Software for the Course

Joel Klein

Setting up Python

Python Distributions

- Using a distribution simplifies the process of setting up your python environment, includes necessary data packages, and integrate useful tools (IDE’s, notebooks, etc.)

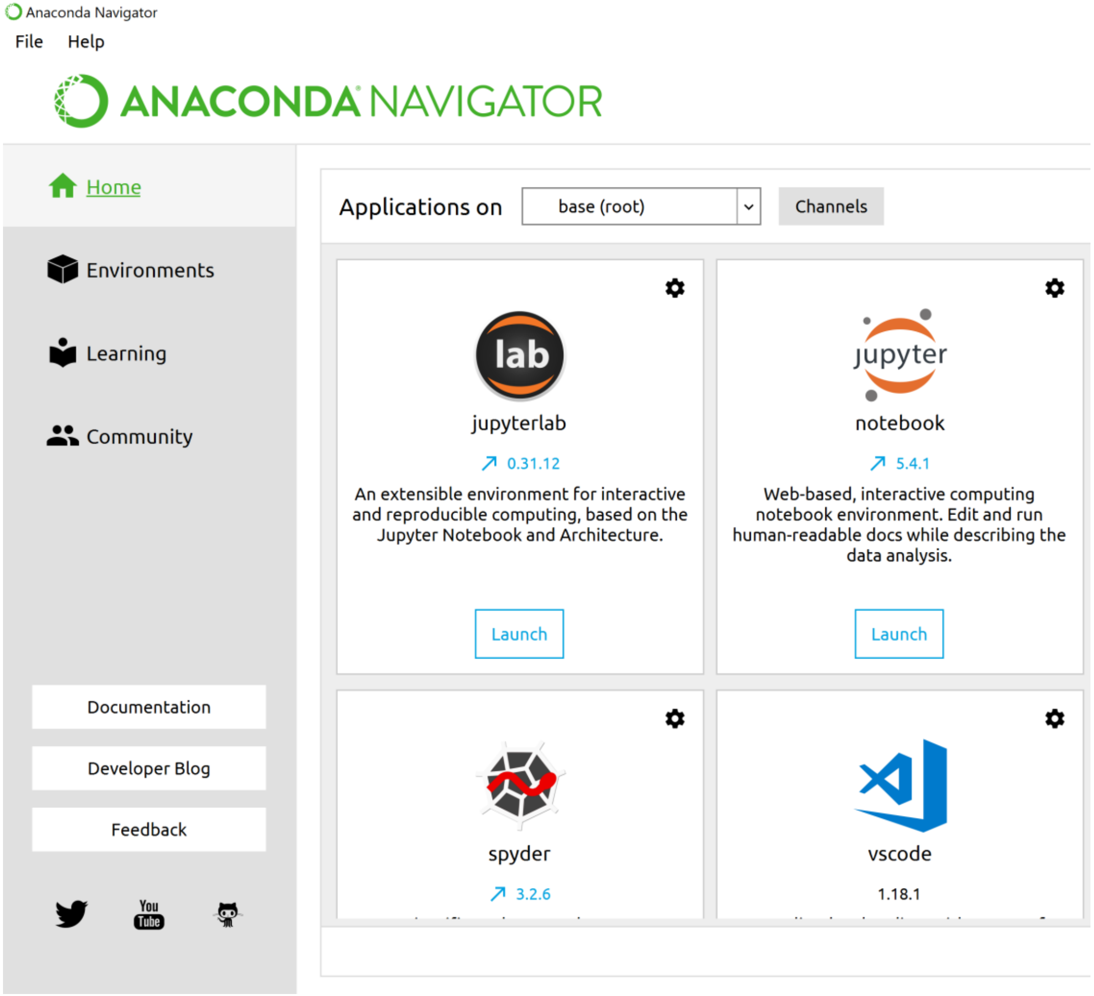
- In class we will be using the **Anaconda** Distribution which is based on the conda package manager

- It provides integration with Jupyter, Virtual Envs, etc.

Anaconda

Anaconda Navigator

- The Navigator is a main landing page for working with your python environment.
- Here we can launch editors (spyder, jupyter notebook, etc.) to write and develop python code
- In addition we can manage (install packages, etc.) our python environment

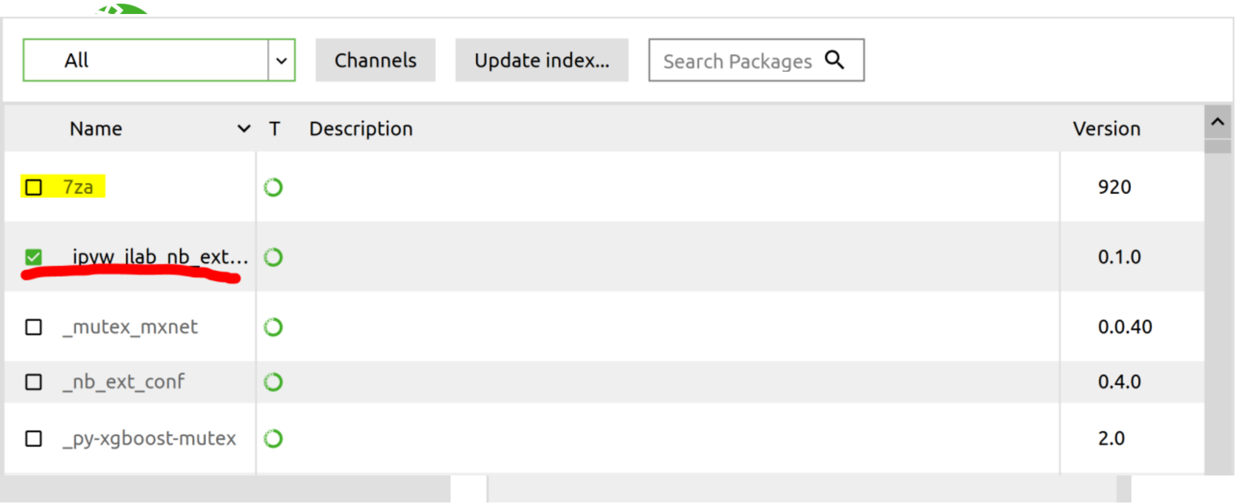


jupyterlab: An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

notebook: Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

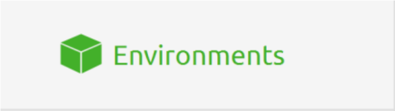
Anaconda Environments

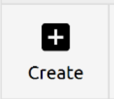
- Clicking on the “Environment” tab will show us what environments are available in Anaconda
  - In the simplest terms, an anaconda “environment” is a self-contained collection of python packages.
- From the “Environment” tab we can see which packages are installed and which packages are available for installation.
  - If you click on a package for installation, you’ll be prompted to Apply your changes



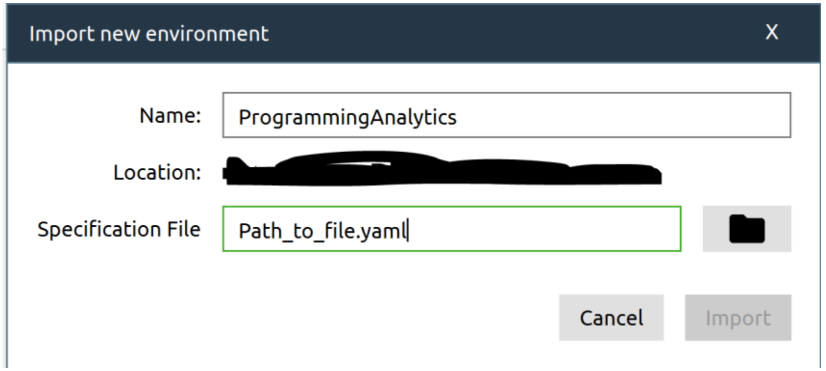
	Name	T	Description	Version
□	7za	○		920
☑	ipyw_jlab_nb_ext...	○		0.1.0
□	_mutex_mxnet	○		0.0.40
□	_nb_ext_conf	○		0.4.0
□	_py-xgboost-mutex	○		2.0

Setting Up Class Environment

- For this class I’ve provided an environment file (and a package list) on blackboard. This environment should include all of the packages necessary for the class and can be installed as follows:
  1. Navigate to the “Environment” tab in Anaconda. 

  2. Click on the “Create” button 

  3. Next fill in the specification file by navigating to the provided .yaml file
  4. Import



Import new environment

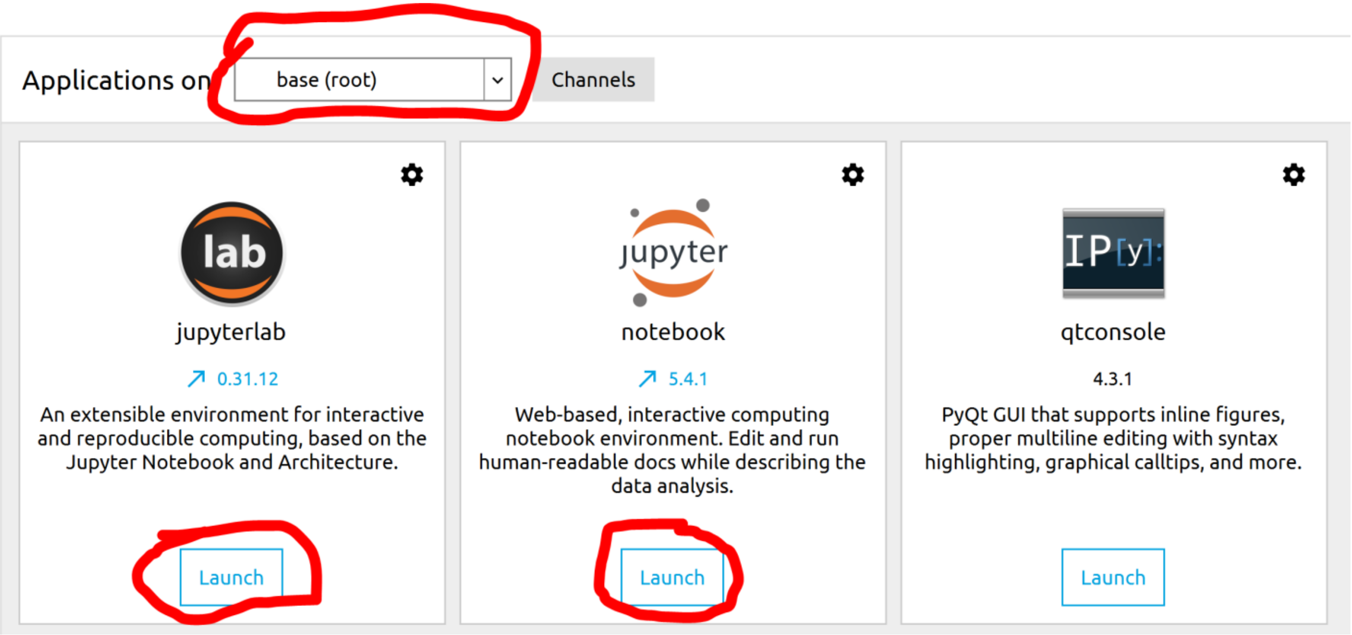
Name: ProgrammingAnalytics

Location: [redacted]

Specification File Path_to_file.yaml

Anaconda Applications

- On the home page we can choose which environment (base(root) in the image) we want to launch applications from.
- Clicking the "Launch" button on any of these applications will launch a separate window.



Applications on: base (root)

jupyterlab 0.31.12

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

notebook 5.4.1

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

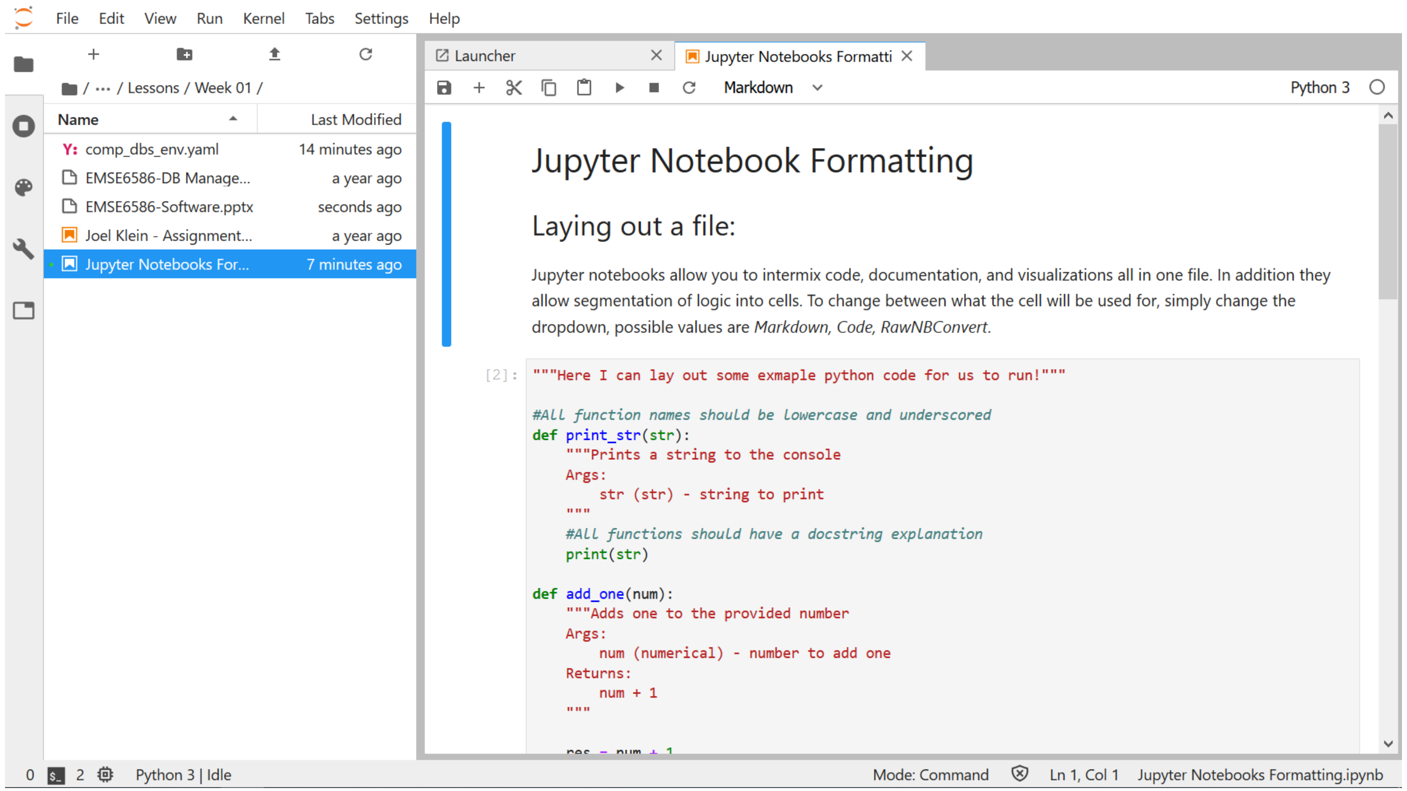
qtconsole 4.3.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Jupyterlab

JupyterLab

- Jupyter Notebooks and Jupyterlab are the IDEs we will use for this course.
- Jupyter Notebooks are used heavily throughout this course, as they provide a means to intermix text, code, and graphics.



Name	Last Modified
comp_dbs_env.yaml	14 minutes ago
EMSE6586-DB Manage...	a year ago
EMSE6586-Software.pptx	seconds ago
Joel Klein - Assignment...	a year ago
Jupyter Notebooks For...	7 minutes ago

Jupyter Notebook Formatting

Laying out a file:

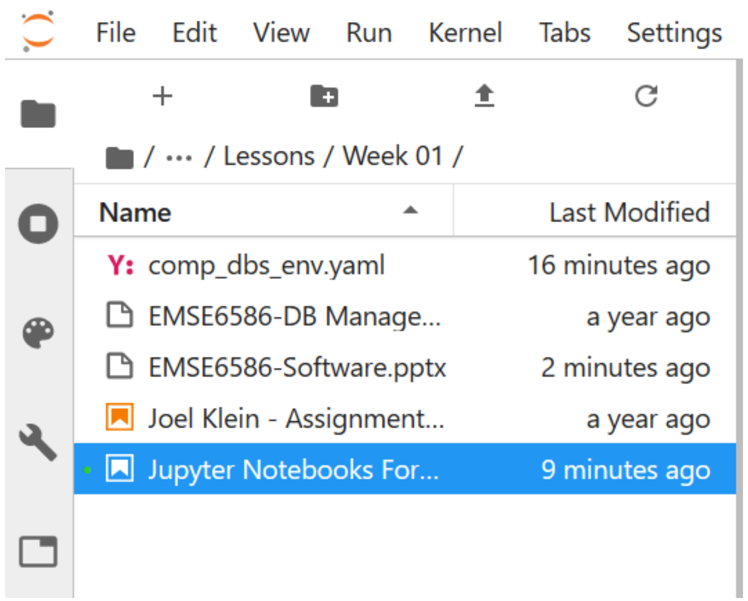
Jupyter notebooks allow you to intermix code, documentation, and visualizations all in one file. In addition they allow segmentation of logic into cells. To change between what the cell will be used for, simply change the dropdown, possible values are *Markdown, Code, RawNBConvert*.

[2]:

```
"""Here I can lay out some exmaple python code for us to run!"""

#All function names should be lowercase and underscored
def print_str(str):
    """Prints a string to the console
    Args:
        str (str) - string to print
    """
    #All functions should have a docstring explanation
    print(str)

def add_one(num):
    """Adds one to the provided number
    Args:
        num (numerical) - number to add one
    Returns:
        num + 1
    """


    res = num + 1
```

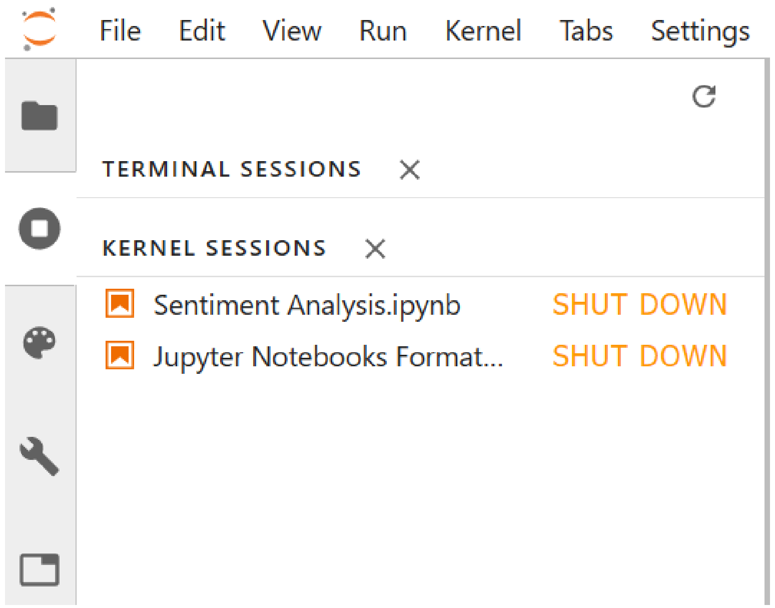
File Navigation and Kernels

We can navigate the file system using the left hand panel



/ ··· / Lessons / Week 01 /

Name	Last Modified
comp_dbs_env.yaml	16 minutes ago
EMSE6586-DB Manage...	a year ago
EMSE6586-Software.pptx	2 minutes ago
Joel Klein - Assignment...	a year ago
Jupyter Notebooks For...	9 minutes ago



TERMINAL SESSIONS

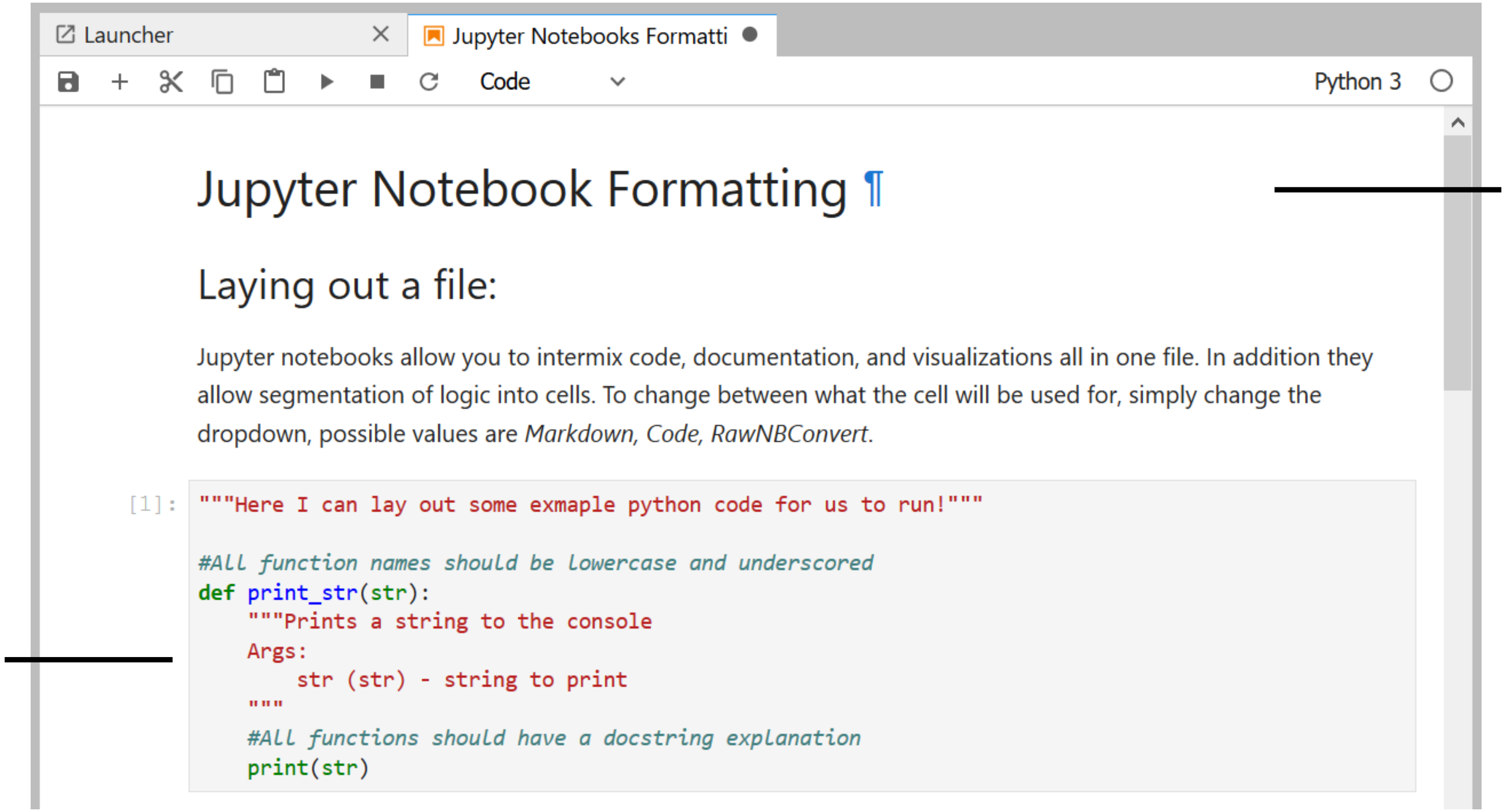
KERNEL SESSIONS

Sentiment Analysis.ipynb SHUT DOWN

Jupyter Notebooks Format... SHUT DOWN

In addition we can monitor the running kernels within that same panel

Working with Notebooks



Jupyter Notebook Formatting ¶

Laying out a file:

Jupyter notebooks allow you to intermix code, documentation, and visualizations all in one file. In addition they allow segmentation of logic into cells. To change between what the cell will be used for, simply change the dropdown, possible values are *Markdown, Code, RawNBConvert*.

[1]:

```
"""Here I can lay out some exmaple python code for us to run!"""

#ALL function names should be lowercase and underscored
def print_str(str):
    """Prints a string to the console
    Args:
        str (str) - string to print
    """
    #ALL functions should have a docstring explanation
    print(str)
```

Notebooks let us mark up (using markdown) our code to provide context and information.

The primary function is to provide cells of execution for our code/logic

The Databases

MySQL
Type – SQL | Environment – AWS RDS

- We will be interacting with an AWS RDS instance running MySQL to learn about SQL query languages
  - RDS supports several different SQL types

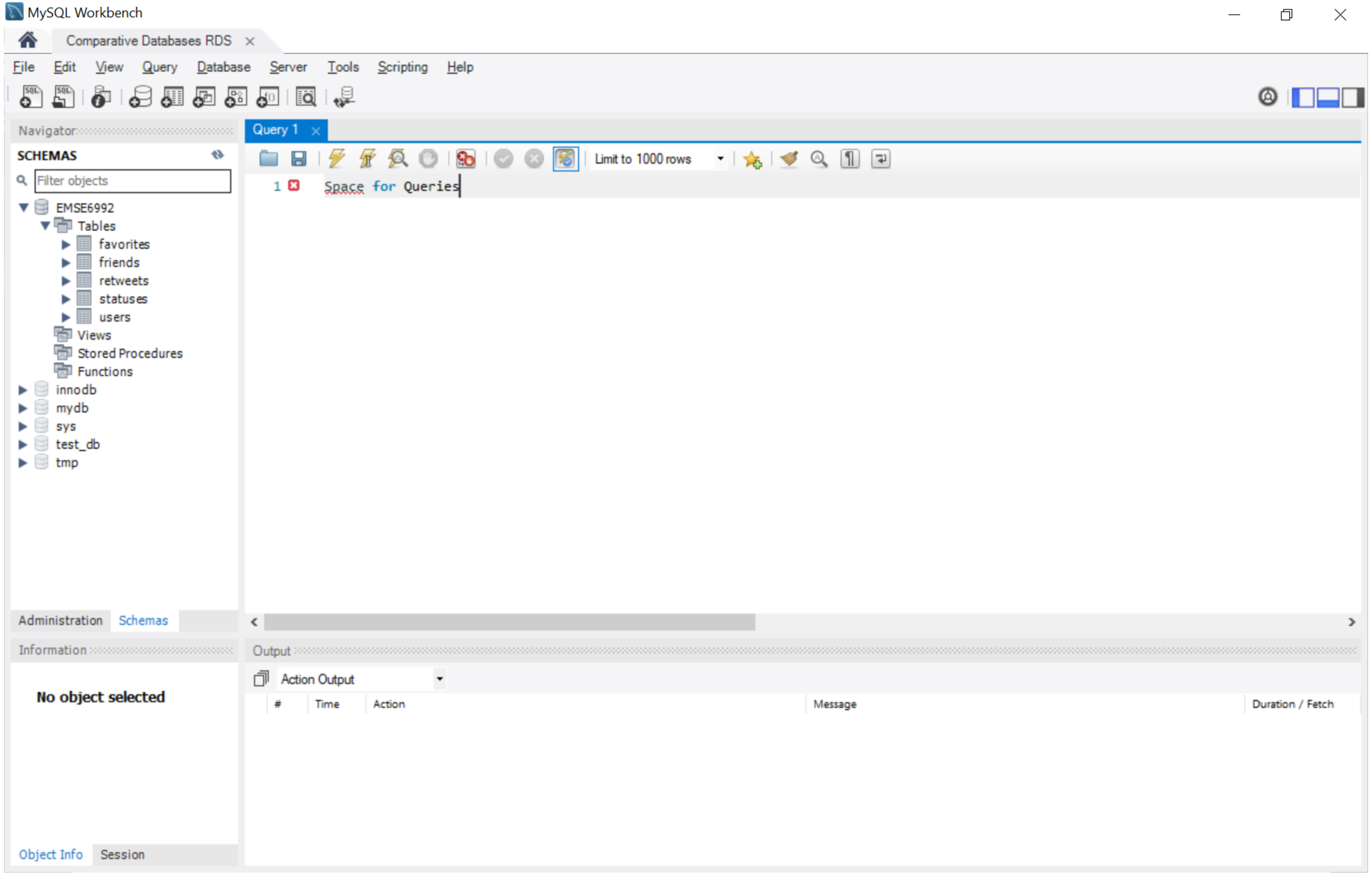
- For non-programmatic access we will be utilizing MySQL Workbench (https://www.mysql.com/products/workbench/)

- For programmatic access we will be using the **pymysql** package for python
  - This can be downloaded through anaconda or through pip

MySQL Workbench

This is a free tool for managing MySQL instances and running queries and working with database schemas

You can download it for free at: https://www.mysql.com/products/workbench/



```
Space for Queries
```

Mongo DB:
Type – Document | Environment – EC2 Instance

- For this course we will be using an instance of Mongo DB running on an AWS EC2 instance
  - Alternatively, there is Azure Cosmos and AWS DocumentDB which both support the Mongo API in managed environments

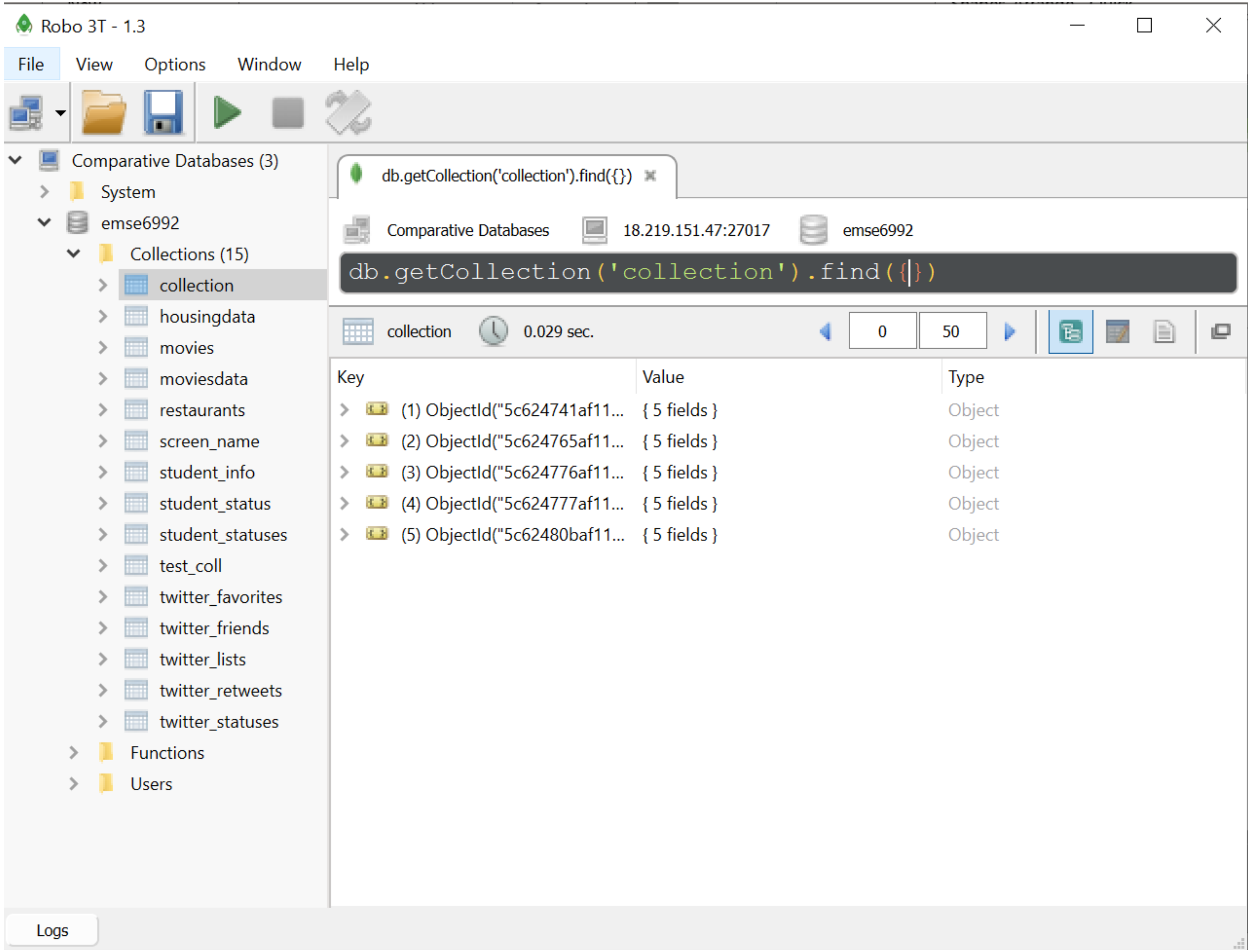
- For non-programmatic access we will be using Robo3T

- For programmatic access we will be using python and the pymongo package

Robo3T

This is a free tool for querying databases that support the mongo API and provides some minimal management options

You can download it for free at: https://robomongo.org/download



```
db.getCollection('collection').find({})
```

Key	Value	Type
(1) ObjectId("5c624741af11...	{ 5 fields }	Object
(2) ObjectId("5c624765af11...	{ 5 fields }	Object
(3) ObjectId("5c624776af11...	{ 5 fields }	Object
(4) ObjectId("5c624777af11...	{ 5 fields }	Object
(5) ObjectId("5c62480baf11...	{ 5 fields }	Object

Arango:
Type – Graph | Environment – EC2 Instance

- For this course we will be using an instance of Arango running on an AWS EC2 instance
  - Alternatives are Azure Cosmos and AWS Neptune for managed solutions

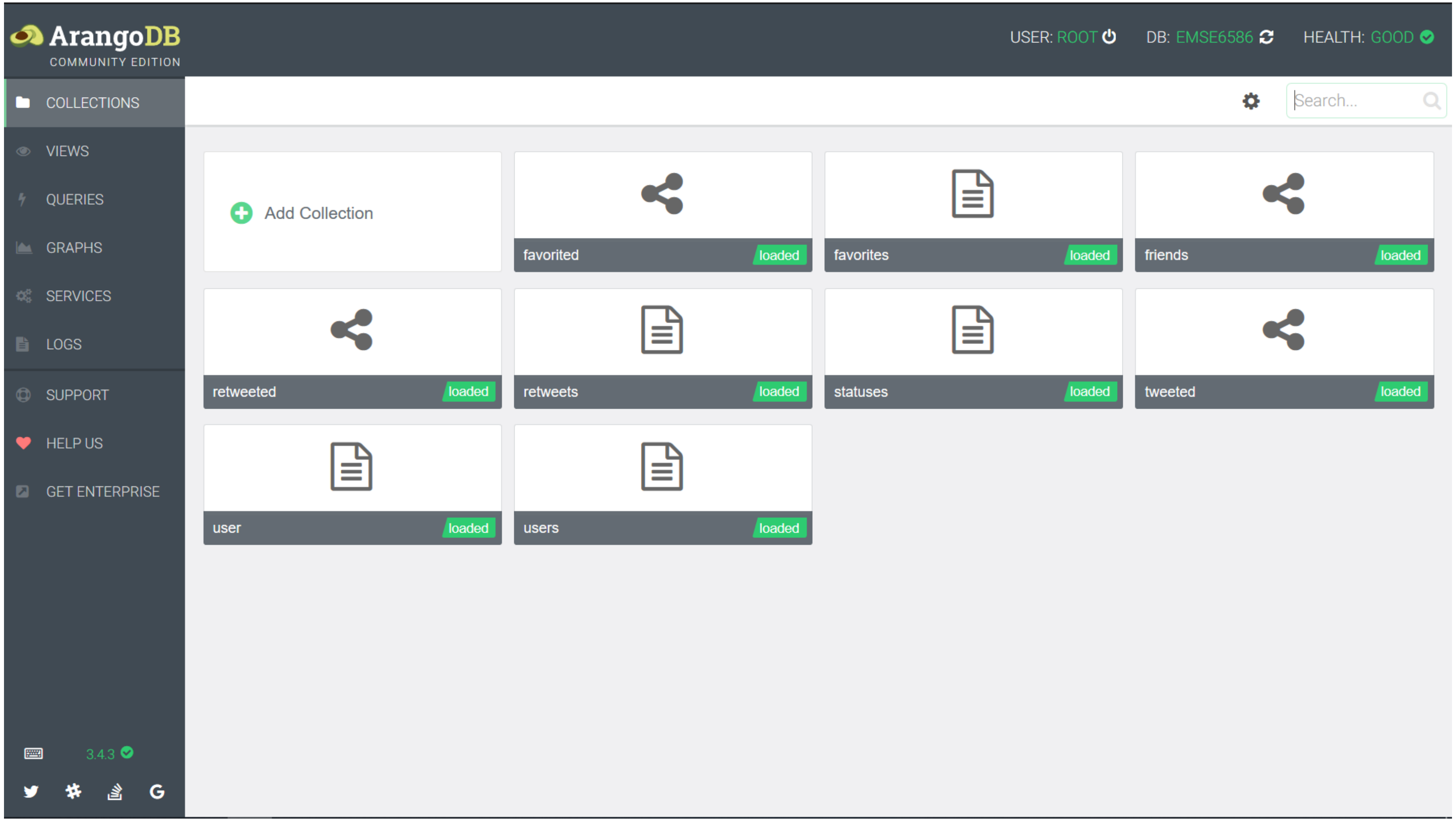
- For non-programmatic access we will be using a web browser (http://18.219.151.47:8529)

- For programmatic access we will be using python and python-arango package

Arango Web-GUI

Arango natively supports a web GUI for managing the database, running queries, and visualizing graphs

You can download it for free at: http://18.219.151.47:8529



USER: ROOT

DB: EMSE6586

HEALTH: GOOD

Cloud Database Solutions

Cloud Databases

- While the means in which I host our database resources leverage cloud infrastructure, they are all open-source solutions not dependent on cloud infrastructure

- The following database solutions are approaches to handling data that are more unconventional and rely more heavily on services.

Azure Cosmos

- Azure Cosmos is a multi-model database provided by Microsoft on their Azure cloud.
  - Multi-Model means that the database supports multiple models of interacting with the data

- The benefits of Cosmos is that it provides the following:
  - Managed Solution
  - "Infinitely" scalable
  - Multi-Model
  - Etc.

AWS Athena

- AWS Athena is Amazon’s approach to SQL-like access to static files.
    - Essentially Athena is focused on providing queryable access to files stored in S3

- The benefits of Athena are:
    - Quick and Flexible access to existing data
    - A Serverless approach to SQL-like interfaces

Hadoop and Spark

Hadoop/Spark Infrastructure

- Time permitting we will also dive into the Hadoop and Spark architecture

- Hadoop is a pseudo database architecture that has evolved into a ecosystem of tools and services for processing large volumes of data

End Slide

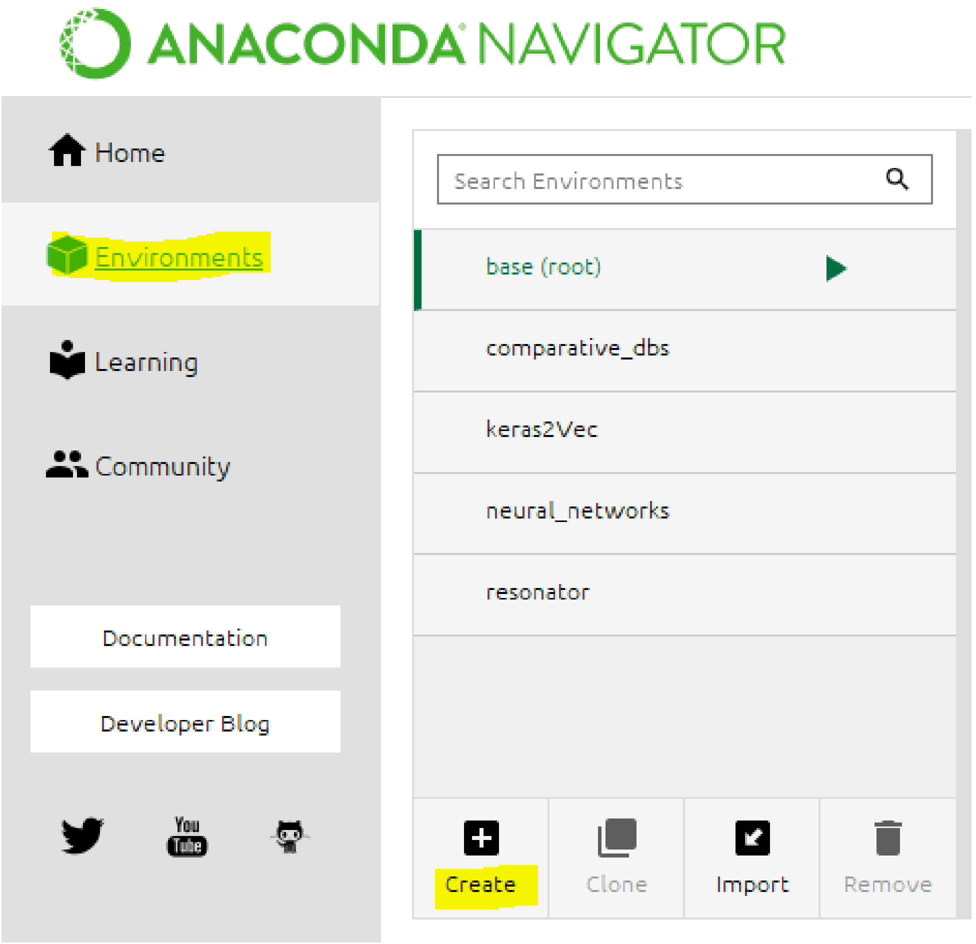
EMSE 6586 – DBMS for Data Analytics

Environment Problems?

Pip Install to Environment

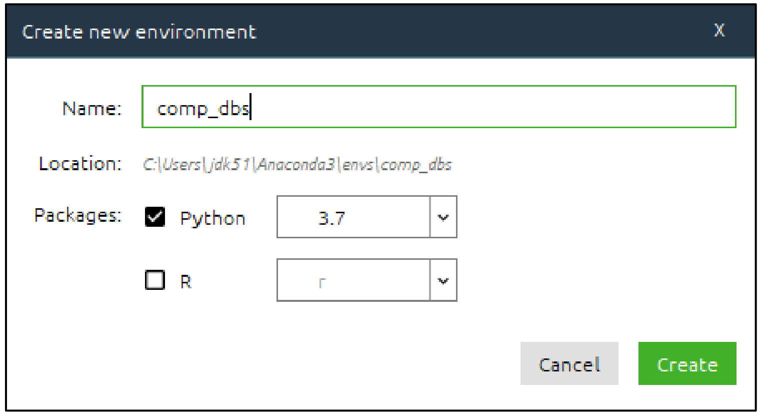
- If Anaconda gives you some problems with environment yaml file, then pip is your next best option.

- To install via pip, you'll want to create a new environment manually.
  1. Click on Environments on the left-hand side
  2. Click on ➕ Create at the bottom



Creating a New Environment

- In the resulting pop-up
  1. Provide a name for the environment
  2. Select “Python 3.7”



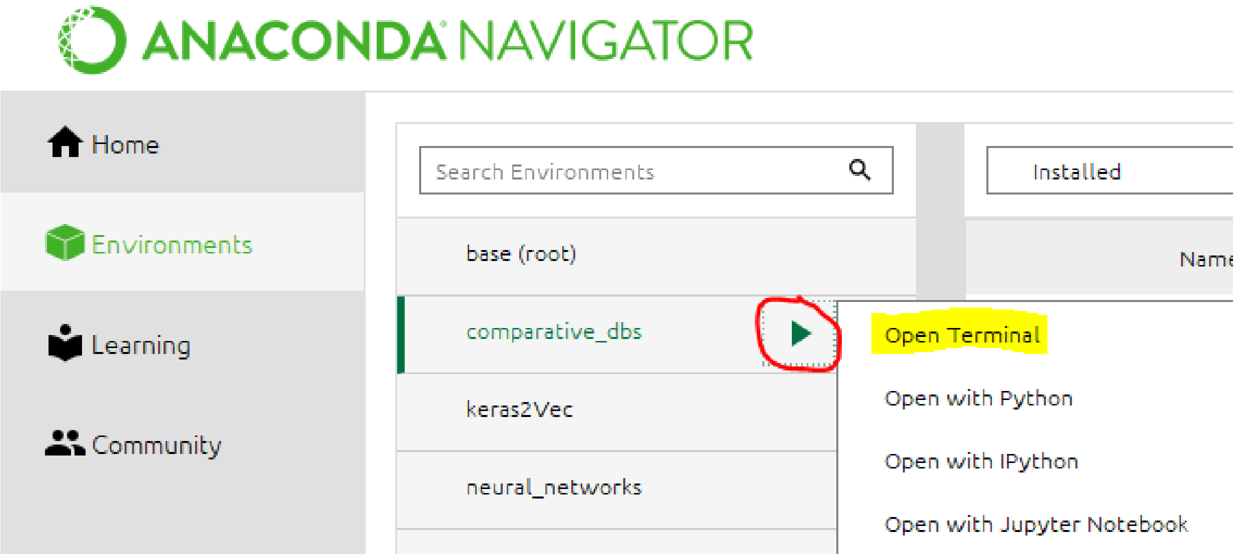
Name: comp_dbs

Location: C:\Users\jdk51\Anaconda3\envs\comp_dbs

Packages: ☑ Python 3.7

□ R r

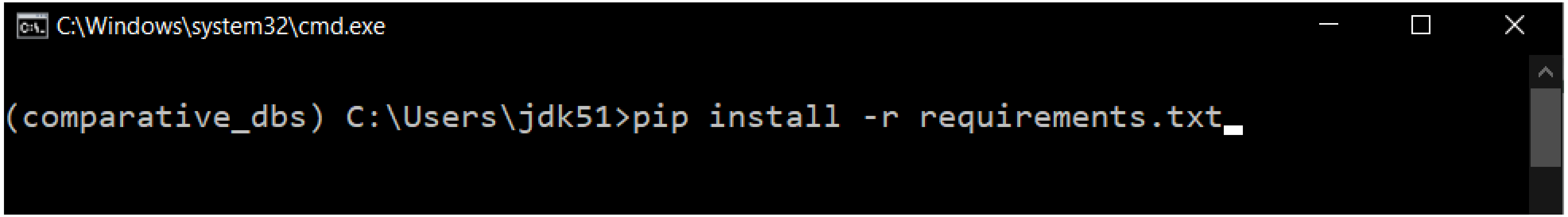
- Once the environment is created, you’ll need to open a terminal.
  1. Click on your environment
  2. Hit ▶ that appears
  3. Click on “Open Terminal”



Finally Pip Install

- In the resulting window you’ll be able to install the requirements for the environment.

- Simply run “pip install –r requirements.txt”
  - Note: Make sure that you provide the correct pathing to the requirements.txt file



```
(comparative_dbs) C:\Users\jdk51>pip install -r requirements.txt
```