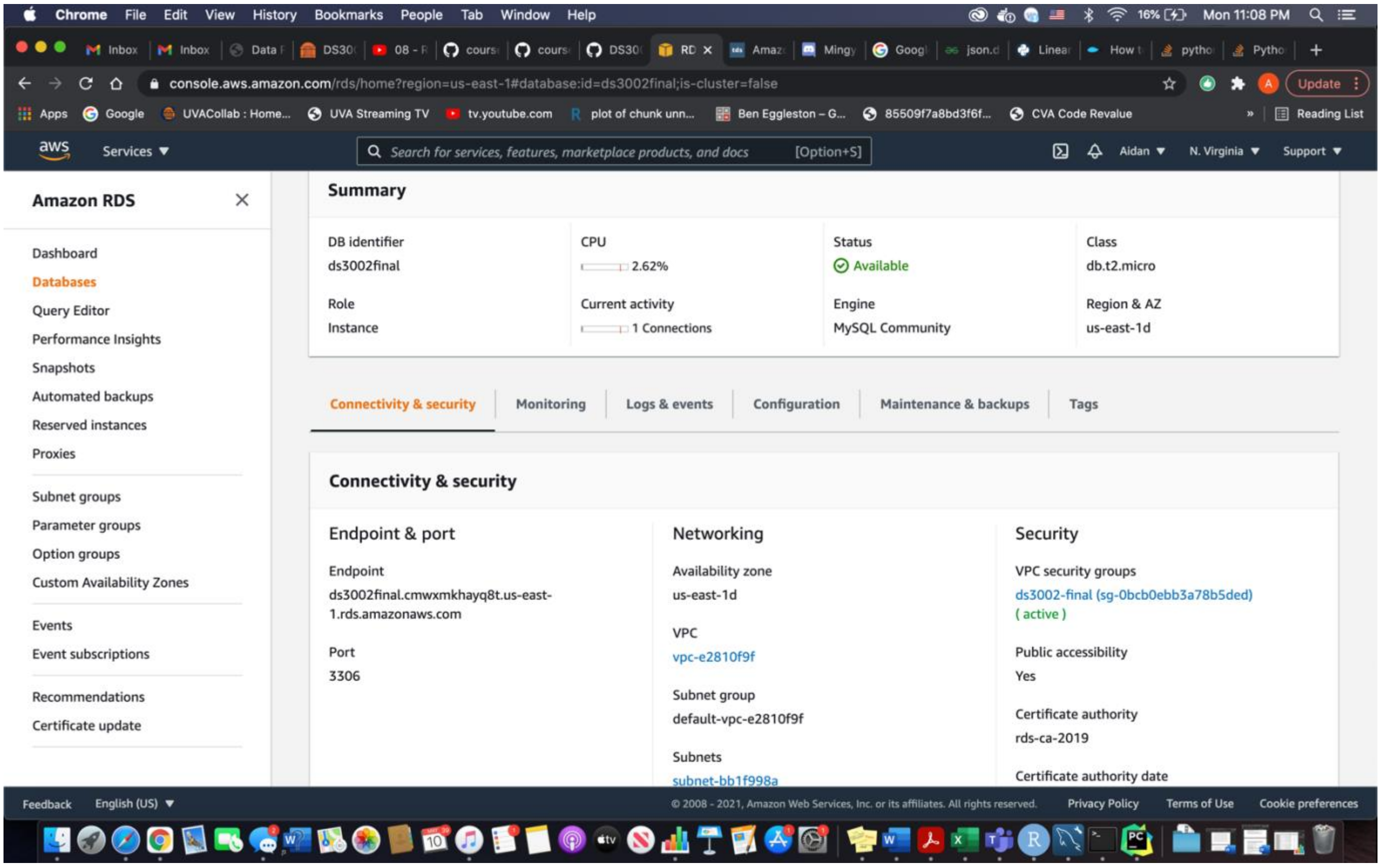
The purpose of this script is to pull data from a given API and upload it to a SQL database as well as conduct analysis on that data. It is very simple to use; if the desired API is already in place, running the script will show the table and the analysis. If the user wanted to pull from a different API, the fields are well marked so that changing them should be simple. There would also need to be manual changes for the analysis because the variables are hard-coded in and the axis and title labels are customized.

Documentation: (*there are comments within the code to supplement)

Setup:

1. I created an instance in AWS RDS
    - I made sure to create a username/pw combination
    - Ensured the correct availability zone among other default settings



Summary

DB identifier ds3002final	CPU 2.62%	Status Available	Class db.t2.micro
Role Instance	Current activity 1 Connections	Engine MySQL Community	Region & AZ us-east-1d

Connectivity & security

Endpoint & port

Endpoint
ds3002final.cmwxmkhayq8t.us-east-1.rds.amazonaws.com

Port
3306

Networking

Availability zone
us-east-1d

VPC
vpc-e2810f9f

Subnet group
default-vpc-e2810f9f

Subnets
subnet-bb1f998a

Security

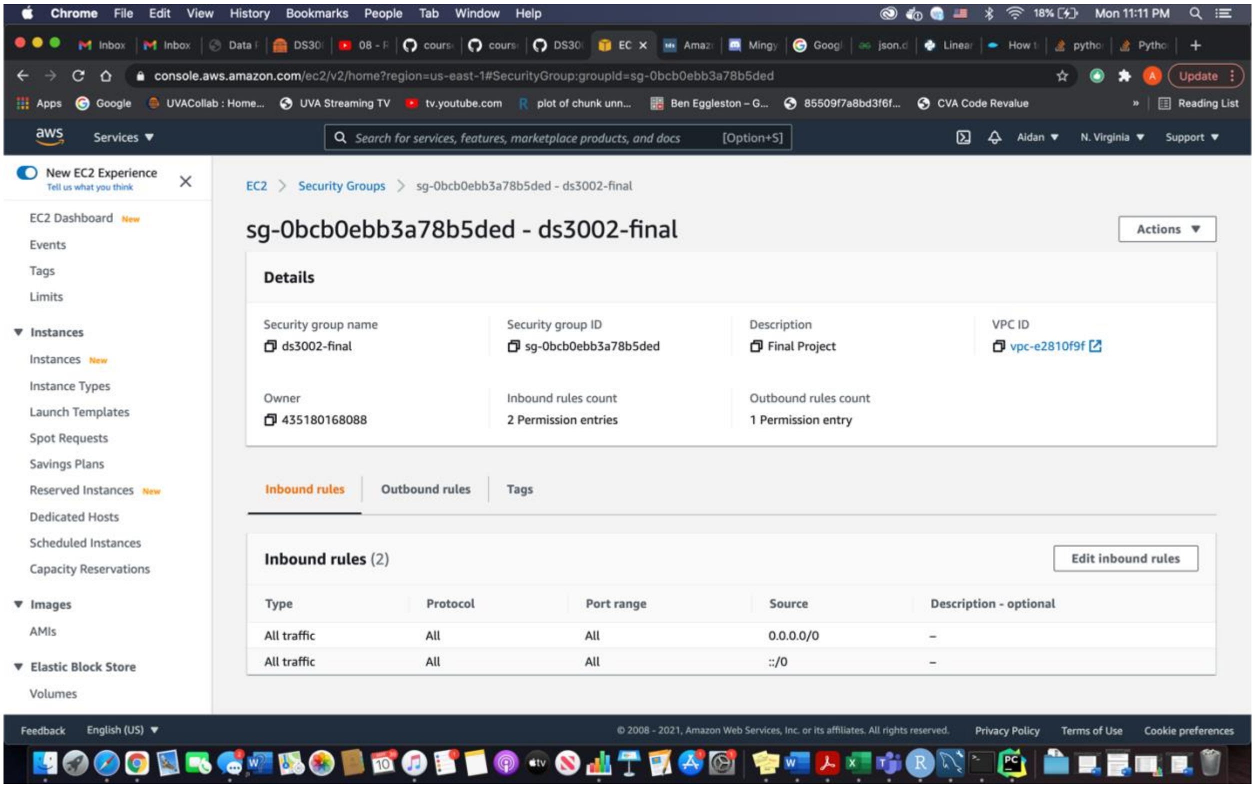
VPC security groups
ds3002-final (sg-0bcb0ebb3a78b5ded)
(active)

Public accessibility
Yes

Certificate authority
rds-ca-2019

Certificate authority date

2. I created a VPC security group for my RDS instance
   - I added an inbound rule to allow all traffic from any source
   - Then, I changed the VPC security to my newly created EC2 security group
   - I also checked the outbound rules, which also allowed traffic anywhere and from any source



console.aws.amazon.com/ec2/v2/home?region=us-east-1#SecurityGroup:groupId=sg-0bcb0ebb3a78b5ded

EC2 > Security Groups > sg-0bcb0ebb3a78b5ded - ds3002-final

sg-0bcb0ebb3a78b5ded - ds3002-final

Details

Security group name	Security group ID	Description	VPC ID
ds3002-final	sg-0bcb0ebb3a78b5ded	Final Project	vpc-e2810f9f
Owner	**Inbound rules count**	**Outbound rules count**	
435180168088	2 Permission entries	1 Permission entry	

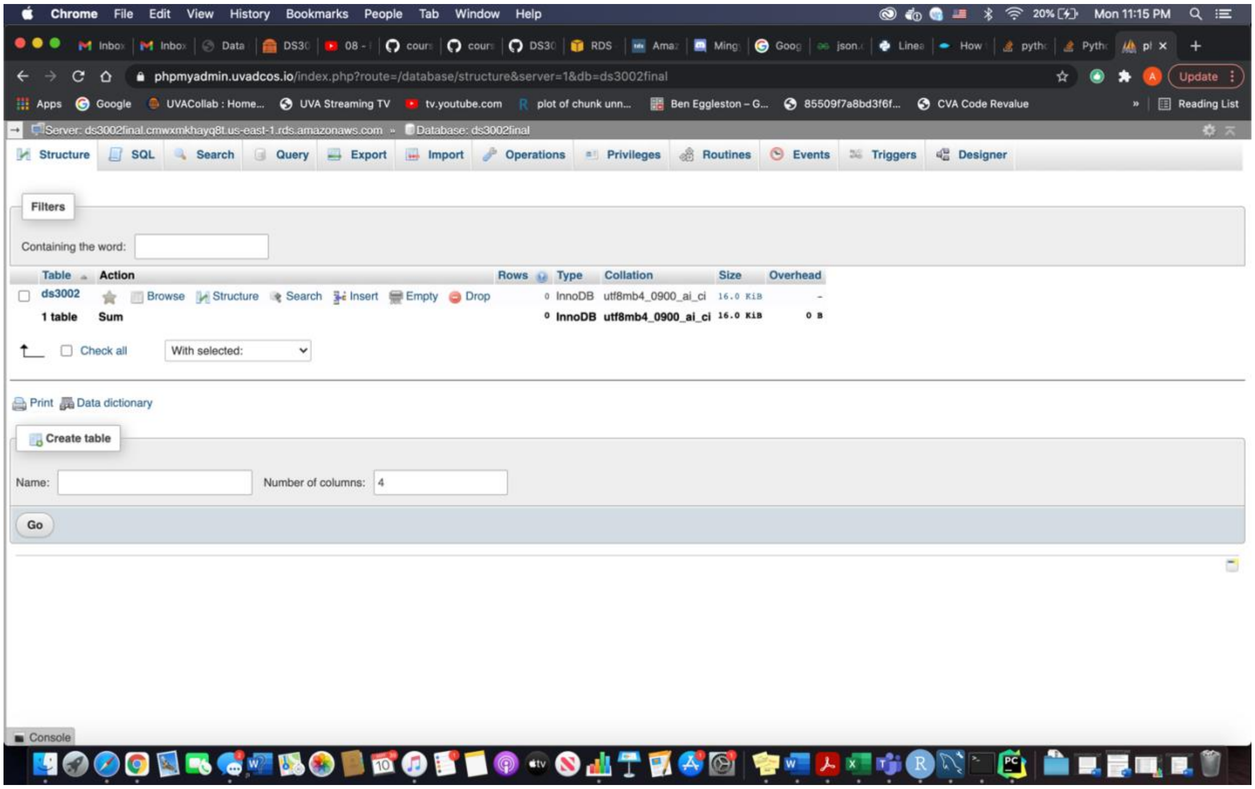
Inbound rules (2)

Type	Protocol	Port range	Source	Description - optional
All traffic	All	All	0.0.0.0/0	–
All traffic	All	All	::/0	–

(see code for the following)

3. I accessed my database using my Python script using the information I made when I created my RDS instance.
4. I created a function to iterate 60 times, once every 60 seconds
5. I read in the API and stored the response
6. I tried to read it to the SQL database using both JSON and text, but both were unsuccessful

7. Then I created my SQL database, and connected it to my AWS RDS instance
   - Within that database, I created a table with columns for factor, pi, and time



phpmyadmin.uvadcos.io/index.php?route=/database/structure&server=1&db=ds3002final

Server: ds3002final.cmwxmkhayq8t.us-east-1.rds.amazonaws.com » Database: ds3002final

Filters

Containing the word:

	Table	Action	Rows	Type	Collation	Size	Overhead
□	ds3002	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_0900_ai_ci	16.0 KiB	-
	1 table	Sum	0	InnoDB	utf8mb4_0900_ai_ci	16.0 KiB	0 B

□ Check all With selected:

Print Data dictionary

Create table

Name: Number of columns: 4

Go

(see code)

8. After the function had finished running, the data needed to be queried back to the script for analysis
9. Analysis was conducted

Some Problems:

I was unable to get the data into the SQL database. I spent a lot of time researching different methods as well as asking the TA for help, but even the TA was unable to figure out the problem. I tried to use both JSON and text (and many, many variations of them) to store the response from the API, but both were unsuccessful. In the end, my code seemed correct to both myself and the TA, but it would consistently show error messages during this step.