Cloudera Data Science Workbench

Using the Workbench

Date published: 2020-02-28
Date modified:

https://docs.cloudera.com/

Legal Notice

Contents

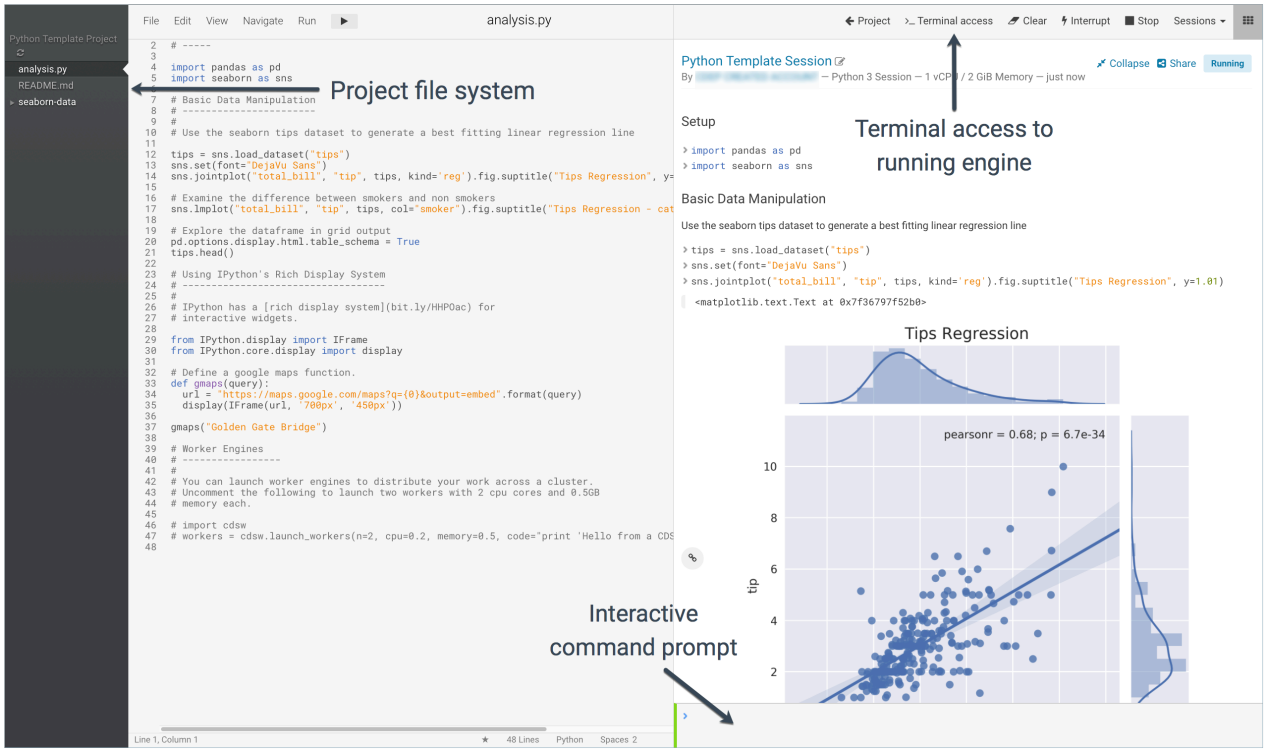
Using the Workbench

The workbench console provides an interactive environment tailored for data science, supporting R, Python and Scala.

The Workbench currently supports R, Python, and Scala engines. You can use these engines in isolation, as you would on your laptop, or connect to your CDH cluster using Cloudera Distribution of Apache Spark 2 and other libraries.

The workbench UI includes four primary components:

- An editor where you can edit your scripts.
- A console where you can track the results of your analysis.
- A command prompt where you can enter commands interactively.
- A terminal where you can use a Bash shell.



Typically, you would use the following steps to run a project in the workbench:

Start a New Session

The first step to run a project in the workbench is to start a new session.

About this task

Note: Shell start up files are not run during session start up. CDSW sessions are not bash shells, so shell start up files such as bashrc, zsh, and ksh are not run. If you want to set an environment variable, you can use the CDSW environment variables feature. This will ensure the environment variable is injected in all contexts: sessions, terminals, experiments, models, jobs, etc. If you want to run more complicated code during startup (for example, conditional statements), consider using PYTHONSTARTUP, see Startup.py or Rprofile, see Managing R with .Rprofile, .Renviron, Rprofile.site, Renviron.site, rsession.conf, and repos.conf.

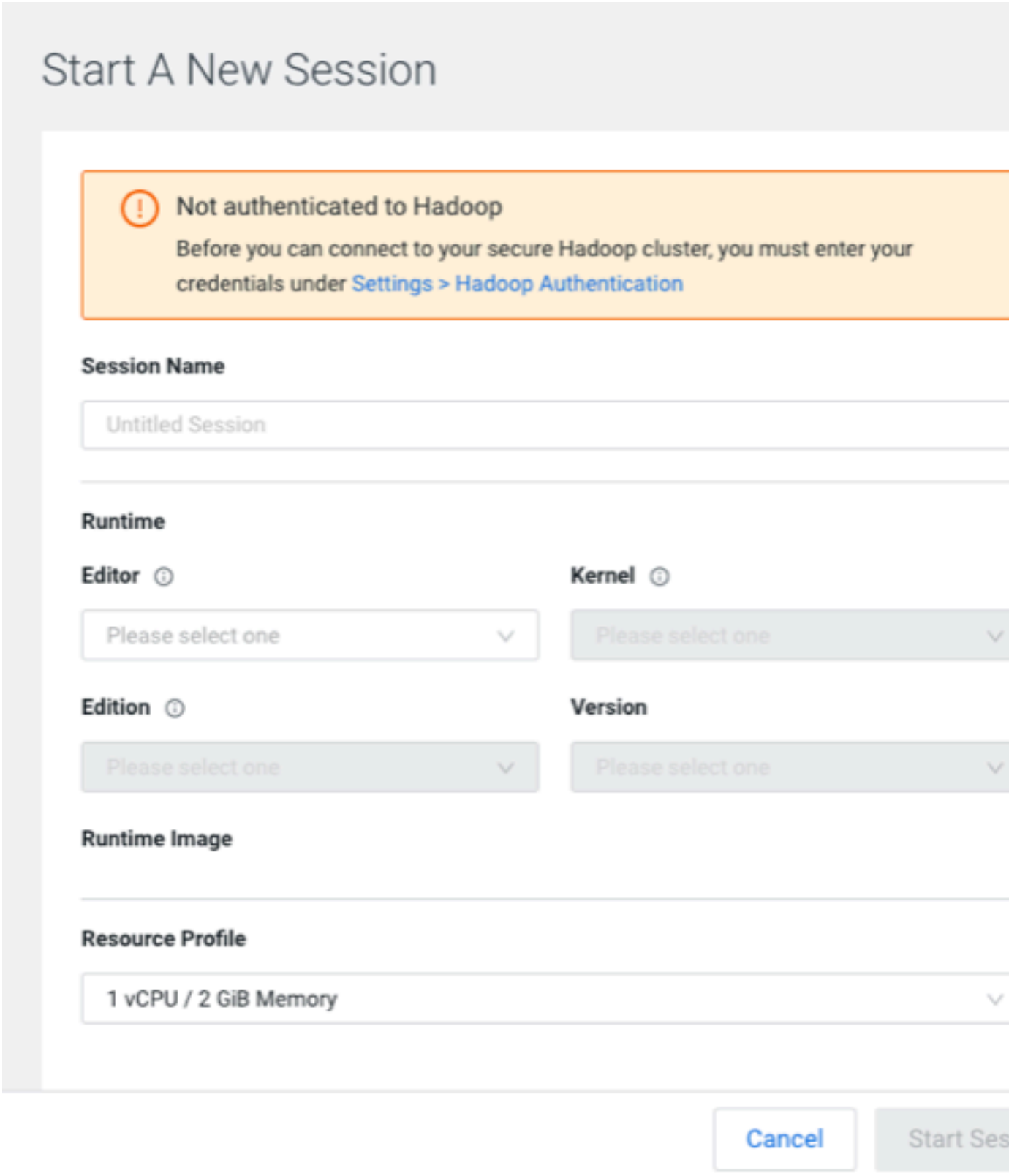
Procedure

1. Navigate to your project's Overview page.
2. Click New Session.

   The information presented on this page will depend on which default engine you have chosen for your project: Runtime or Legacy Engines. You can change the default engine later in this task.

   The **Start A New Session** dialog box opens.

3. Check the settings for your session:
   If your project is using ML Runtimes, you will see the following settings:



Editor

Selects the Editor; currently only Workbench is supported and therefore the selector is static.

Kernel

Selects the Kernel, for example Python 3.7, R4.0.

Edition

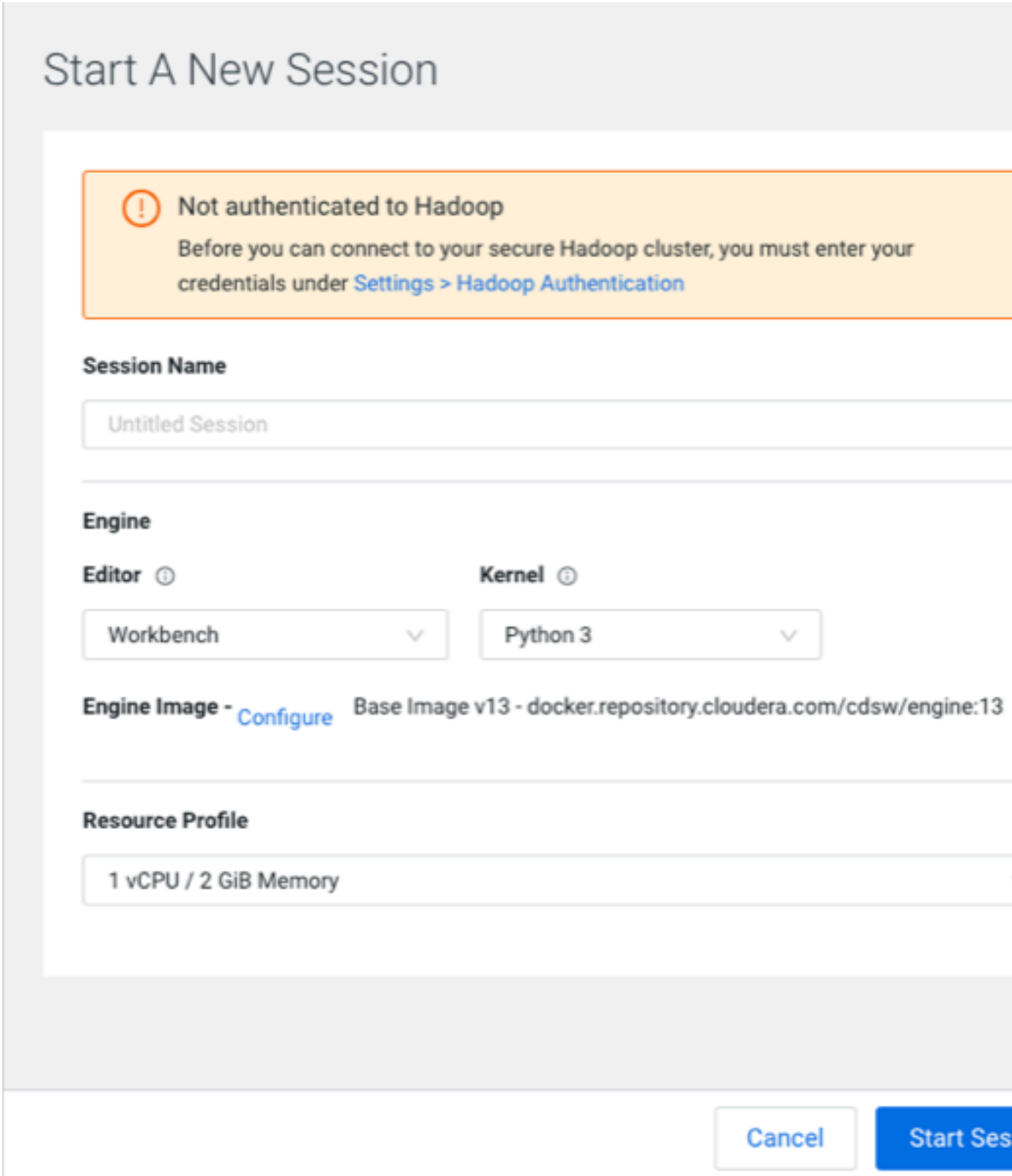
Selects the Runtime Edition. Initially only Standard variants are supported.

Version

Selects the ML Runtimes version.

Note: The selector options only consider the configurations supported by the actual deployments and certain selections will automatically limit others. For example, certain versions are only relevant for Python or certain editors are supported only with certain kernels.

If your project is using Legacy Engines, you will see the following settings:



Editor

Selects the Editor; currently only Workbench is supported and therefore the selector is static.

Kernel

Selects the Kernel. Initially only Python Runtimes are supported.

Engine Image

Selects the engine image. Click Configure to display the Project Setting > Advanced window to modify your environment variables and shared memory limit.

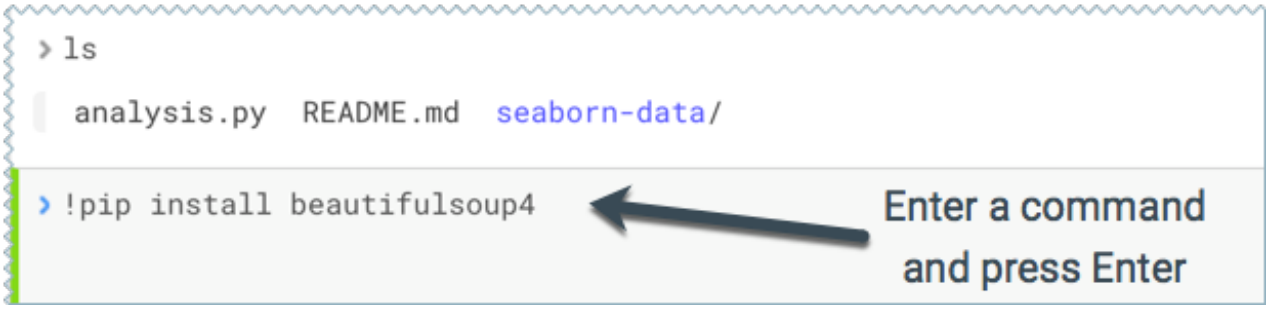
4. If your project is using Legacy Engines, you can modify the engine image used by this session:
   a) By Engine Image, click Configure.
      Cloudera Machine Learning displays the Project Settings page.
   b) Select the Runtime/Engine tab.
   c) Next to Default Engine, select ML Runtime or Legacy Engine.
   d) Click Save Engine.
5. Specify your Resource Profile.
   This attribute will define how many vCPUs and how much memory will be reserved to run the workload (for example, session including the runtime itself). The minimum configuration is 1vCPU and 2 GB memory.
6. Click Start Session.
   The command prompt at the bottom right of your browser window will turn green when the engine is ready. Sessions typically take between 10 and 20 seconds to start.

Run Code

You can enter and run code at the command prompt or the editor. The editor is best for code you want to keep, while the command prompt is best for quick interactive exploration.

About this task

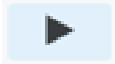
Command Prompt - The command prompt functions largely like any other. Enter a command and press Enter to run it. If you want to enter more than one line of code, use Shift+Enter to move to the next line. The output of your code, including plots, appears in the console.



If you created your project from a template, you should see project files in the editor. You can open a file in the editor by clicking the file name in the file navigation bar on the left.

Editor - To run code from the editor:

Procedure

1. Select a script from the project files on the left sidebar.
2. To run the whole script click  on the top navigation bar, or, highlight the code you want to run and press Ctrl+Enter (Windows/Linux) or cmd+Enter (macOS).
   When doing real analysis, writing and executing your code from the editor rather than the command prompt makes it easy to iteratively develop your code and save it along the way.
   If you require more space for your editor, you can collapse the file list by double-clicking between the file list pane and the editor pane. You can hide the editor using editor's View menu.

Code Autocomplete

You can enter and run code at the command prompt or the editor. The editor is best for code you want to keep, while the command prompt is best for quick interactive exploration.

The Python and R kernels include support for automatic code completion, both in the editor and the command prompt. Use single tab to display suggestions and double tab for autocomplete.

Project Code Files

All project files are stored to persistent storage within the respective project directory at /var/lib/cdsw/current/projects.

Project files can be accessed within the project just as you would in a typical directory structure. For example, you can import functions from one file to another within the same project.

Access the Terminal

Cloudera Data Science Workbench provides full terminal access to running engines from the web console. You can use the terminal to move files around, run Git commands, access the YARN and Hadoop CLIs, or install libraries that cannot be installed directly from the engine.

About this task

To access the Terminal from a running session:

Procedure

click Terminal Access above the session log pane.

The terminal's default working directory is /home/cdsw, which is where all your project files are stored. Any modifications you make to this folder will persist across runs, while modifications to other folders are discarded.

If you are using Kerberos authentication, you can run klist to see your Kerberos principal. If you run hdfs dfs -ls you will see the files stored in your HDFS home directory.

Note: The terminal does not provide root or sudo access to the container. To install packages that require root access, see Customizing Engine Images .

Stop a Session

When you are done with a session, you can stop it.

Procedure

1. Click Stop in the menu bar above the console.

2. Alternatively you can stop a session by typing the following command:

   R

   ```
   quit()
   ```

   Python

   ```
   exit
   ```

   Scala

   ```
   quit()
   ```

   Sessions automatically stop after an hour of inactivity.

Workbench editor file types

The default workbench editor supports the following file types:

- Text
- CSS
- HTML
- JavaScript
- JSON
- PHP
- Scala
- C++
- C#
- CLike
- Java
- CoffeeScript
- R
- Julia
- Ruby
- Clojure
- Perl
- Python
- SASS
- Lua
- SQL
- Diff
- Markdown
- YAML
- Haxe

Jupyter Magic Commands

Cloudera Data Science Workbench's Scala and Python kernels are based on Jupyter kernels. Jupyter kernels support varying magic commands that extend the core language with useful shortcuts. This section details the magic commands (magics) supported by Cloudera Data Science Workbench.

Note: Jupyter magic commands apply only to legacy engine projects.

Line magics begin with a single %: for example, %timeit. Cell magics begin with a double %%: for example, %%bash.

Python

The examples below show how to retrieve the password from an environment variable and use it to connect.

You can access data using pyodbc or SQLAlchemy.

```
# pyodbc lets you make direct SQL queries.
!wget https://pyodbc.googlecode.com/files/pyodbc-3.0.7.zip
!unzip pyodbc-3.0.7.zip
!cd pyodbc-3.0.7;python setup.py install --prefix /home/cdsw
import os

# See http://www.connectionstrings.com/ for information on how to construct
 ODBC connection strings.
db = pyodbc.connect("DRIVER={PostgreSQL Unicode};SERVER=localhost;PORT=54
32;DATABASE=test_db;USER=cdswuser;OPTION=3;PASSWORD=%s" % os.environ["POSTGR
ESQL_PASSWORD"])
cursor = cnxn.cursor()
cursor.execute("select user_id, user_name from users")

# sqlalchemy is an object relational database client that lets you make data
base queries in a more Pythonic way.
!pip install sqlalchemy
import os
import sqlalchemy
from sqlalchemy.orm import sessionmaker
from sqlalchemy import create_engine
db = create_engine("postgresql://cdswuser:%s@localhost:5432/test_db" % os.
environ["POSTGRESQL_PASSWORD"])
session = sessionmaker(bind=db)
user = session.query(User).filter_by(name='ed').first()
# python-oracledb can be used to connect directly to Oracle databases witho
ut need to install oracle drivers
# See https://python-oracledb.readthedocs.io/en/latest/user_guide/installa
tion.html#quickstart
!pip install oracledb
import oracledb
import os

un = os.environ.get('PYTHON_USERNAME')
pw = os.environ.get('PYTHON_PASSWORD')
cs = os.environ.get('PYTHON_CONNECTSTRING')
with oracledb.connect(user=un, password=pw, dsn=cs) as connection:
    with connection.cursor() as cursor:
        sql = """select sysdate from dual"""
        for r in cursor.execute(sql):
            print(r)
```

Scala

Cloudera Data Science Workbench's Scala kernel is based on Apache Toree.

It supports the line magics documented in the Apache Toree magic tutorial.