

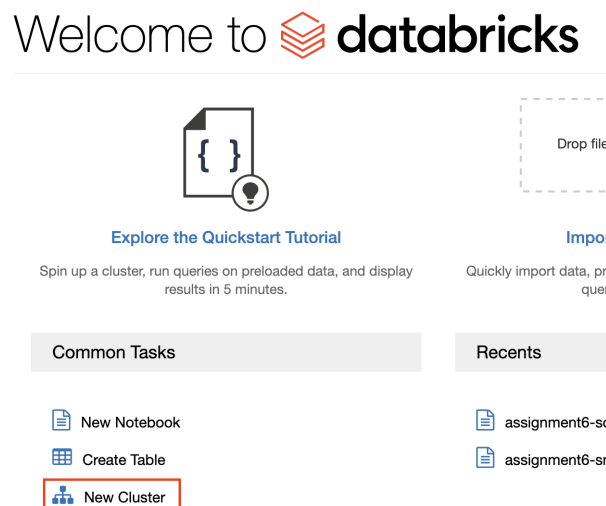
Homework Assignment #6 Setup

(Spark)

1. Create a Databricks account by following this link and be very careful to choose the **community edition** <https://databricks.com/try-databricks>.
2. Fill out the form. For the Company name you can type UCI, and for work email please provide your UCI email. You will receive an email back to confirm your email address and create a password.

Note: Clusters in Databricks self-terminate every 2 hours. You can either manually recreate the cluster using the steps 3-4 below, or you can let Databricks create a new cluster for you when running your notebook (**recommended**). When running a cell on a detached notebook, it will ask you to automatically attach and launch a cluster.

3. Click on the create cluster link when you wish to create a new cluster.



4. Pick any name for your cluster and make sure that you have the same settings as in the picture below and then create the cluster.

New Cluster Cancel Create Cluster **0 Workers:** 0.0 GB Memory, 0 Cores, 0 DBU
1 Driver: 15.3 GB Memory, 2 Cores, 1 DBU ?

Cluster Name

cs122d

Databricks Runtime Version ?

Runtime: 8.2 (Scala 2.12, Spark 3.1.1) | v

Note Databricks Runtime 8.x uses Delta Lake as the default table format. [Learn more](#)

Instance

Free 15GB Memory: As a Community Edition user, your cluster will automatically terminate after an idle period of two hours. For more configuration options, please [upgrade your Databricks subscription](#).


Instances Spark

Availability Zone ?

us-west-2c | v

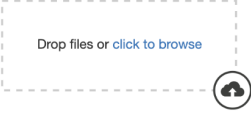
- Download the json files from this Google Drive folder
<https://drive.google.com/drive/folders/1dOGaWH5wPfcTP72fw8mdjMMIW7zvSNXK?usp=sharing>
- Go to the main databricks page. Click the link for import & explore data.

Welcome to  **databricks**




Explore the Quickstart Tutorial

Spin up a cluster, run queries on preloaded data, and display results in 5 minutes.



Import & Explore Data

Quickly import data, preview its schema, create a table, and query it in a notebook.



Create a Blank Notebook

Create a notebook to start querying, visualizing, and modeling your data.

- Click browse in order to select and import all of the json files.

Create New Table

Data source ?

Upload File S3 DBFS Other Data Sources

Upload to DBFS ?

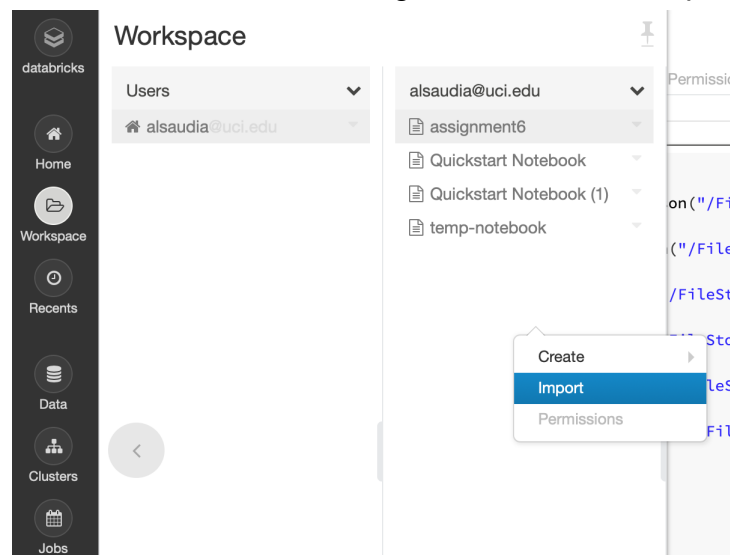
/FileStore/tables/ (optional) Select

File ?

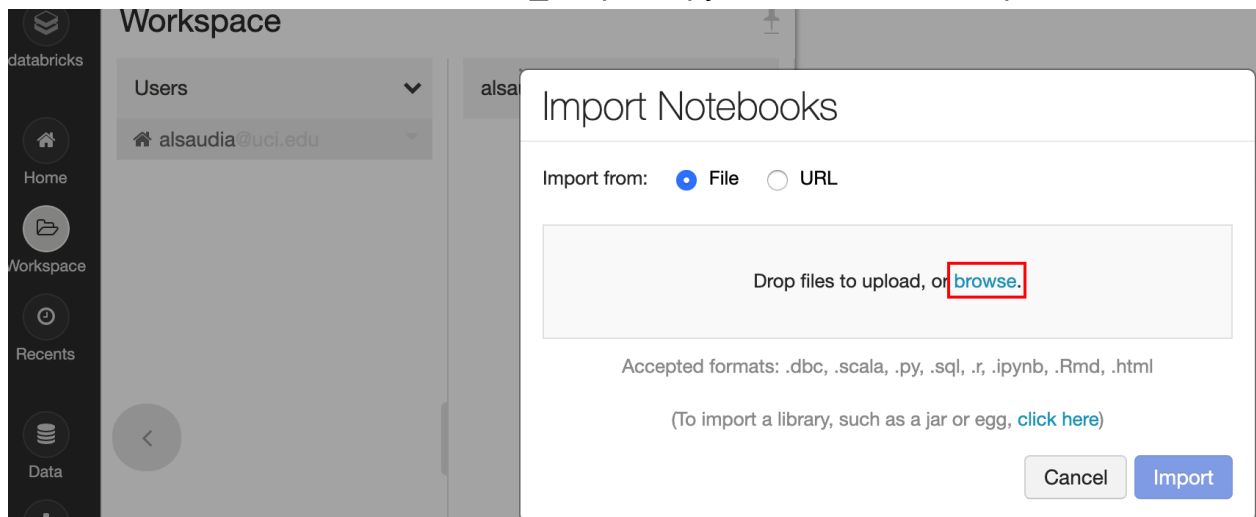
Drop files to upload, or browse.

The files should now be stored in the cloud under the “/FileStore/tables” directory! You have also been provided with a notebook (ipynb) template file that contains some initial environment setup code that will load the files into dataframes that you can run queries on. Let’s load that into Databricks as well by doing the following:

8. Download the hw6_template.ipynb file from the class website to your computer.
9. Click on workspace -> Users -> <user> -> right click and then import.



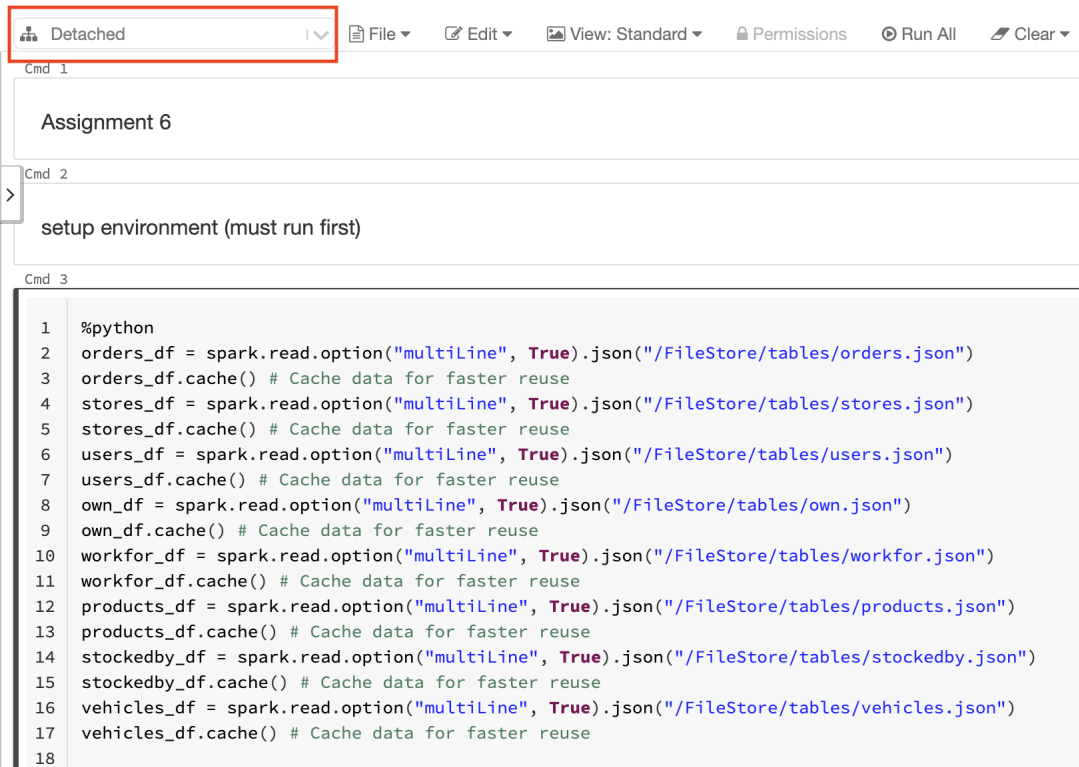
10. Click browse and choose the file hw6_template.ipynb and then click import.



11. After importing and opening the file, run the setup environment cell.

NOTE before running the cell: if the dropdown below says “Detached”, make sure to attach the notebook to a cluster! (You can’t compute without nodes to compute on...)

assignment6-smith_john-1234 (Python)



The screenshot shows a JupyterLab interface with a detached terminal window. The terminal window has a title bar with a red border and contains the text "Detached". Below the title bar, there are three tabs labeled "Cmd 1", "Cmd 2", and "Cmd 3". The "Cmd 1" tab is active and shows the text "Assignment 6". The "Cmd 2" tab is also active and shows the text "setup environment (must run first)". The "Cmd 3" tab is active and shows a Python script for loading data from JSON files into Spark DataFrames. The script includes comments for caching data for faster reuse.

```
1 %python
2 orders_df = spark.read.option("multiLine", True).json("/FileStore/tables/orders.json")
3 orders_df.cache() # Cache data for faster reuse
4 stores_df = spark.read.option("multiLine", True).json("/FileStore/tables/stores.json")
5 stores_df.cache() # Cache data for faster reuse
6 users_df = spark.read.option("multiLine", True).json("/FileStore/tables/users.json")
7 users_df.cache() # Cache data for faster reuse
8 own_df = spark.read.option("multiLine", True).json("/FileStore/tables/own.json")
9 own_df.cache() # Cache data for faster reuse
10 workfor_df = spark.read.option("multiLine", True).json("/FileStore/tables/workfor.json")
11 workfor_df.cache() # Cache data for faster reuse
12 products_df = spark.read.option("multiLine", True).json("/FileStore/tables/products.json")
13 products_df.cache() # Cache data for faster reuse
14 stockedby_df = spark.read.option("multiLine", True).json("/FileStore/tables/stockedby.json")
15 stockedby_df.cache() # Cache data for faster reuse
16 vehicles_df = spark.read.option("multiLine", True).json("/FileStore/tables/vehicles.json")
17 vehicles_df.cache() # Cache data for faster reuse
18
```

12. You are now ready to proceed with the assignment itself!