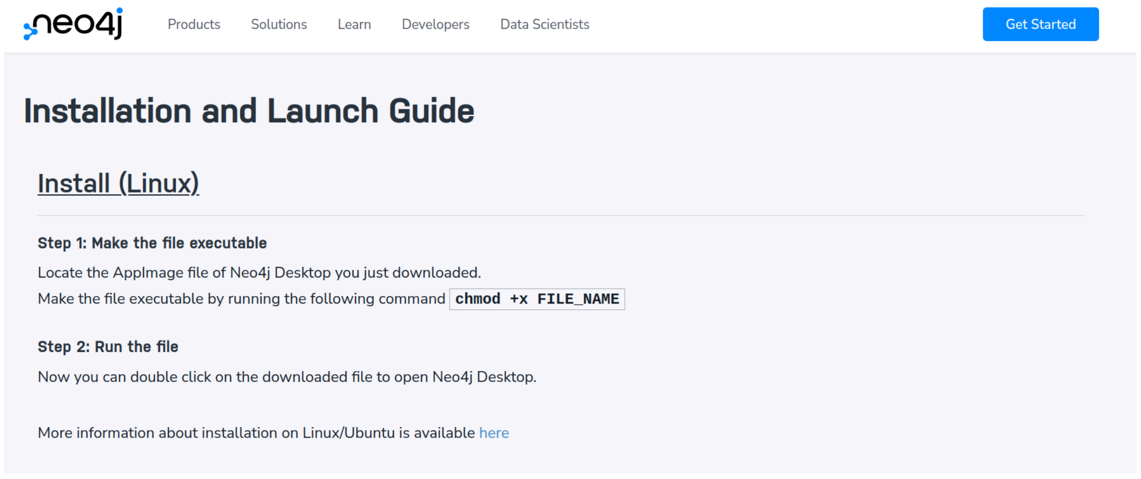
Download Your Free Copy of Neo4j

* First Name
* Last Name
* Business Email
* Company Name
* Country/Territory

By downloading you agree to the Neo4j License Agreement for Neo4j Desktop Software.

The information you provide will be used in accordance with the terms of our privacy policy.

   b. The website will **automatically** check the system information and download the corresponding version of the installation file. The website will jump to a "Thanks for Downloading Neo4j Desktop" page and, by **scrolling down** the page, you will find the **installation guide specific to your operating system**. In my case, I use Ubuntu 20.04 and the page shows a guide for Linux. You may have a different guide. Follow the instructions and finish the installation.



Installation and Launch Guide

Install (Linux)

Step 1: Make the file executable

Locate the AppImage file of Neo4j Desktop you just downloaded.
Make the file executable by running the following command `chmod +x FILE_NAME`

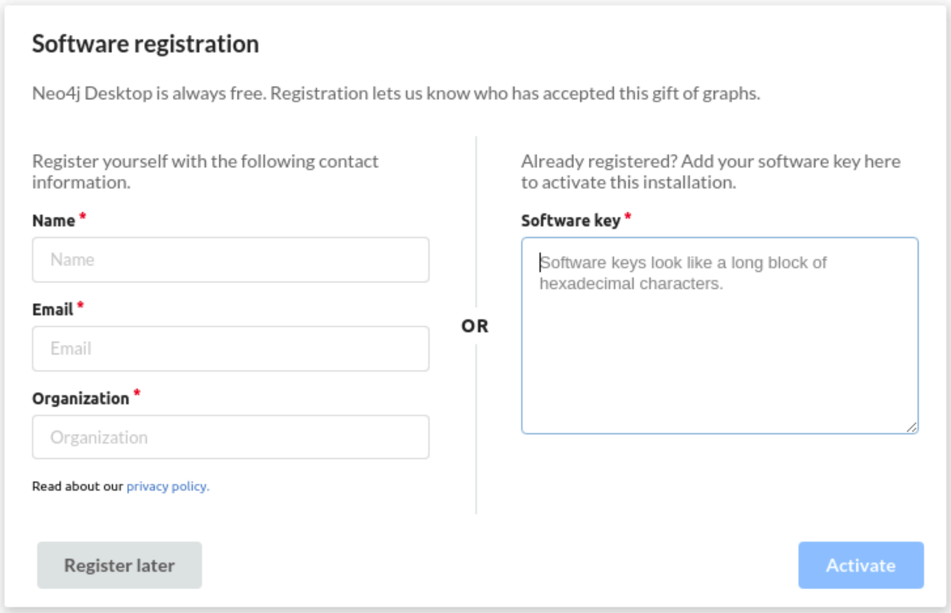
Step 2: Run the file

Now you can double click on the downloaded file to open Neo4j Desktop.

More information about installation on Linux/Ubuntu is available here

c. When launching the Neo4j Desktop, a window will show up requesting registration information. Go back to the “Thanks for Downloading Neo4j Desktop” page and copy the **Neo4j Desktop Activation Key** by clicking the button “**Copy to clipboard**”

d. Go back to the installation window and paste the activation key in the box on the right under the bolded text “**Software key**”. Click “**Activate**”.



Software registration

Neo4j Desktop is always free. Registration lets us know who has accepted this gift of graphs.

Register yourself with the following contact information.

Name *

Email *

Organization *

Read about our privacy policy.

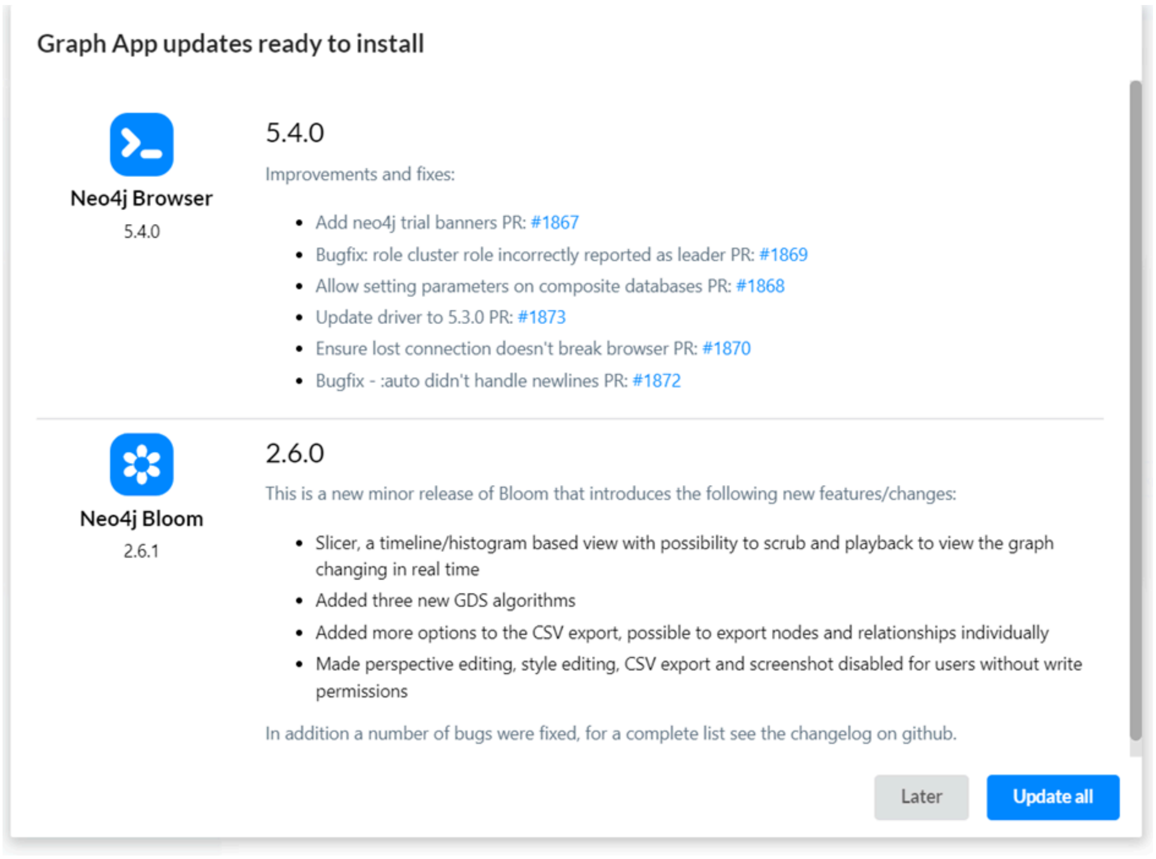
OR

Already registered? Add your software key here to activate this installation.

Software key *

Software keys look like a long block of hexadecimal characters.

e. After installation is complete, the window may ask for the update. Click Update all if this window appears. (Some may not get this window)



Graph App updates ready to install

Neo4j Browser
5.4.0

5.4.0

Improvements and fixes:

- Add neo4j trial banners PR: #1867
- Bugfix: role cluster role incorrectly reported as leader PR: #1869
- Allow setting parameters on composite databases PR: #1868
- Update driver to 5.3.0 PR: #1873
- Ensure lost connection doesn't break browser PR: #1870
- Bugfix - :auto didn't handle newlines PR: #1872

Neo4j Bloom
2.6.1

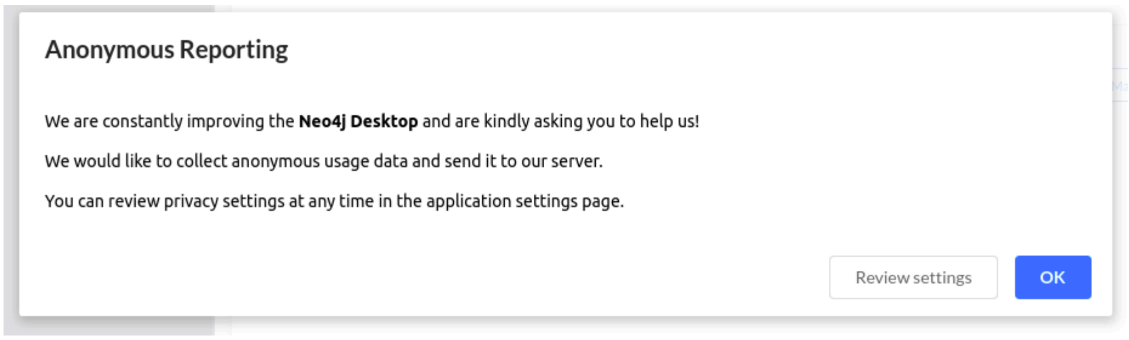
2.6.0

This is a new minor release of Bloom that introduces the following new features/changes:

- Slicer, a timeline/histogram based view with possibility to scrub and playback to view the graph changing in real time
- Added three new GDS algorithms
- Added more options to the CSV export, possible to export nodes and relationships individually
- Made perspective editing, style editing, CSV export and screenshot disabled for users without write permissions

In addition a number of bugs were fixed, for a complete list see the changelog on github.

f. The installation might take a while. When it finishes, it will come up with a notice about "**Anonymous Reporting**", click "**OK**" to continue.



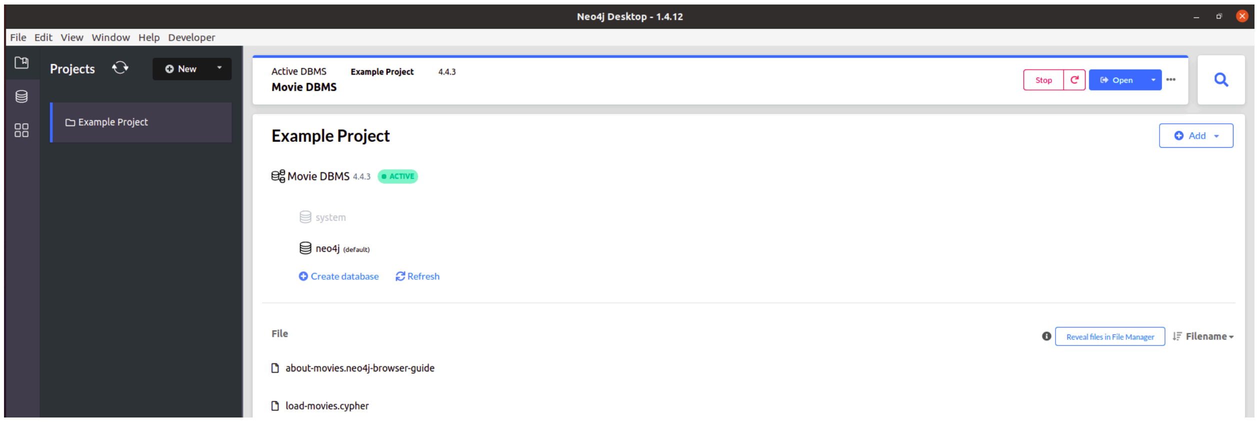
Anonymous Reporting

We are constantly improving the **Neo4j Desktop** and are kindly asking you to help us!

We would like to collect anonymous usage data and send it to our server.

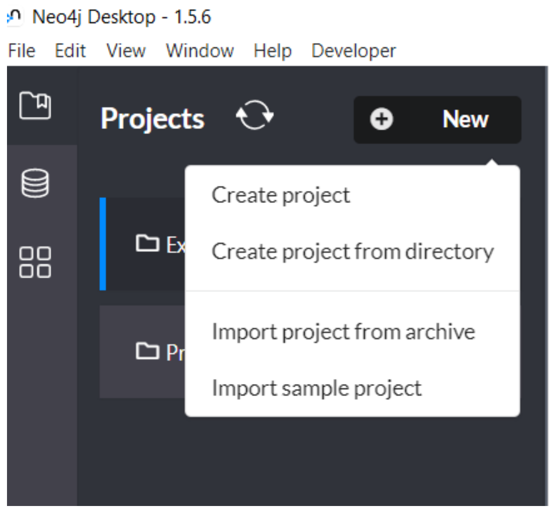
You can review privacy settings at any time in the application settings page.

g. Now Neo4j is installed on your computer!

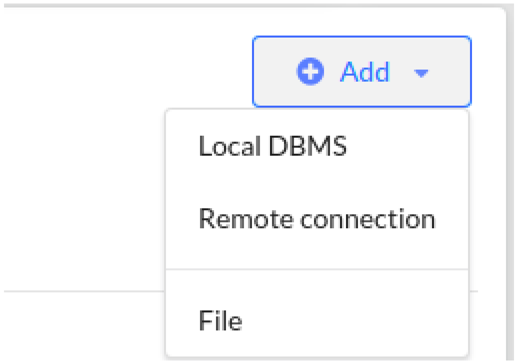


2. Create Sample Database

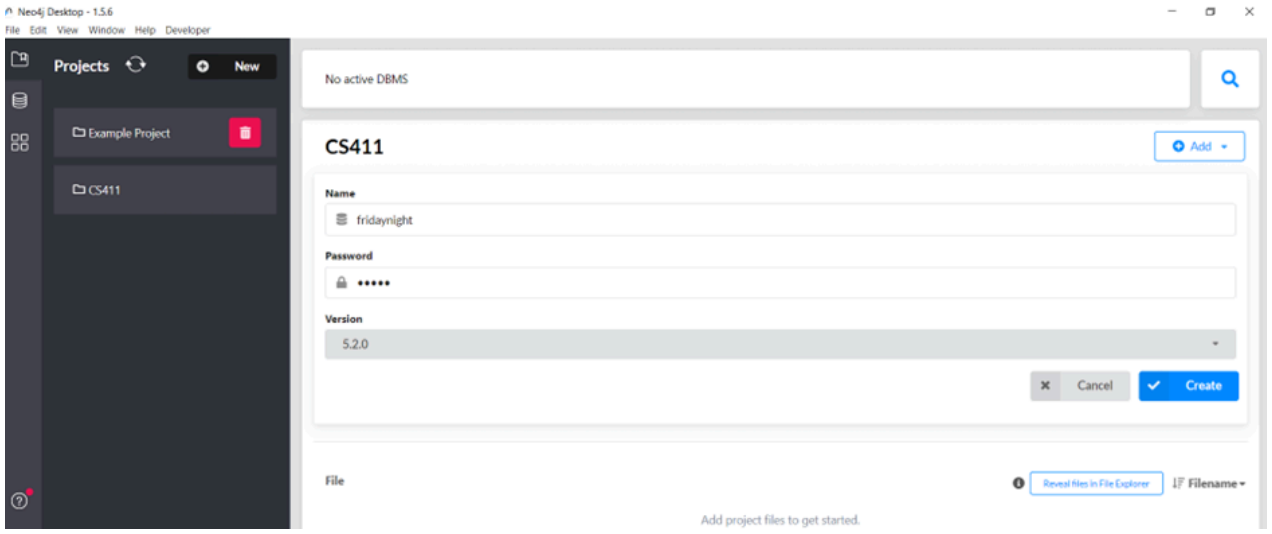
1. In the Neo4j UI, click the "**New**" button on the top left to create a new project. Let's set the project name to "**CS411**"


Neo4j Desktop - 1.5.6
File Edit View Window Help Developer
Projects
New
Create project
Create project from directory
Import project from archive
Import sample project
Ex
Pr

2. Then we create a new DBMS in the **CS411** project by clicking the "**Add**" button on the top right and then choose "**Local DBMS**".

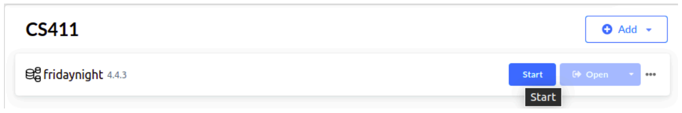


3. Let's set the DBMS's name to be "**fridaynight**" and password to be "**cs411**". You can also choose your own password. Click "**Create**".
(Please use version **5.2.0**. The latest version- *5.3.0* has some issues with authetication)

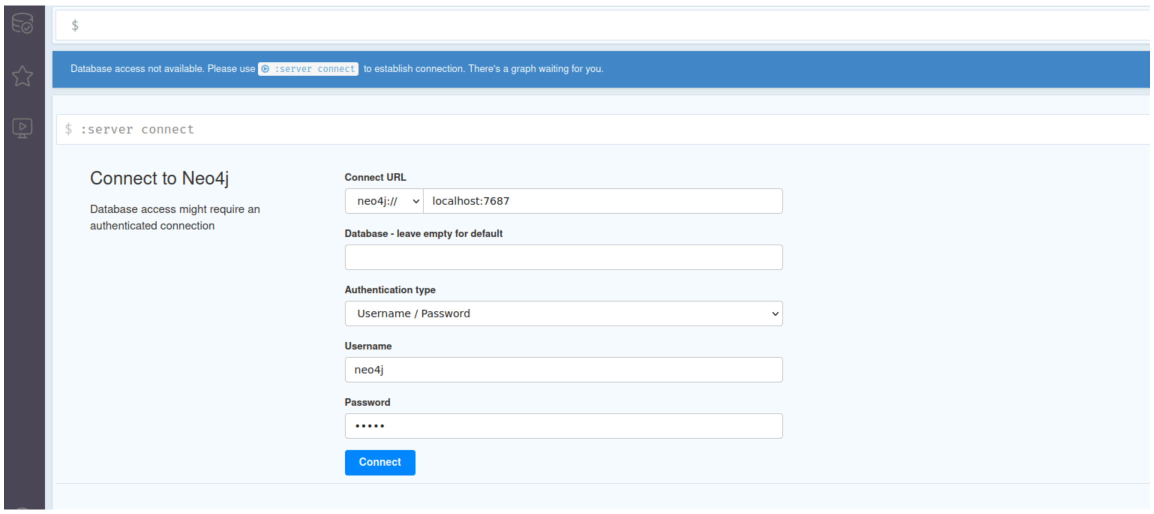

Neo4j Desktop - 1.5.6
File Edit View Window Help Developer
Projects
New
Example Project
CS411
No active DBMS
CS411
Add
Cancel
Create
File
Reveal files in File Explorer
Filename
Add project files to get started.

Name: fridaynight
Password: •••••
Version: 5.2.0

Click "**Start**" on the right to start running the DBMS.



4. Open your browser and go to "**localhost:7474**", this will lead you to the DBMS browser page. You may see a login page like the following. Leave the "**Database**" empty, enter name as "**neo4j**" and password "**cs411**". If you choose your own password in step 3, enter that password here. Then click "**Connect**".



```
$
```

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for you.

```
$ :server connect
```

Connect to Neo4j

Database access might require an authenticated connection

Connect URL: neo4j:// localhost:7687

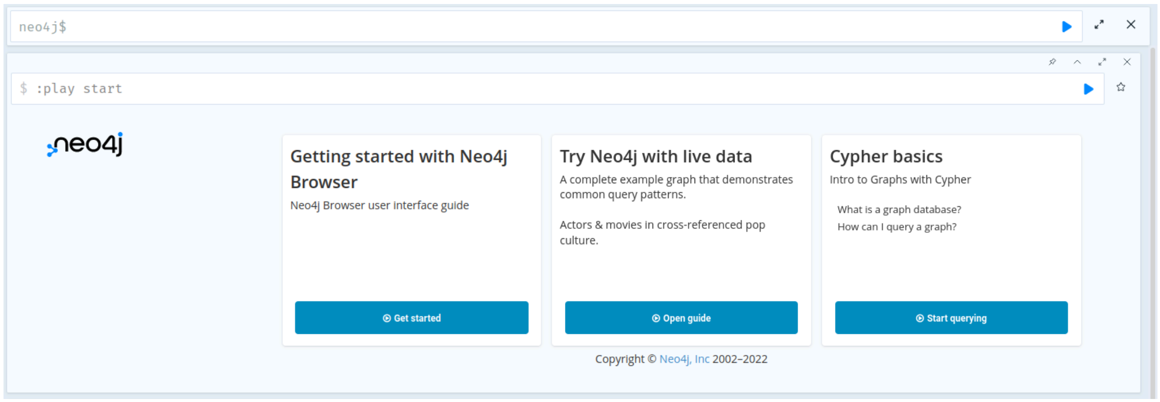
Database - leave empty for default:

Authentication type: Username / Password

Username: neo4j

Password: •••••

5. If it connects successfully, you can see an input box with the prefix "**neo4j$**" at the very top of the page, this is where you can enter and execute cypher commands

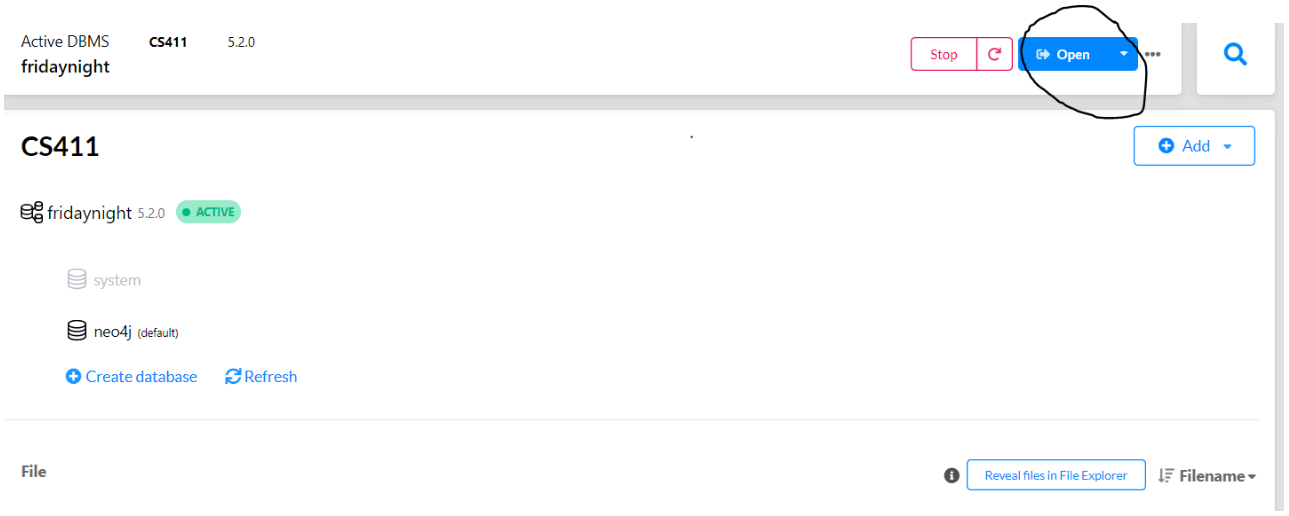


```
neo4j$
```

```
$ :play start
```

neo4j
Getting started with Neo4j Browser
Neo4j Browser user interface guide
Get started
Try Neo4j with live data
A complete example graph that demonstrates common query patterns.
Actors & movies in cross-referenced pop culture.
Open guide
Cypher basics
Intro to Graphs with Cypher
What is a graph database?
How can I query a graph?
Start querying
Copyright © Neo4j, Inc 2002–2022

6. (This is alternative of step 4) We can also open the window to execute queries by clicking **Open** option present on top right corner of project window. It will open window same window shown in step-5.



7. Run command

```
create database fridaynight
```

8. Run command

```
show databases
```

9. **(Submission 3.1)** Take a screenshot of the results. Paste it into your work document.

3. Populate Sample Database

1. switch to the database we just created

```
:use fridaynight
```

2. Paste the commands in **Appendix C** and execute them to insert data into the database.

4. Query Sample Database

1. Run this query to show you the data graph with everything in the **fridaynight** database.

```
MATCH (n) RETURN (n)
```

2. (**Submission 3.2**) Take a screenshot of the results. Paste it into your work document.
3. Run the following queries

```
MATCH (n:Bar{name:'Purple Bar'}) MATCH (n)-[r]-(m) RETURN n,r,m
```

4. **(Submission 3.3)** Take a screenshot of the results. Paste it into your work document.

Appendix A: MySQL Dataset Population

```
USE fridaynight;

CREATE TABLE Drinkers (
    name VARCHAR(40) PRIMARY KEY,
    addr text,
    hobby text,
    frequent text
    );

CREATE TABLE Bars (
    name VARCHAR(40)  PRIMARY KEY,
    addr text,
    owner text
    );

CREATE TABLE Beers (
    name VARCHAR(40)  PRIMARY KEY,
    brewer text,
    alcohol real
    );

CREATE TABLE Sells (
    bar VARCHAR(40) ,
    beer VARCHAR(40) ,
    price real,
    discount real,
    PRIMARY KEY (bar, beer)
    );

CREATE TABLE Drinks (
    drinker VARCHAR(40),
    beer VARCHAR(40),
    rating integer,
    PRIMARY KEY (drinker, beer)
    );

CREATE TABLE Favorites (
    drinker VARCHAR(40) ,
    bar VARCHAR(40) ,
    beer text,
    season text,
    PRIMARY KEY (drinker, bar)
    );

INSERT INTO Drinkers VALUES ('Alex', 'Green St', 'Reading', 'Sober Bar');
INSERT INTO Drinkers VALUES ('Betty', 'King St', 'Singing', 'Green Bar');
INSERT INTO Drinkers VALUES ('Cindy', 'Green St', 'Hiking', 'Green Bar');

INSERT INTO Bars VALUES ('Sober Bar', 'Purple St', 'Jim');
INSERT INTO Bars VALUES ('Green Bar', 'Green St','Sally');
INSERT INTO Bars VALUES ('Purple Bar', 'Purple St','Paul');

INSERT INTO Beers VALUES ('Sam Adams', 'Boston Beer', 0.049);
INSERT INTO Beers VALUES ('Bud', 'AB InBev', 0.05);
INSERT INTO Beers VALUES ('Bud Lite', 'AB InBev', 0.042);
INSERT INTO Beers VALUES ('Coors', 'Coors', 0.05);

INSERT INTO Sells VALUES ('Sober Bar', 'Bud', 5.0, 0.05);
INSERT INTO Sells VALUES ('Sober Bar', 'Bud Lite', 3.0, 0);
INSERT INTO Sells VALUES ('Sober Bar', 'Sam Adams', 6.0, 0.1);
INSERT INTO Sells VALUES ('Green Bar', 'Bud', 2.0, 0);
INSERT INTO Sells VALUES ('Green Bar', 'Sam Adams', 4.5, 0.2);
INSERT INTO Sells VALUES ('Green Bar', 'Coors', null, null);
INSERT INTO Sells VALUES ('Purple Bar', 'Bud', 4.5, 0.2);
INSERT INTO Sells VALUES ('Purple Bar', 'Sam Adams', 5.0, 0.3);

INSERT INTO Drinks VALUES ('Alex', 'Bud', 3);
INSERT INTO Drinks VALUES ('Alex', 'Sam Adams', 5);
INSERT INTO Drinks VALUES ('Betty', 'Sam Adams', 2);
INSERT INTO Drinks VALUES ('Cindy', 'Bud Lite', 3);

INSERT INTO Favorites VALUES ('Alex', 'Sober Bar', 'Bud', 'Spring');
INSERT INTO Favorites VALUES ('Alex', 'Green Bar', 'Sam Adams', 'Summer');
INSERT INTO Favorites VALUES ('Cindy', 'Green Bar', 'Sam Adams', 'Summer');
```

Appendix B: MongoDB Dataset Population

```
/* Drinkers */
db.Drinkers.insertOne({name:'Alex', addr:'Green St', hobby:'Reading', frequents:'Sober Bar'})
db.Drinkers.insertOne({name:'Betty', addr:'King St', hobby:'Singing', frequents:'Green Bar'})
db.Drinkers.insertOne({name:'Cindy', addr:'Green St', hobby:'Hiking', frequents:'Green Bar'})


/* Bars */
db.Bars.insertOne({name:'Sober Bar', addr:'Purple St', owner:'Jim'})
db.Bars.insertOne({name:'Green Bar', addr:'Green St',owner:'Sally'})
db.Bars.insertOne({name:'Purple Bar', addr:'Purple St', owner:'Paul'})

/* Beers */
db.Beers.insertOne({name:'Sam Adams', brewer:'Boston Beer', alcohol:0.049})
db.Beers.insertOne({name:'Bud', brewer:'AB InBev', alcohol:0.05})
db.Beers.insertOne({name:'Bud Lite', brewer:'AB InBev', alcohol:0.042})
db.Beers.insertOne({name:'Coors', brewer:'Coors', alcohol:0.05})

/* Sells */
db.Sells.insertOne({bar:'Sober Bar', beer:'Bud', price:5.0, discount:0.05})
db.Sells.insertOne({bar:'Sober Bar', beer:'Bud Lite', price:3.0, discount:0})
db.Sells.insertOne({bar:'Sober Bar', beer:'Sam Adams', price:6.0, discount:0.1})
db.Sells.insertOne({bar:'Green Bar', beer:'Sam Adams', price:4.5, discount:0.2})
db.Sells.insertOne({bar:'Green Bar', beer:'Bud', price:2.0, discount:0})
db.Sells.insertOne({bar:'Green Bar', beer:'Coors', price:null, discount:null})
db.Sells.insertOne({bar:'Purple Bar', beer:'Bud', price:'4.5', discount:0.2})
db.Sells.insertOne({bar:'Purple Bar', beer:'Sam Adams', price:5.0, discount:0.3})

/* Drinks */
db.Drinks.insertOne({drinker:'Alex', beer:'Bud', rating:3})
db.Drinks.insertOne({drinker:'Alex', beer:'Sam Adams', rating:5})
db.Drinks.insertOne({drinker:'Betty', beer:'Sam Adams', rating:2})
db.Drinks.insertOne({drinker:'Cindy', beer:'Bud Lite', rating:3})

/* Favorites */
db.Favorites.insertOne({drinker:'Alex', bar:'Sober Bar', beer:'Bud', season:'Spring'})
db.Favorites.insertOne({drinker:'Alex', bar:'Green Bar', beer:'Sam Adams', season:'Summer'})
db.Favorites.insertOne({drinker:'Cindy', bar:'Green Bar', beer:'Sam Adams', season:'Summer'})

/* CompleteDrinkers: "Complete" version of Drinkers */
db.CompleteDrinkers.insertOne({name:'Alex', addr:'Green St', hobby:'Reading',
        frequents:{name:'Sober Bar', addr:'Purple St', owner:'Jim'},
        drinks:[{beer:{name:'Bud', brewer:'AB InBev', alcohol:5.0}, rating:3}, {beer: {name:'Sam Adams',
brewer:'Boston Beer', alcohol:4.9}, rating:5}]
        })

db.CompleteDrinkers.insertOne({name:'Betty', addr:'King St', hobby:'Singing',
        frequents:{name:'Green Bar', addr:'Green St', owner:'Sally'},
        drinks:[{beer: {name:'Sam Adams', brewer:'Boston Beer', alcohol:4.9}, rating:2}]
        })
```

```
db.CompleteDrinkers.insertOne({name:'Cindy', addr:'Green St', hobby:'Hiking',
        frequents:{name:'Green Bar', addr:'Green St', owner:'Sally'},
        drinks:[{beer: {name:'Bud Lite', brewer:'AB InBev', alcohol:4.2}, rating:3}]
        })

/* CompleteBars: "Complete" version of Bars */
db.CompleteBars.insertOne({name:'Sober Bar', addr:'Purple St', owner:'Jim',
        sells: [{beer:{name:'Bud', brewer:'AB InBev', alcohol:5.0}, price:5.0, discount:0.05},
{beer:{name:'Bud Lite', brewer:'AB InBev', alcohol:4.2}, price:3.0, discount:0}, {beer:{name:'Sam Adams',
brewer:'Boston Beer', alcohol:4.9}, price:6.0, discount:0.1}]
        })

db.CompleteBars.insertOne({name:'Green Bar', addr:'Green St',owner:'Sally',
        sells: [{beer:{name:'Sam Adams', brewer:'Boston Beer', alcohol:4.9}, price:4.5, discount:0.2},
{beer:{name:'Bud', brewer:'AB InBev', alcohol:5.0}, price:2.0, discount:0}, {beer:{name:'Coors',
brewer:'Coors', alcohol:5.0}, price:null, discount:null}]
        })

db.CompleteBars.insertOne({name:'Purple Bar', addr:'Purple St', owner:'Paul',
        sells: [{beer:{name:'Bud', brewer:'AB InBev', alcohol:5.0}, price:'4.5', discount:0.2},
{beer:{name:'Sam Adams', brewer:'Boston Beer', alcohol:4.9}, price:5.0, discount:0.3}]
        })

/* CompleteRefDrinkers: "Complete" "Reference" version of Drinkers */
db.CompleteRefDrinkers.insertOne({name:'Alex', addr:'Green St', hobby:'Reading',
        frequents:ObjectId("59e9a24eeabd8c0d2c5dd16c"),
        drinks:[{beer:ObjectId("59e62a3a93b695daed6adf33"), rating:3},
{beer:ObjectId("59e3a4889108219c38e8a95b"), rating:5}]
        })

db.CompleteRefDrinkers.insertOne({name:'Betty', addr:'King St', hobby:'Singing',
        frequents:ObjectId("59e9a24eeabd8c0d2c5dd16d"),
        drinks:[{beer: ObjectId("59e3a4889108219c38e8a95b"), rating:2}]
        })

db.CompleteRefDrinkers.insertOne({name:'Cindy', addr:'Green St', hobby:'Hiking',
        frequents:ObjectId("59e9a24eeabd8c0d2c5dd16d"),
        drinks:[{beer: ObjectId("59e3a4889108219c38e8a95d"), rating:3}]
        })
```

Appendix C: Neo4j Dataset Population

```
/* Node Drinker --------- */
CREATE(:Drinker {name:'Alex', addr:'Green St', hobby:'Reading', frequents:'Sober Bar'});
CREATE(:Drinker {name:'Betty', addr:'King St', hobby:'Singing', frequents:'Green Bar'});
CREATE(:Drinker {name:'Cindy', addr:'Green St', hobby:'Hiking', frequents:'Green Bar'});

/* Node Bar --------- */
CREATE(:Bar {name:'Sober Bar', addr:'Purple St', owner:'Jim'});
CREATE(:Bar {name:'Green Bar', addr:'Green St',owner:'Sally'});
CREATE(:Bar {name:'Purple Bar', addr:'Purple St', owner:'Paul'});

/* Node Beer --------- */
CREATE (:Beer {name:'Sam Adams', brewer:'Boston Beer', alcohol:0.049});
CREATE (:Beer {name:'Bud', brewer:'AB InBev', alcohol:0.05});
CREATE (:Beer {name:'Bud Lite', brewer:'AB InBev', alcohol:0.042});
CREATE (:Beer {name:'Coors', brewer:'Coors', alcohol:0.05});

/* Relationship Sells --------- */

MATCH (bar:Bar {name:'Sober Bar'}), (beer:Beer {name:'Bud'})
CREATE (bar)-[:Sells {price:5.0, discount:0.05}]->(beer);

MATCH (bar:Bar {name:'Sober Bar'}), (beer:Beer {name:'Bud Lite'})
CREATE (bar)-[:Sells {price:3.0, discount:0}]->(beer);

MATCH (bar:Bar {name:'Sober Bar'}), (beer:Beer {name:'Sam Adams'})
CREATE (bar)-[:Sells {price:6.0, discount:0.1}]->(beer);

MATCH (bar:Bar {name:'Green Bar'}), (beer:Beer {name:'Bud'})
CREATE (bar)-[:Sells {price:2.0, discount:0}]->(beer);

MATCH (bar:Bar {name:'Green Bar'}), (beer:Beer {name:'Sam Adams'})
CREATE (bar)-[:Sells {price:4.5, discount:0.2}]->(beer);

MATCH (bar:Bar {name:'Green Bar'}), (beer:Beer {name:'Coors'})
CREATE (bar)-[:Sells {price:null, discount:null}]->(beer);

MATCH (bar:Bar {name:'Purple Bar'}), (beer:Beer {name:'Bud'})
CREATE (bar)-[:Sells {price:4.5, discount:0.2}]->(beer);

MATCH (bar:Bar {name:'Purple Bar'}), (beer:Beer {name:'Sam Adams'})
CREATE (bar)-[:Sells {price:5.0, discount:0.3}]->(beer);

/* Relationship Drinks --------- */

MATCH (drinker:Drinker {name:'Alex'}), (beer:Beer {name:'Bud'})
CREATE (drinker)-[:Drinks {rating:3}]->(beer);

MATCH (drinker:Drinker {name:'Alex'}), (beer:Beer {name:'Sam Adams'})
CREATE (drinker)-[:Drinks {rating:5}]->(beer);
```

```
MATCH (drinker:Drinker {name:'Betty'}), (beer:Beer {name:'Sam Adams'})
CREATE (drinker)-[:Drinks {rating:2}]->(beer);

MATCH (drinker:Drinker {name:'Cindy'}), (beer:Beer {name:'Bud Lite'})
CREATE (drinker)-[:Drinks {rating:3}]->(beer);

/* Favorites as a 3-way Relationship. */
/* Create Favorites as node,
/* and connect Drinker, Bar, Beer to it, for a 3-way relationship. */

CREATE (:Favorites {drinker:'Alex', bar:'Sober Bar', beer:'Bud', season:'Spring'});
CREATE (:Favorites {drinker:'Alex', bar:'Green Bar', beer:'Sam Adams', season:'Summer'});
CREATE (:Favorites {drinker:'Cindy', bar:'Green Bar', beer:'Sam Adams', season:'Summer'});

MATCH (f:Favorites), (d:Drinker {name:f.drinker})
CREATE (d)-[:FavoritesAsDrinker]->(f);

MATCH (f:Favorites), (b:Bar {name:f.bar})
CREATE (b)-[:FavoritesAsBar]->(f);

MATCH (f:Favorites), (b:Beer {name:f.beer})
CREATE (b)-[:FavoritesAsBeer]->(f);
```