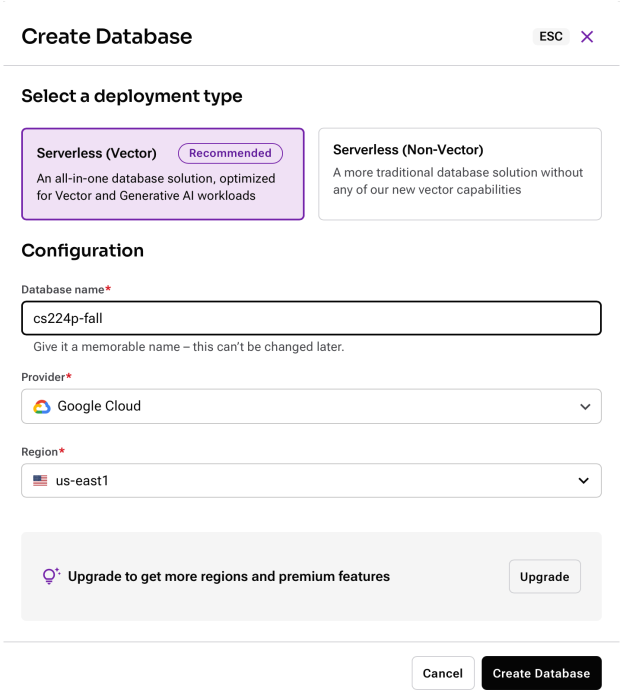
Fall 2024 CS 224P **Instructor: Prof. Chen Li** **TAs: Xiaozhen Liu, Jiadong Bai**

Homework Assignment #2 Setup

(Apache Cassandra)

For this assignment, we will be using an online DBaaS service called DataStax Astra; it provides a "serverless" cloud-based implementation of Apache Cassandra. In this tutorial, we will use a small example database called “Hoofers” to walk you through the steps needed to get started with Astra/Cassandra. This textbook example contains 3 tables, namely “Boats”, “Sailors”, and “Reserves”. We will also be using your PostgreSQL database from Assignment #1 to create and export CSV files, so make sure you keep your PostgreSQL database on hand (or rerun a prefix of your HW1 script to recreate it).

1. Go to https://astra.datastax.com/ and sign up for a free Astra account.
2. After verifying your email and answering a few questions, you will be directed to your DataStax dashboard. Your free account is allowed up to $25 for free for Astra service use, and you should not reach that limit while doing this assignment!
3. Go to the “Databases” tab and create a database called **cs224p-fall.** You will need to choose between two deployment types. Both the “Severless (Vector)” and the “Serveless (Non-Vector)” deployment types will work for this assignment. This instruction will use the “Serverless (Vector)” option. For the “Provider” and “Region” sections, choose **Google Cloud** and **us-east1**.


Create Database

Select a deployment type

[✓ Serverless (Vector) Recommended
An all-in-one database solution, optimized for Vector and Generative AI workloads]

Serverless (Non-Vector)
A more traditional database solution without any of our new vector capabilities

Configuration

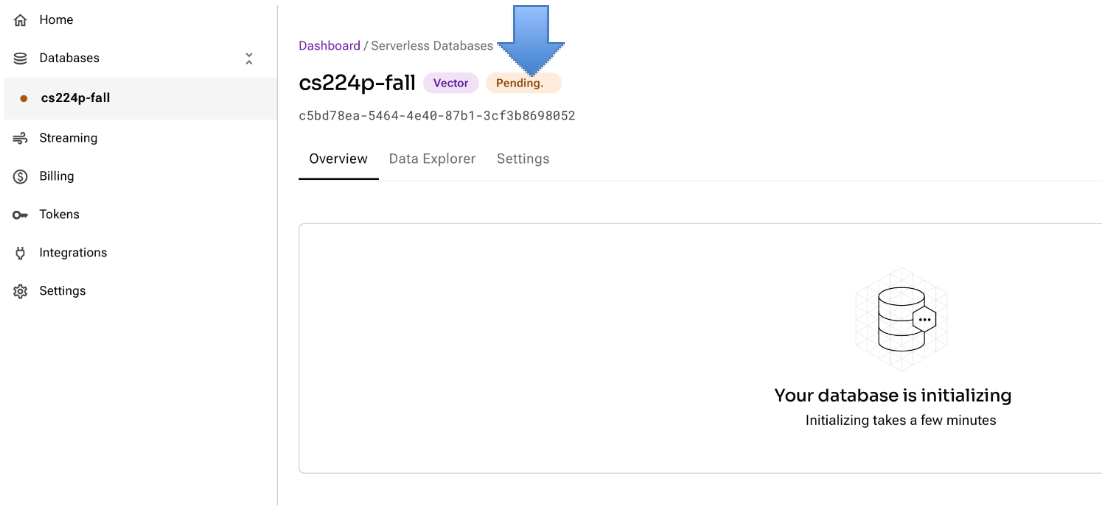
Database name*: cs224p-fall
Give it a memorable name – this can't be changed later.

Provider*: Google Cloud

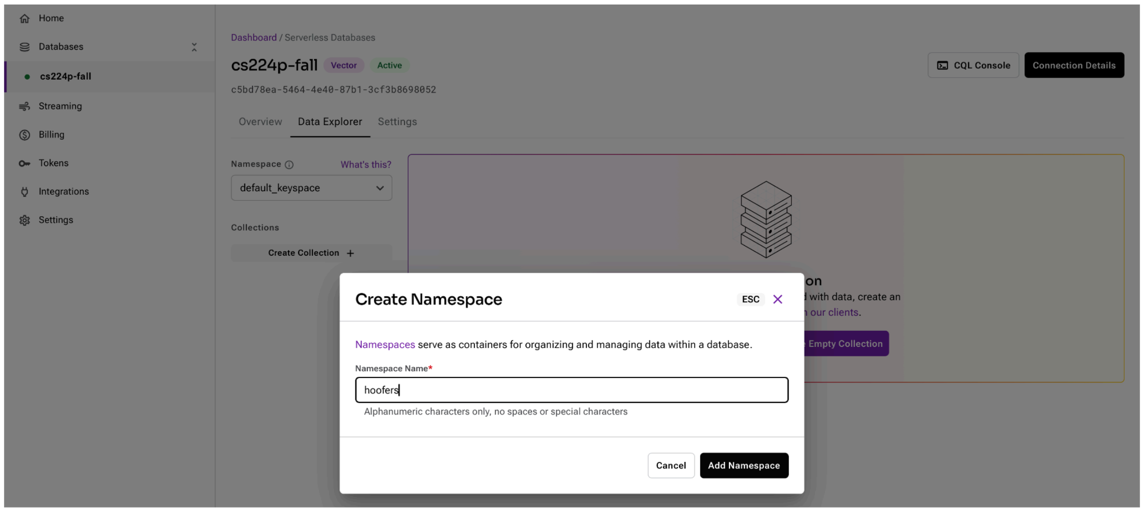
Region*: us-east1

Upgrade to get more regions and premium features

Note: your database might take a few minutes to be successfully created. While it is being created, you will see “pending” next to the database name.

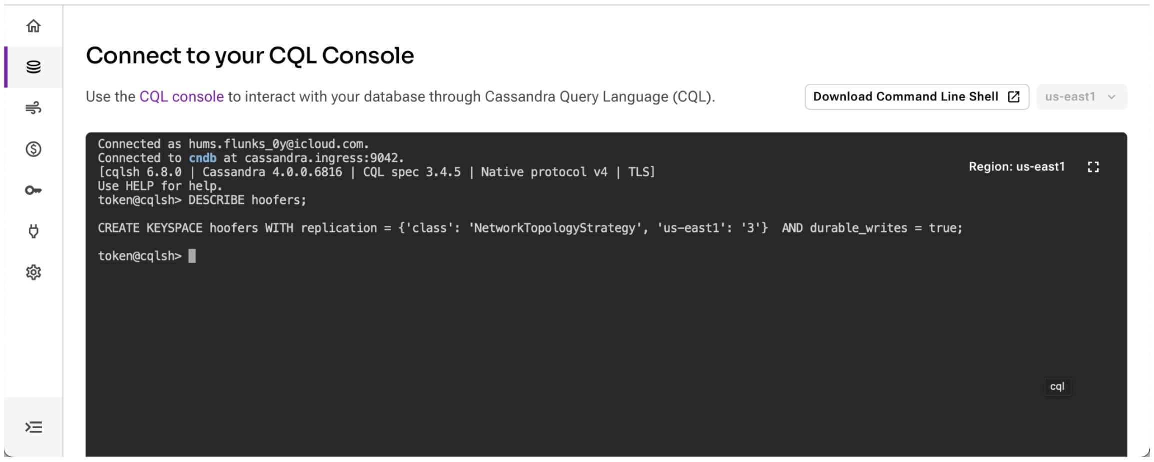

Home
Databases
cs224p-fall
Streaming
Billing
Tokens
Integrations
Settings
Dashboard / Serverless Databases
cs224p-fall
Vector
Pending.
c5bd78ea-5464-4e40-87b1-3cf3b8698052
Overview
Data Explorer
Settings
Your database is initializing
Initializing takes a few minutes

4. Once you receive confirmation that your database was created, head over to the dashboard by clicking on your database name. Go to “Data Explorer” and create a keyspace (also known as namespace) called “**hoofers**” which we will load data into later. It will also take some time for the keyspace to be successfully created. *Note Astra only allows creating a keyspace via the UI.*



5. Open the CQL console by clicking “CQL Console”. This is where you will be running your queries. In the console, execute the following command to verify if the keyspace has been successfully created.

```
DESCRIBE hoofers;
```


Connect to your CQL Console

Use the CQL console to interact with your database through Cassandra Query Language (CQL).

```
Connected as hums.flunks_0y@icloud.com.
Connected to cndb at cassandra.ingress:9042.
[cqlsh 6.8.0 | Cassandra 4.0.0.6816 | CQL spec 3.4.5 | Native protocol v4 | TLS]
Use HELP for help.
token@cqlsh> DESCRIBE hoofers;

CREATE KEYSPACE hoofers WITH replication = {'class': 'NetworkTopologyStrategy', 'us-east1': '3'}  AND durable_writes = true;

token@cqlsh> 
```

6. For purposes of loading data into Datastax Astra, we will be using a software tool called the **DataStax Bulk Loader**. This is to make sure that you learn how to manually create tables in CQL before loading data, and also that you experience the task of uploading bulk data into a database service in the cloud (a skill whose relevance is rapidly increasing over time).
7. Download DSBulk from the following link: https://downloads.datastax.com/#bulk-loader and unpack the downloaded distribution.
   Note: DSBulk requires a Java Runtime Environment. (If you do not have a JRE, make sure to download that as well, but most of you probably have Java set up already.)
8. Add the downloaded file to your PATH so that you are able to use the dsbulk command. This process can differ depending on your OS.
   a. For MAC/Linux users: on the command line run the following:

```
export PATH=path-to-unpacked-location/dsbulk-1.10.0/bin:$PATH
```

   (You might want to add this to your favorite shell's .xxxrc file. Please make sure .xxxrc file has the correct PATH, of bin, and reload the terminal once changes have been made to .xxxrc by saying, e.g., source ~/.bashrc .)

   b. For Windows users: the following link might help.

https://www.architectryan.com/2018/03/17/add-to-the-path-on-windows-10/

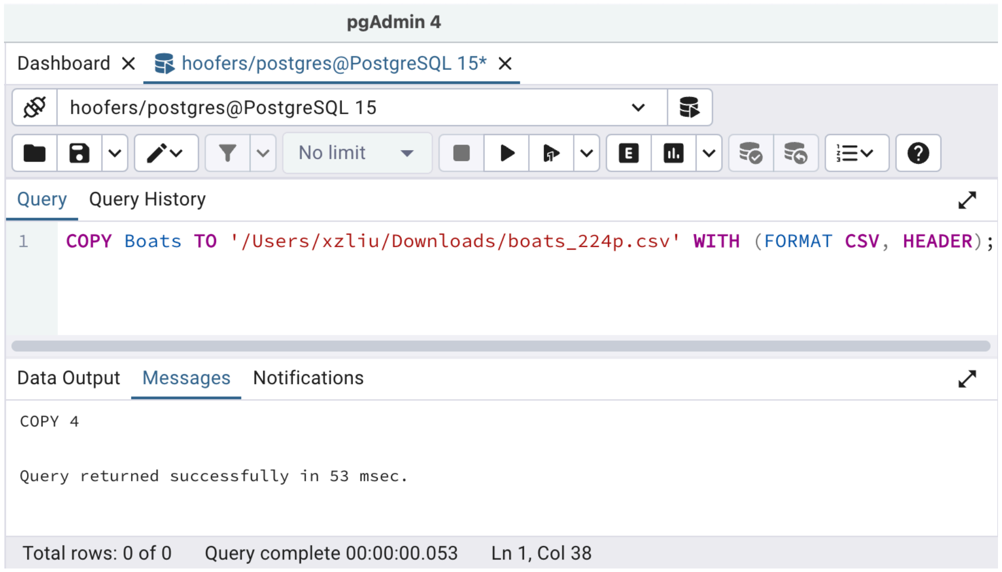
9. Verify that dsbulk is working properly by running the command:

```
$ dsbulk --version

DataStax Bulk Loader v1.11.0
```

Let's test out the workflow involved in loading data into Astra by first exporting data from PostgreSQL and then importing it using DSBulk by experimenting with the 'hoofers' data.

10. Create a database called "**hoofers**" in PostgreSQL. Run the following SQL script in PostgreSQL to create the tables and insert the data: [Download HoofersDB.sql]
11. Export the "**Boats**" table into a CSV file; below are some useful hints on how to do so in PostgreSQL either using pdAdmin or command line:

    Using the Query Tool in pgAdmin:



```
COPY Boats TO '/Users/xzliu/Downloads/boats_224p.csv' WITH (FORMAT CSV, HEADER);
```

```
COPY 4

Query returned successfully in 53 msec.
```

Total rows: 0 of 0 Query complete 00:00:00.053 Ln 1, Col 38

Using pSQL command line:

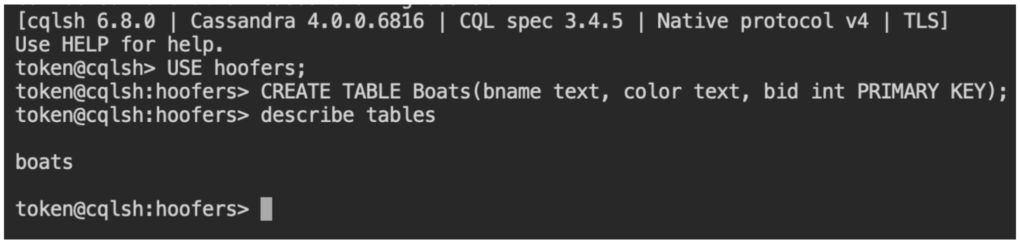
```
-- Suppose you want to fully export a table named Person into a csv file
called persons.csv, this file will be created at runtime --
\copy Person to '/desired-path/persons.csv' DELIMITER ',' CSV HEADER;

-- Now suppose you instead want to export the results of a query on the
Person table into a csv file called results.csv, this file will be created
at runtime --
\copy (SELECT * FROM Person P WHERE P.person_id = '1234') to
'/desired-path/persons.csv' DELIMITER ',' CSV HEADER;

-- The process would be the same for exporting the results of a JOIN query
--
\copy (SELECT * FROM Person P, Username U WHERE P.uid = U.uid) to
'/desired-path/persons.csv' DELIMITER ',' CSV HEADER;
```

Now that we have the CSV file, we can begin the process of loading data into Astra.

12. In the Astra CQL console, first execute the following command to switch to hoofers keyspace.

```
USE hoofers;
```

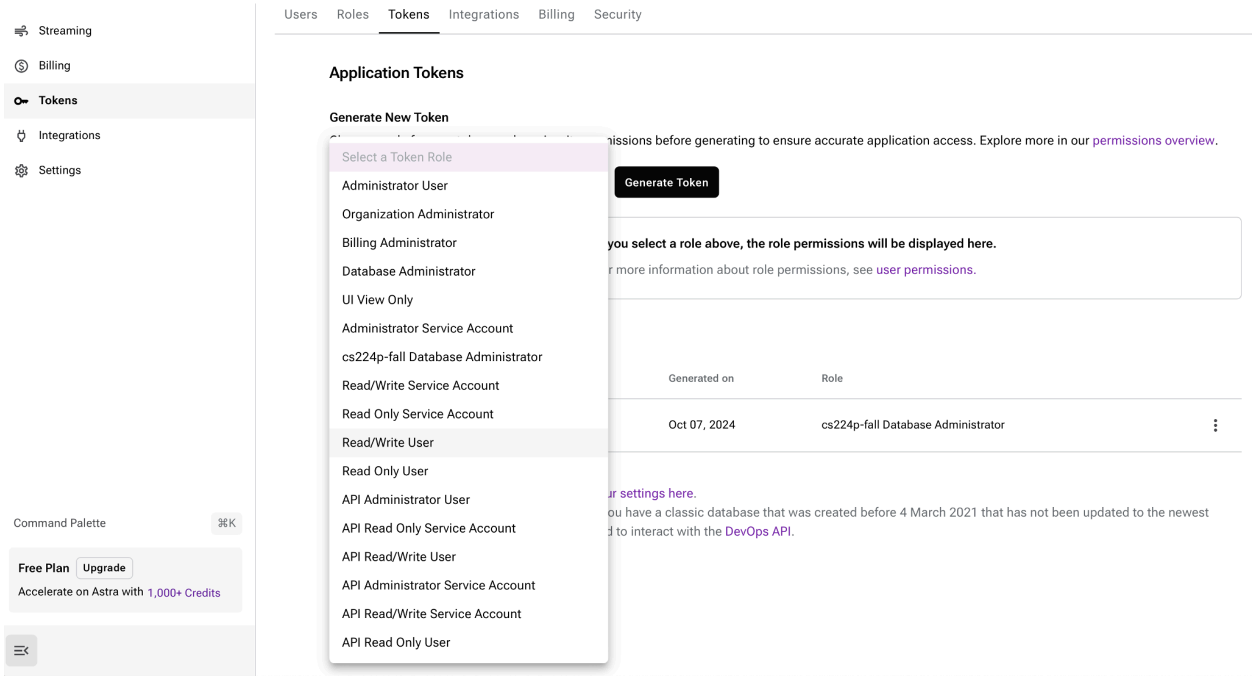
Then create a table called "Boats" by translating the PostgreSQL CREATE TABLE statement given in the script.



```
[cqlsh 6.8.0 | Cassandra 4.0.0.6816 | CQL spec 3.4.5 | Native protocol v4 | TLS]
Use HELP for help.
token@cqlsh> USE hoofers;
token@cqlsh:hoofers> CREATE TABLE Boats(bname text, color text, bid int PRIMARY KEY);
token@cqlsh:hoofers> describe tables

boats

token@cqlsh:hoofers> 
```

Next we need to load data into Cassandra. This involves the following four steps:

13. Generate a token:
    a. Select the "Tokens" tab from the left bar to go to the Application Tokens page.



Application Tokens

Generate New Token

issions before generating to ensure accurate application access. Explore more in our permissions overview.

you select a role above, the role permissions will be displayed here.

r more information about role permissions, see user permissions.

Generated on	Role
Oct 07, 2024	cs224p-fall Database Administrator

r settings here.

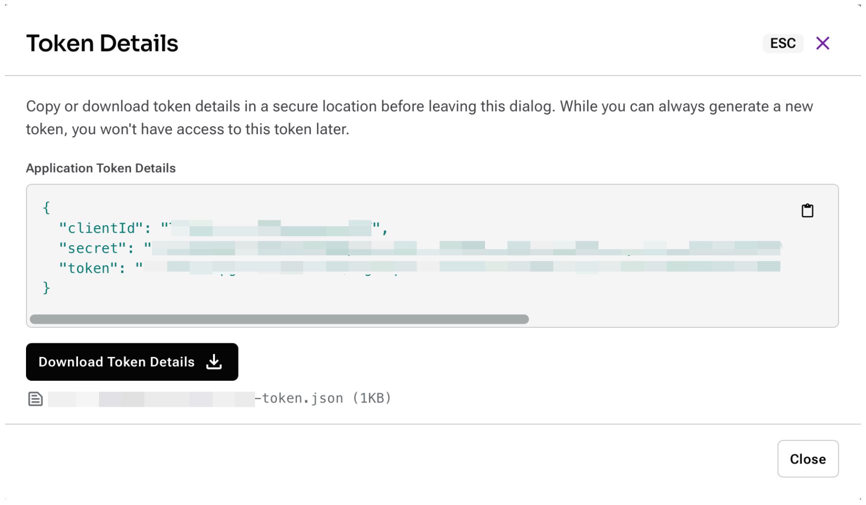
ou have a classic database that was created before 4 March 2021 that has not been updated to the newest

d to interact with the DevOps API.

Accelerate on Astra with 1,000+ Credits

    b. Select the role "Read/Write User" and click the "Generate Token" button.

c. You will be shown the token details with a link to download these as a csv file. Click “Download Token Details”.



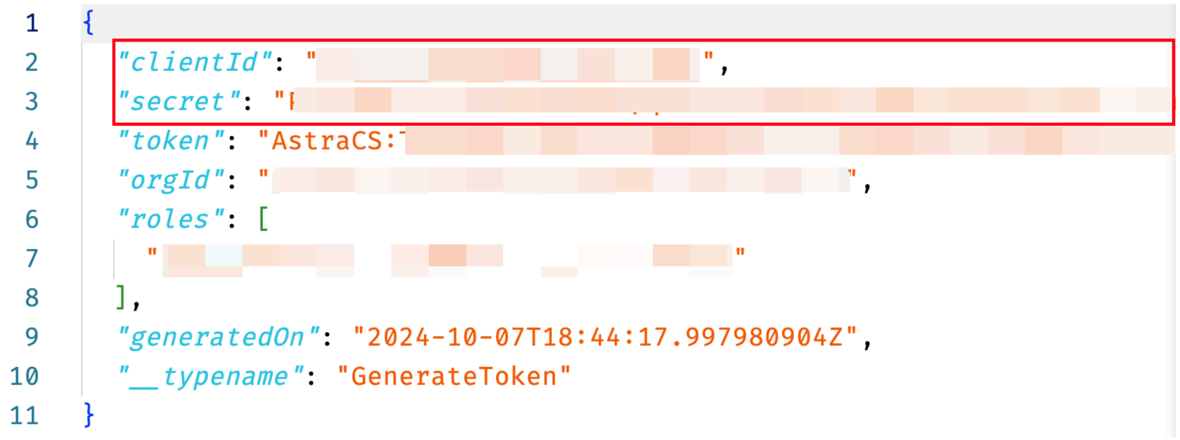
Token Details

Copy or download token details in a secure location before leaving this dialog. While you can always generate a new token, you won't have access to this token later.

Application Token Details

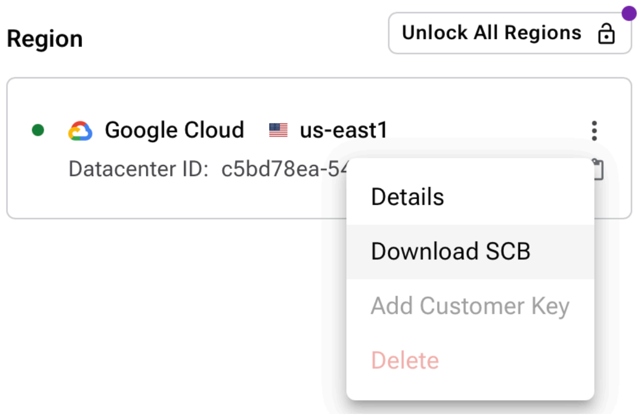
```
{
  "clientId": "[redacted]",
  "secret": "[redacted]
  "token": "[redacted]
}
```

d. In the downloaded file, the first two fields - client ID and client secret – will be used as credentials for the bulk load command.

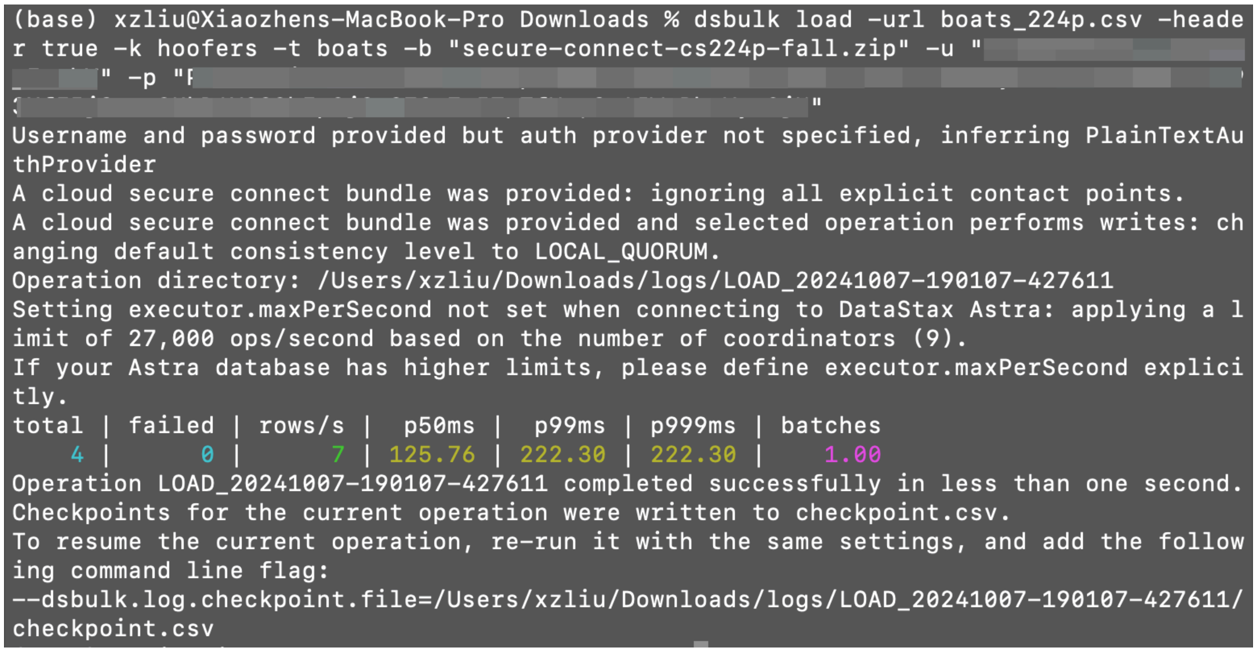


```
{
  "clientId": "[redacted]",
  "secret": "[redacted]
  "token": "AstraCS:[redacted]
  "orgId": "[redacted]",
  "roles": [
    "[redacted]"
  ],
  "generatedOn": "2024-10-07T18:44:17.997980904Z",
  "__typename": "GenerateToken"
}
```

14. Go back to your database’s Dashboard and download the Secure Connect Bundle(SCB) by clicking the “...” in the Region panel - [Secure Connect DataStax Astra Documentation](#).



15. The downloaded result is a compressed .zip file. Safari may automatically uncompress downloaded zip files which ***does not work*** with Astra/DataStax even after manually compressing the download. If that happens, please use another browser for this step (e.g. Firefox or Chrome).
16. [Recommended] Put the secure connect bundle, dsbulk and the boats.csv file in the same folder, and navigate to this location in your terminal window.
17. Bulk load the boats data using the following command:
    ```
    % dsbulk load -url boats.csv -header true -k hoofers -t boats -b "/Users/YOURUSERNAME/folder/path/secure-connect-cs224p-fall.zip" -u CLIENTID -p CLIENTSECRET
    ```
    a. -url: the path to the data file (which, for us, is boats.csv)
    b. -header: specifies that the first row in the data file is a header with information about the attributes
    c. -k: the keyspace name
    d. -t: the table name
    e. -b: the path to the application bundle
    f. -u and -p are the client id and secret values that will be found in the app token JSON file from Step 13.



```
(base) xzliu@Xiaozhens-MacBook-Pro Downloads % dsbulk load -url boats_224p.csv -heade
r true -k hoofers -t boats -b "secure-connect-cs224p-fall.zip" -u "[redacted]
[redacted]" -p "[redacted]
[redacted]"
Username and password provided but auth provider not specified, inferring PlainTextAu
thProvider
A cloud secure connect bundle was provided: ignoring all explicit contact points.
A cloud secure connect bundle was provided and selected operation performs writes: ch
anging default consistency level to LOCAL_QUORUM.
Operation directory: /Users/xzliu/Downloads/logs/LOAD_20241007-190107-427611
Setting executor.maxPerSecond not set when connecting to DataStax Astra: applying a l
imit of 27,000 ops/second based on the number of coordinators (9).
If your Astra database has higher limits, please define executor.maxPerSecond explici
tly.
total | failed | rows/s |  p50ms |  p99ms | p999ms | batches
    4 |      0 |      7 | 125.76 | 222.30 | 222.30 |    1.00
Operation LOAD_20241007-190107-427611 completed successfully in less than one second.
Checkpoints for the current operation were written to checkpoint.csv.
To resume the current operation, re-run it with the same settings, and add the follow
ing command line flag:
--dsbulk.log.checkpoint.file=/Users/xzliu/Downloads/logs/LOAD_20241007-190107-427611/
checkpoint.csv
```

You are now armed and ready to start this assignment! Good luck!