Faculty of Engineering

School of Electrical Engineering and Computer Science

ELG5166 - Cloud Analytics

Group 4 Assignment 1

Personal Ethics & Academic Integrity Statement

Student name: Abdelrhman Rezkallah **Student ID:** 300327290

Student Name: Abdulrahman Ahmed **Student ID:** 300327218

Student Name: Ali El-Sherif **Student ID:** 300327246

Student Name: Basma Abd-Elwahab **Student ID:** 300327209

- **I attest to the fact that my work in this project adheres to the fraud policies as outlined in the Academic Regulations in the University's Graduate Studies Calendar.**
- **I further attest that I have knowledge of and have respected the "Beware of Plagiarism" brochure for the university. To the best of my knowledge, I also believe that each of my group colleagues has also met the aforementioned requirements and regulations.**
- **I understand that if my group assignment is submitted without a completed copy of this Personal Work Statement from each group member, it will be interpreted by the school that the missing student(s) name is confirmation of the non-participation of the aforementioned student(s) in the required work**
- **We, by typing in our names and student IDs on this form and submitting it electronically:**
    - **warrant that the work submitted herein is our own group members' work and not the work of others**
    - **acknowledge that we have read and understood the University Regulations on Academic Misconduct**
    - **acknowledge that it is a breach of University Regulations to give or receive unauthorized and/or unacknowledged assistance on a graded piece of work**

Part 1:

1- Describe briefly what a NoSQL database means. Select a NoSQL database (except MongoDB & Cassandra) and describe how this database can be used for the storage and management of big data.

- The NoSQL database is also called not SQL database, while others called it "not only SQL" database, which is a non-relational database, it stores the data in a format different from relational tables, (**IBM Cloud Education, 2019).**
  - Types of NoSQL database:
    1- Key-value store.
    2- Document-based store.
    3- Column-based store.
    4- Graph-based store.

- **Azure Cosmos DB**: is a fully managed distributed NoSQL database. It is a document-based store. It supports the multi-model database schema which can store the data in key-value pairs, Graph-based, Document-based, and column Family-based databases. It takes advantages of Azure's tools from Microsoft, provides low latency, high availability, and high throughput, (**seesharprun, n.d).**
  - Azure Cosmos DB has multi-master support, it means that the data can be simultaneously written into different databases, which can spread out globally, it offers multiple levels of consistency with varying performance and availability.
    It supports big data storage and management, as it supports ACID transactions, on-demand, and provisioned capacity modes, in transit and at rest. It has big data encryption and access control and can be integrated with Azure Synapse Analytics for real-time no-ETL analytics on operational data. It is best for operations management, gaming, ecommerce, and Internet of Things applications.

2- Investigate and describe one application of Big Data Analytics that was not described in class.

- Big Data Analytics in Education: an important challenge in the education industry is to integrate data from different vendors and sources on different platforms, and the issues of privacy, protection of personal data for educational purposes.
- For example, Big Data used in higher education, such as Tasmania University in Australia, has over than 26,000 students. The University has deployed a learning and management system that tracks when the students log into the University system, the overall progress of each student, and how much time the student spends on the system. **(*Top 10 Big Data Applications across Industries*, n.d.)**.

- Another example is to use Big Data in education to measure the effectiveness of the teacher. The teacher's performance is measured and tuned against the number of students, students' participation, and behaviors to ensure that the education process is pleasant for students and teachers.
- Big Data Analytics helps the Office of Education Technology in the United States choose the correct courses for the students who are going to stray away from courses and detect the boredom of the students while studying courses.

3- Briefly describe the transaction management features of Cassandra and MongoDB in the context of ACID vs. BASE properties.

ACID	BASE
Stands for Atomicity, Consistency, Isolation, Durability.	Stands for Basic-Availability, Soft-State, Eventual-Consistency.

Cassandra	Cassandra organizes the data into partitions, each partition consists of multiple columns that are stored in a node. It provides tuning the consistency levels as per as you need, it provides eventual consistency; you may need to make a request to complete if only one node responds, or you wait until all nodes respond, it supports availability, soft state, so Cassandra supports **BASE** transaction management, (***Introduction | Apache Cassandra Documentation*, n.d).** Cassandra can't allow and serve as a replacement for the traditional Relational Database Management Systems (RDBMS), so it doesn't support **ACID** transaction management that indicates that the database transactions are processed reliably.
MongoDB	MongoDB supports multi-document **ACID** transactions. It allows developers to group the database operations together, so, the transactions will succeed or fail together. The document model of MongoDB enables snapshot isolation, data integrity, and highly distributed shared clusters. Since MongoDB is a consistency, atomicity, and durable, so it supports **ACID** transaction management, **(Anon, 2022).** The **Base** transaction refers to distributing the data through the cluster's nodes instead of being consistent and basically available.

4- You are working on a project that requires you to capture data from millions of IoT devices in people's homes. Each IoT device uploads a JSON document with the data elements required for analytics.

a) Identify potential NoSQL databases that you can capture data from the IoT devices.

- NoSQL databases are most suitable for Internet of Things (IOT) devices, they sense data all the time and need large capacity storage. These devices require flexibility, scalability, high availability, built-in replication, and auto-sharing. So, Redis, Cassandra, MongoDB, Neo4j and Couchbase databases are suitable for IOT applications.
- One of these NoSQL databases is MongoDB, which will be a good selection for IOT applications, as the MongoDB Atlas will help in meeting your business needs regards to the IOT. It provides additional features that will help your IOT architecture and device to be more efficient as it provides high-speed ingestion and real-time analytics.
  Also, there are time-series databases, which are designed to hold time-oriented data and are ideal for audit logs, IoT applications, and machine learning models that use time-series data.

b) What are your design and analytics considerations and rationale behind your choice?

- Because of this medium's versatility, technologies such as MongoDB which stores data as JSON documents and is among the most popular general-purpose options.
- Document databases are frequently write-optimized by default, allowing them to resist the influx of data from IoT devices while compromising robust consistency guarantees.
- IoT use cases are well-suited for maintaining and updating an IoT device's current state since they are often write-heavy and involve unexpected bursts of traffic.
- Furthermore, the fundamental limitation of SQL databases is their static schema, which makes RDBMS unsuitable for IoT applications, no SQL like MongoDB which can expand dynamically according to demand, means you no longer have to take care of infrastructure or handle failover, **(Zharovskikh, 2022).**
- MongoDB projects can be scaled easily and archiving and sharding can be used for the increase of localization and performance of servers and data, nevertheless, high availability. Distributed geographical locations can also provide maximum uptime.

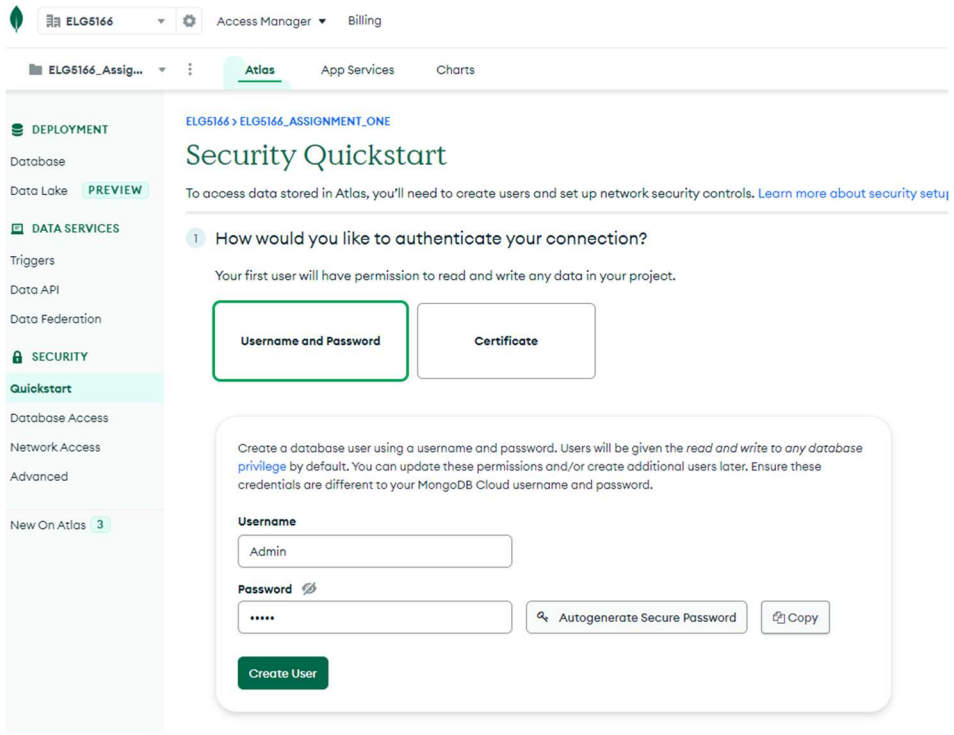
Part 2: NoSQL Labs

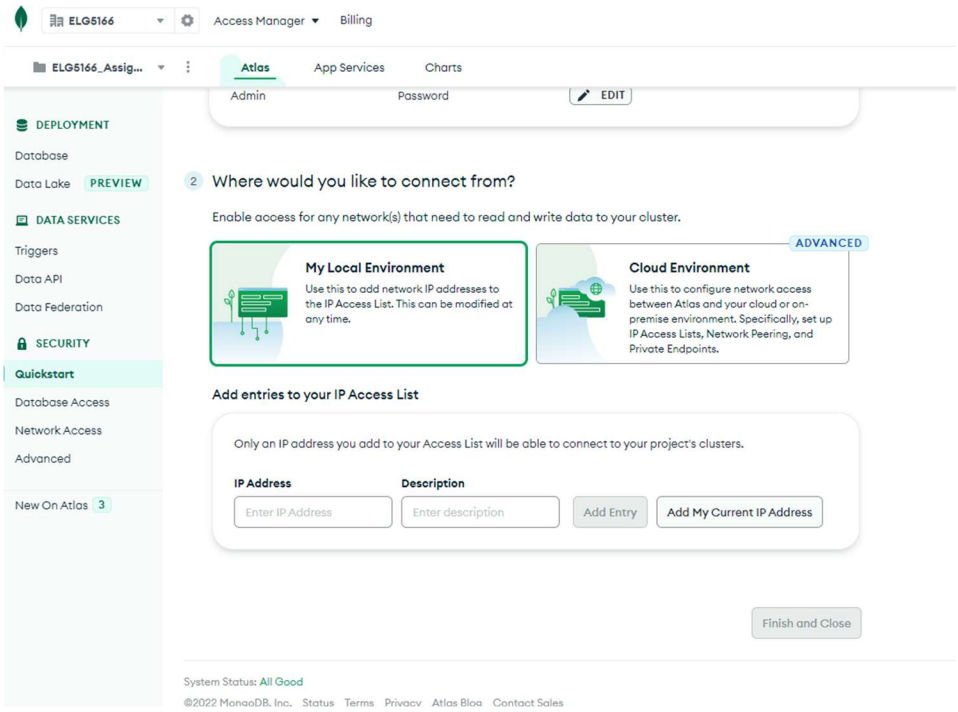
1) MongoDB Lab

1- Setup:

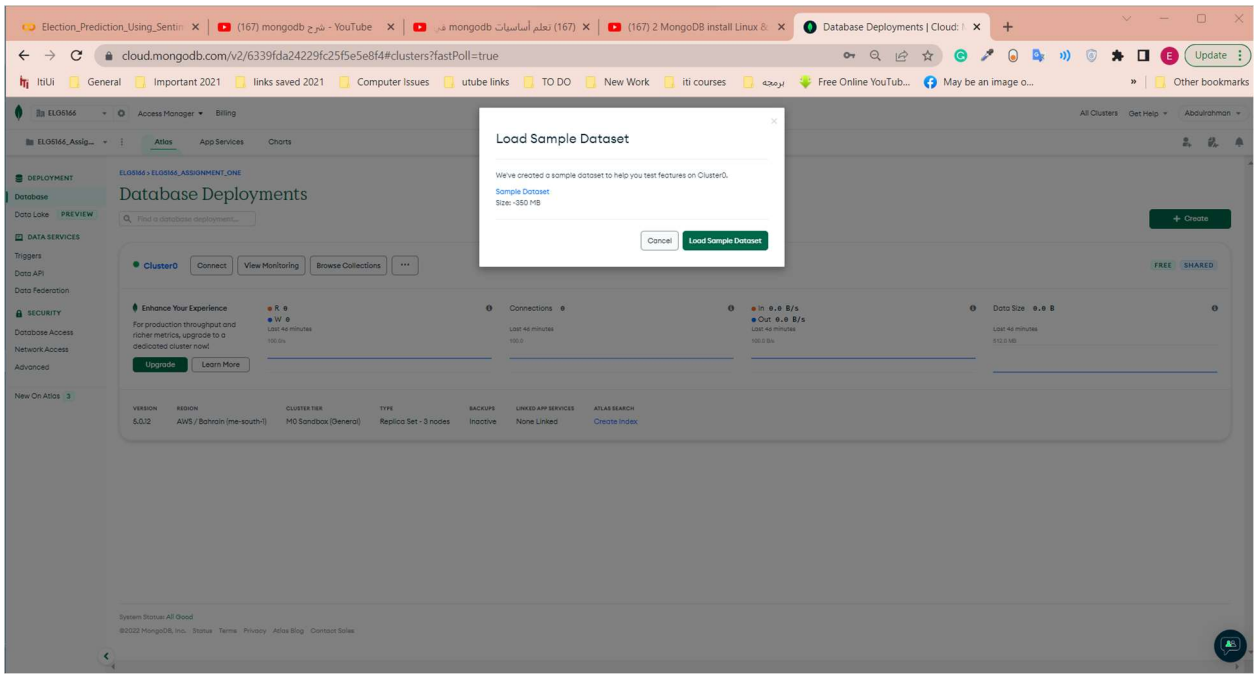
- **Set an account on MongoDB Atlas - https://cloud.mongodb.com**

Creating username and password for MongoDB Atlas

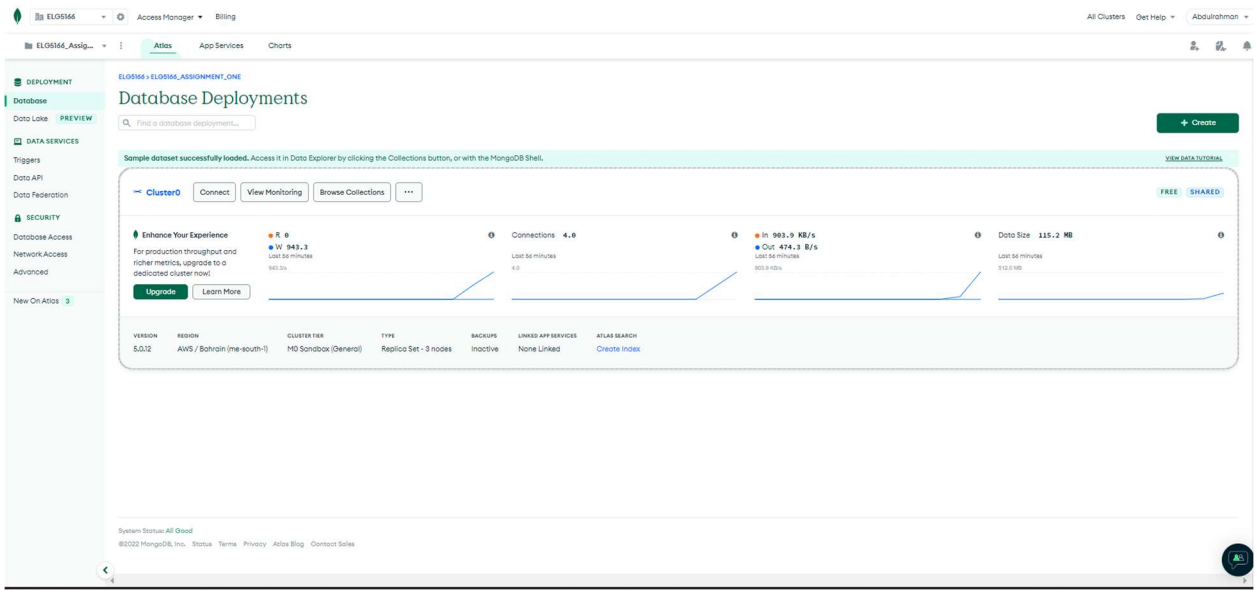




- **Load the Sample Netflix Movies Database to your Data Lake.**

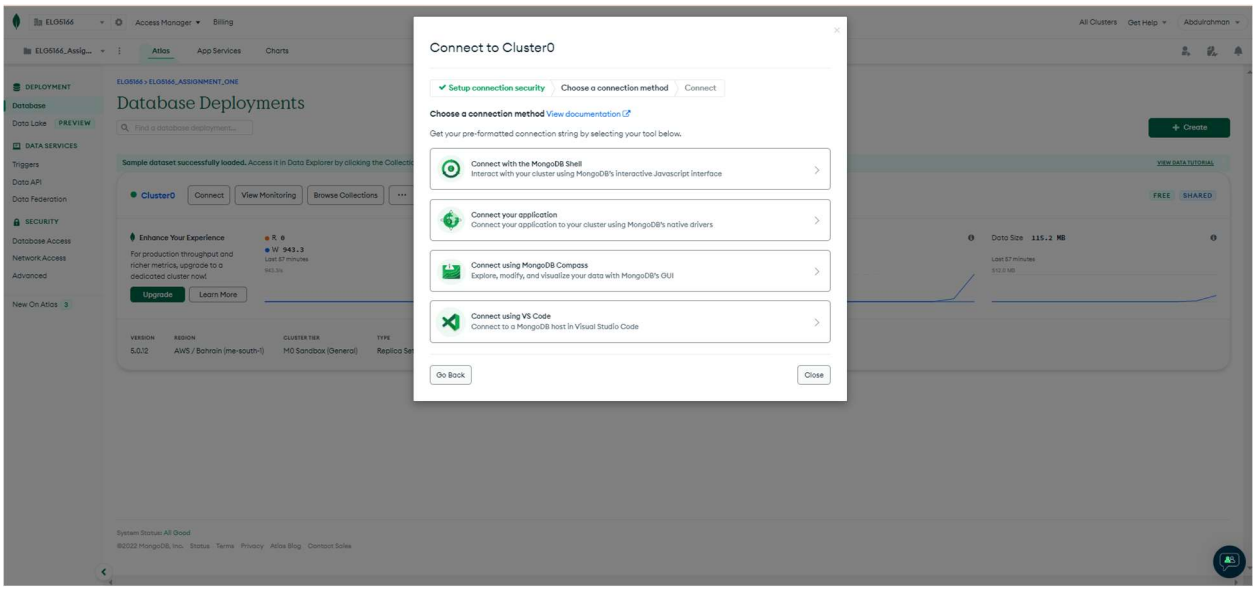


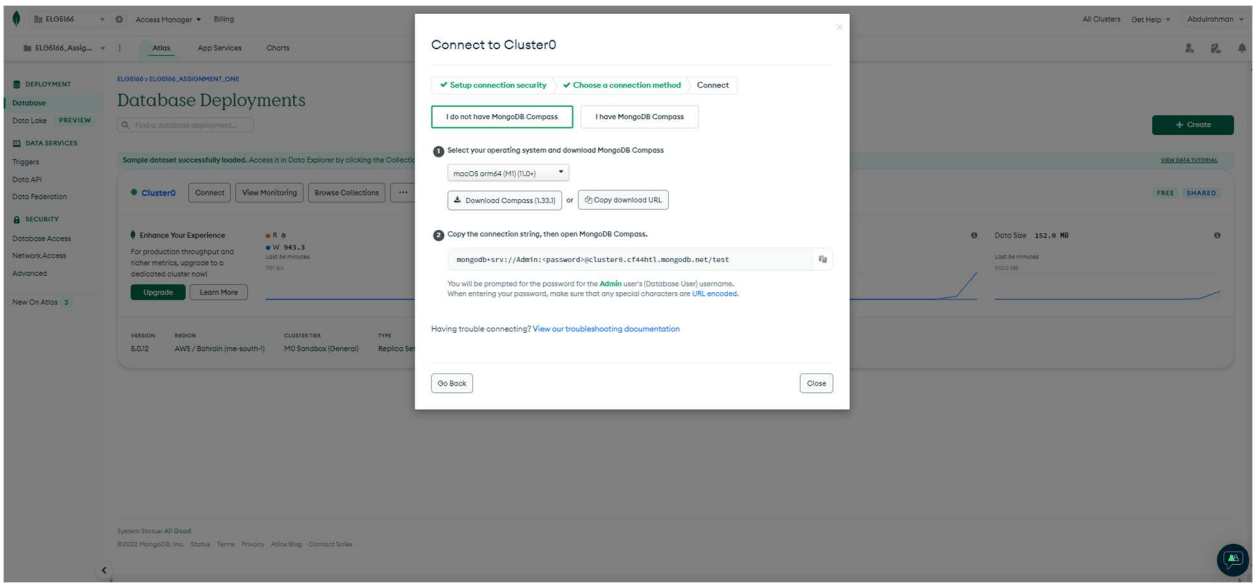
After uploading the sample Netflix database:



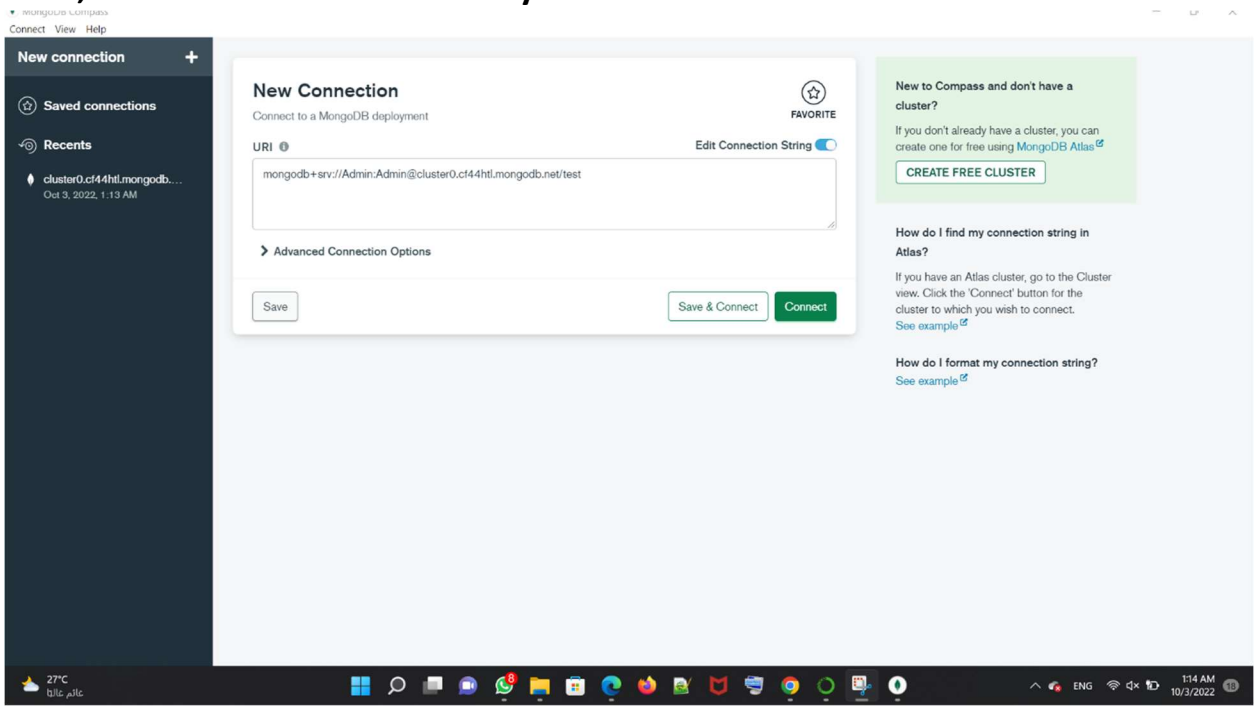
- **Set up a connection to this database instance from MongoDB compass or any other MongoDB client.**

Connecting to the cluster:

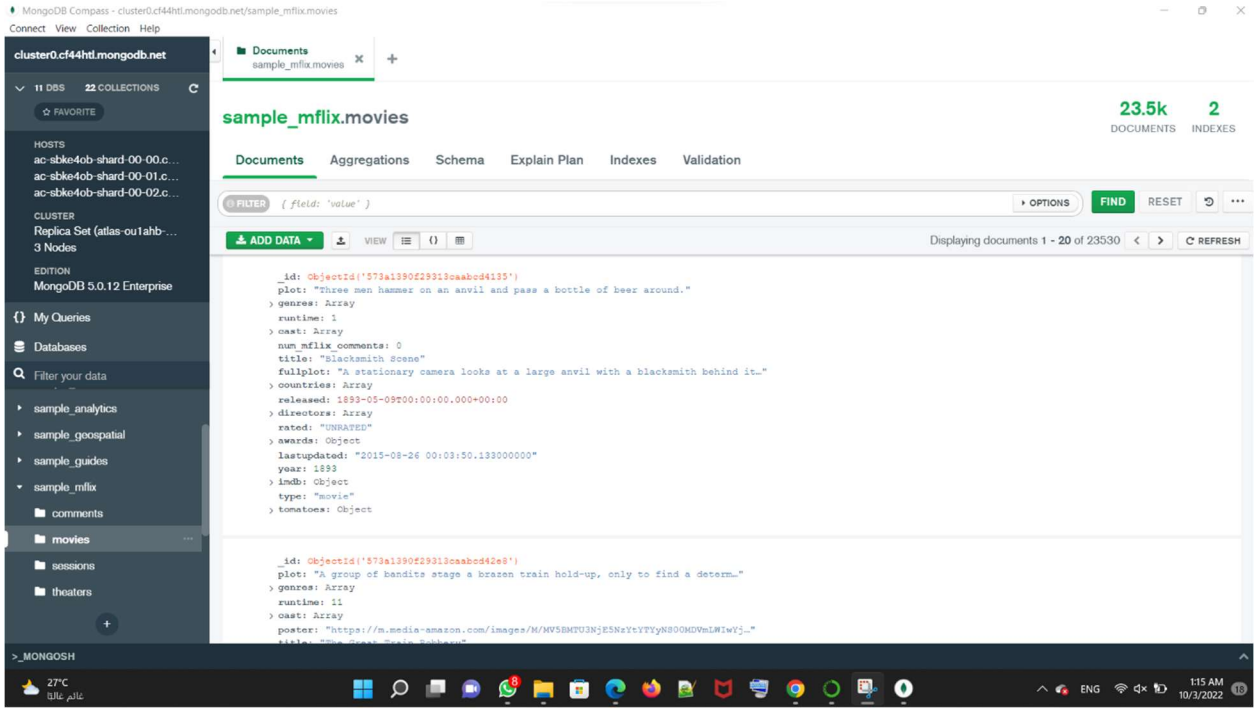




Now, the connection will be ready:



The database is loaded in the MongoDB compass:



Queries)

1- Briefly describe the movies database document model.

- This database collection contains details about movies, each document contains one movie, and information about it, such as name, cast, and release date, poster link, directors, country, simple description, and other information.

Column	Type	Column	Type
Field ID	Object ID	num_mflix comments	Integer number
Awards	Object	Tomatoes	Object
Cast	Array	Plot	String
Countries	Array	Poster	String
Directors	Array	Rated	String
Full plot	String	title	String
Genres	Array	Year	Integer number
Imbd	Object	Runtime	Integer number
Language	Array	Release	Date
Last updated	String	Type	String
metecritic	Integer number	Writers	Array

Awards object	
Column	Type
nimination	Integer number
text	String
Wins	Integer number

imbd object	
Column	Type
Id	Integer number
rating	double
votes	Integer number

Tomatoes object			
Column	Type	Column	Type
Boxoffice	string	lastUpdated	data
Consensus	String	Production	String
Critic	Object	Viewer	Object
Dvd	Data	Rotten	Integer number
fresh	Integer number	website	String

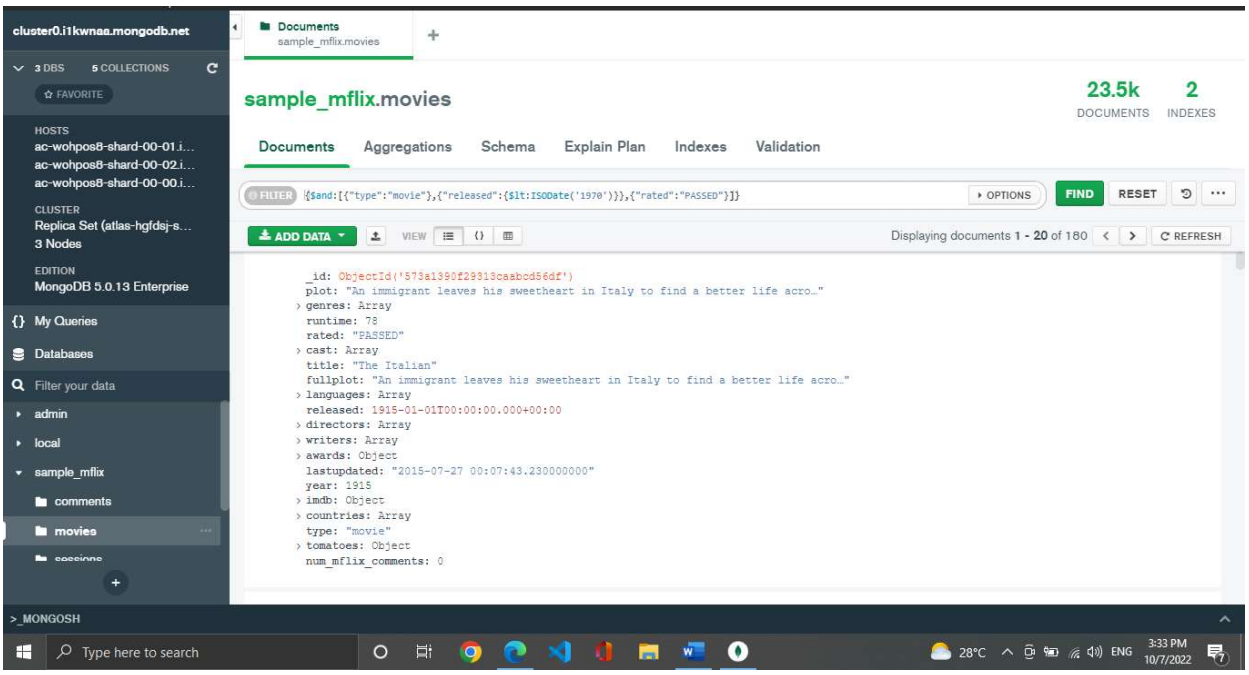
critic object	
Column	Type
Meter	Integer number
Numreviews	Integer number
rating	double

viewer object	
Column	Type
Meter	Integer number
Numreviews	Integer number
rating	double

2- Filter the documents for type “movies” that are released before 1970 and rated as “PASSED”.

Filter

{$and:[{"type":"movie"},{"released":{$lt:ISODate('1970')}},{"rated":"PASSED"}]}



3- Build an Aggregation Pipeline that shows all entries of type movie that have won at least one award and return the release year aggregate counts.

```
Db.movie.aggregrate([
    { $match: {type: "movie", "awards.wins": {$gte:1}}},
    {$group:  {_id:"$year", count_year:{$sum:1}}}
```

Output after $match stage (Sample of 10 documents)

```
_id: ObjectId('573a1390f29313caabcd4323')
plot: "A young boy, opressed by his
      mother, goes on an outing in the
      country …"
genres: Array
runtime: 14
rated: "UNRATED"
cast: Array
```

```
_id: ObjectId('573a1390f29313caa
plot: "A greedy tycoon decides,
      to corner the world market
> genres: Array
runtime: 14
> cast: Array
num_mflix_comments: 1
title: "A Corner in Wheat"
```

$group

```
1 {_id:"$year",
2  count_year:{$sum:1}}
```

Output after $group stage (Sample of 10 documents)

```
_id: 1965
count_year: 81
```

```
_id: 1989
count_year: 169
```

Output after $group stage (Sample of 10 documents)

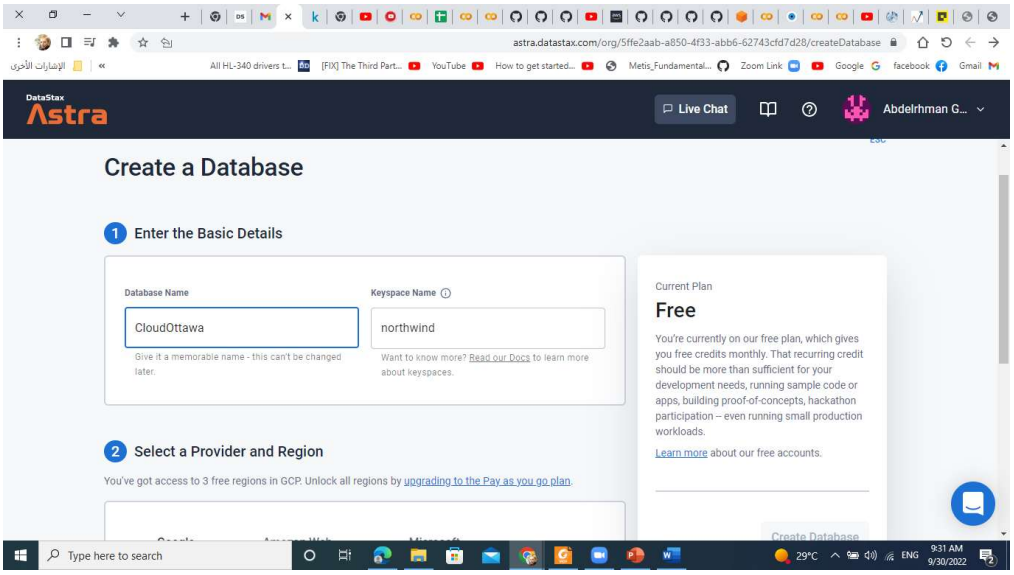
```
_id: 2014
count_year: 784
```

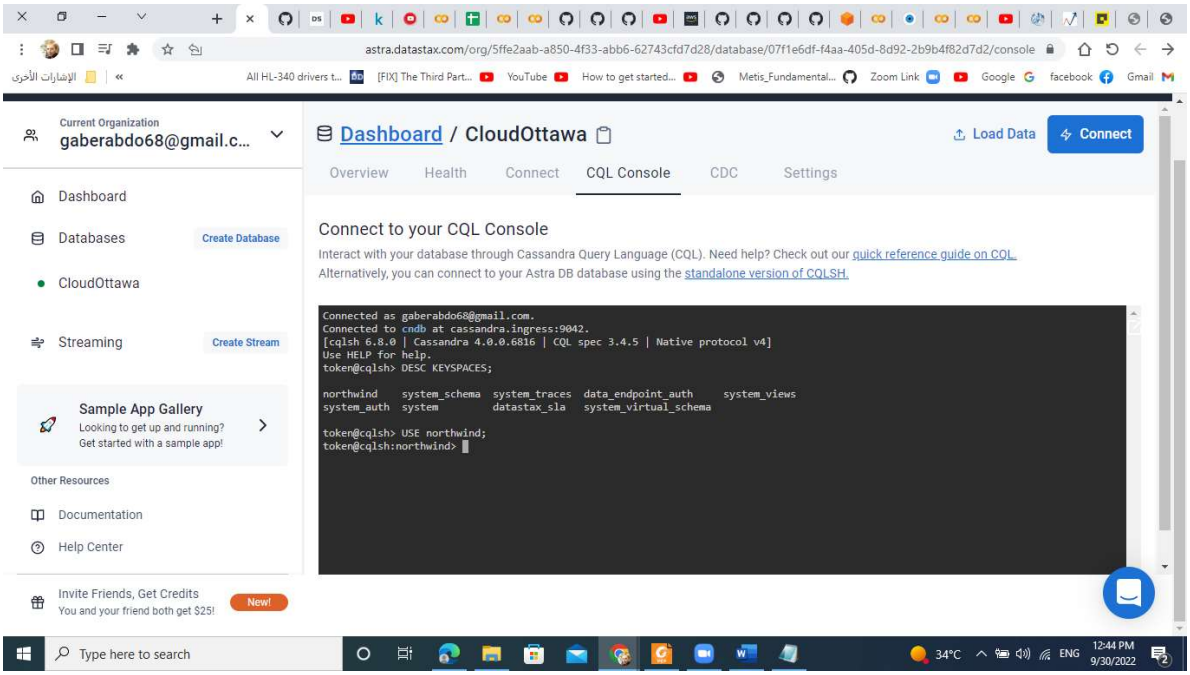
```
_id: 1893
count_year:
```

2) Cassandra Lab:

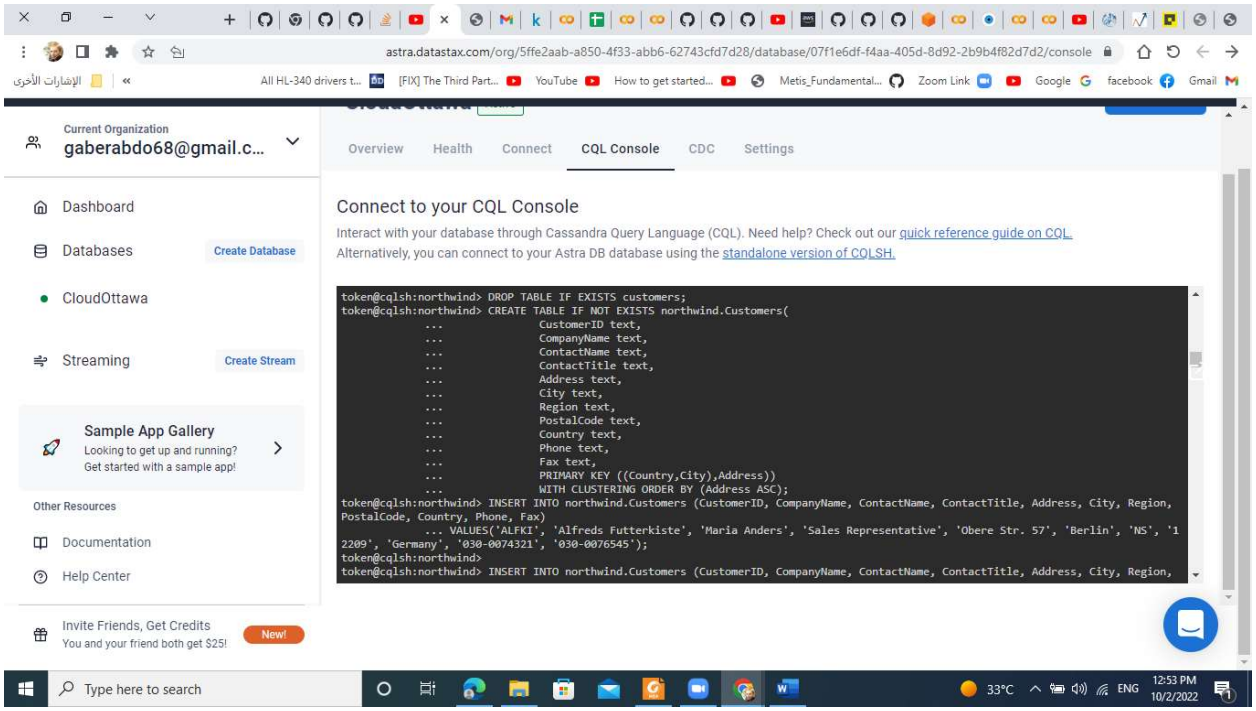
1- Setup:

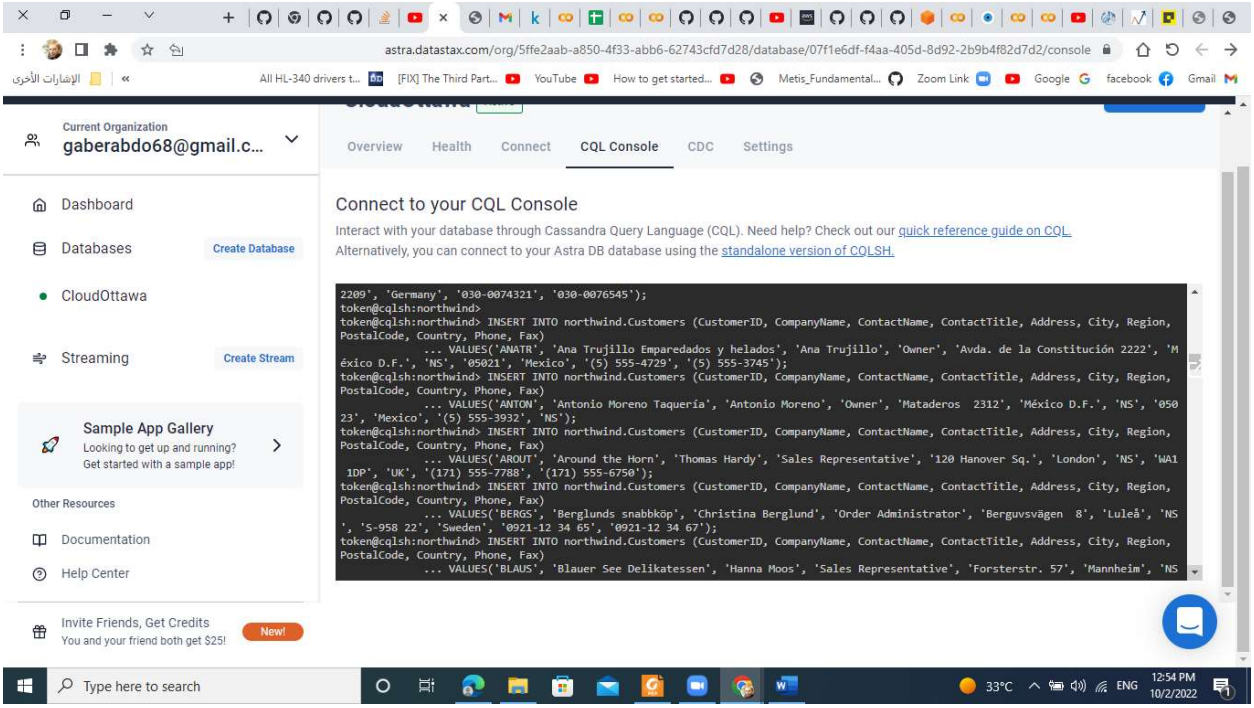
- **Create a Keyspace called northwind:**
  After creating an account on Cassandra and login:



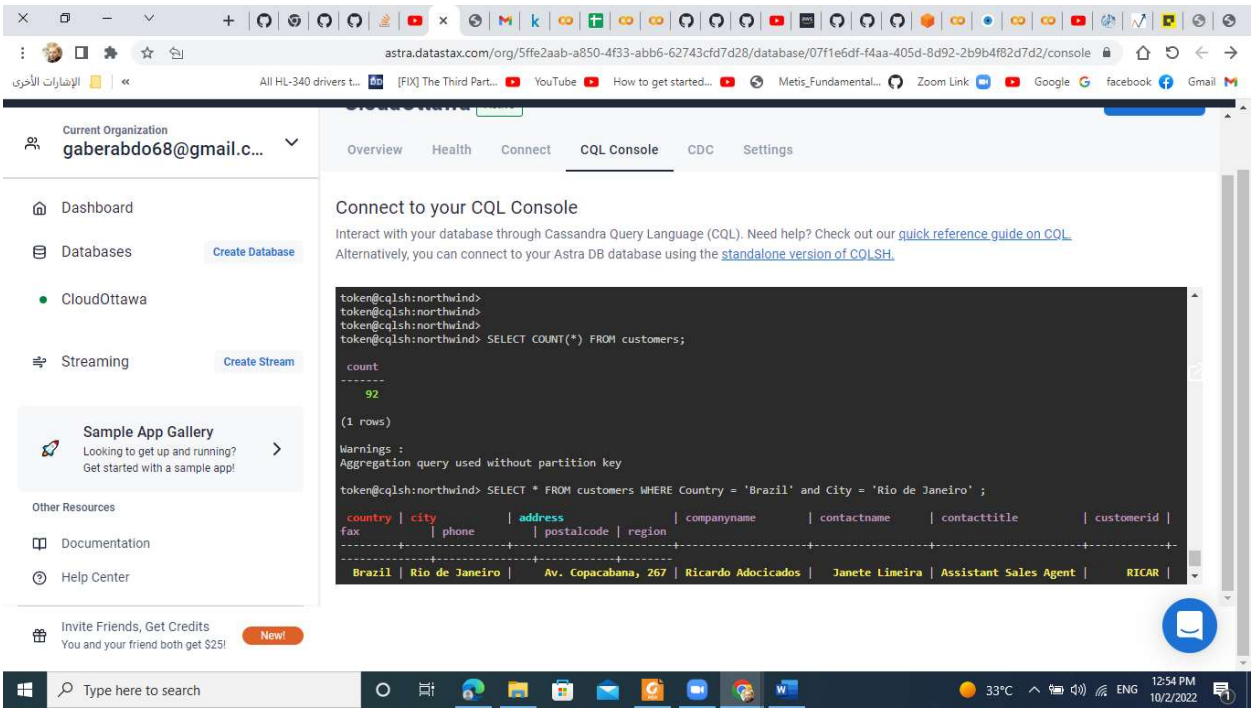


- **Create the customer tables (the attached SQLite definition will serve as a guide) Review the questions in the queries section below and create one or more tables that partition and cluster data so these queries will execute without using Cassandra "ALLOW FILTERING" that scans all partitions.**





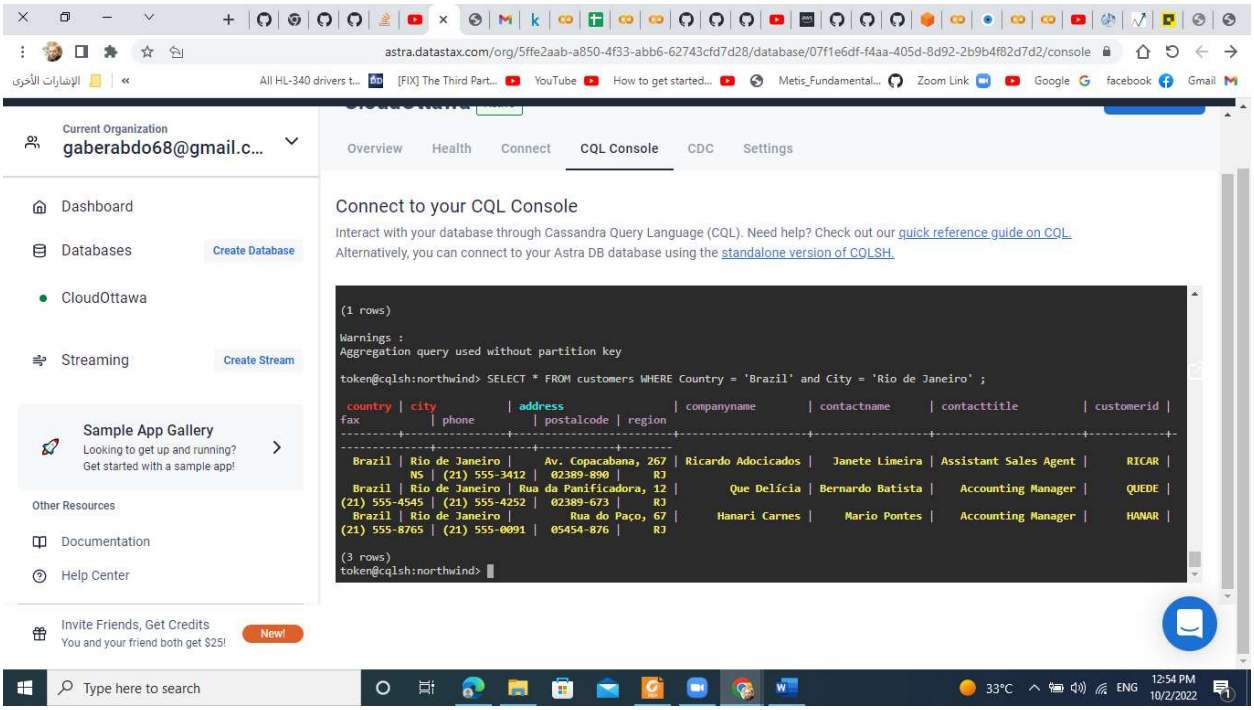
- **Load the attached data into your table(s) using the insert statements (minor modifications may be needed if your definitions include multiple tables). Please include screenshots of table record counts after loading your data.**



Queries)

1. **Provide the query and the results (screenshots and a copy of your query) that show the customers from Rio de Janeiro, Brazil ordered by their addresses.**

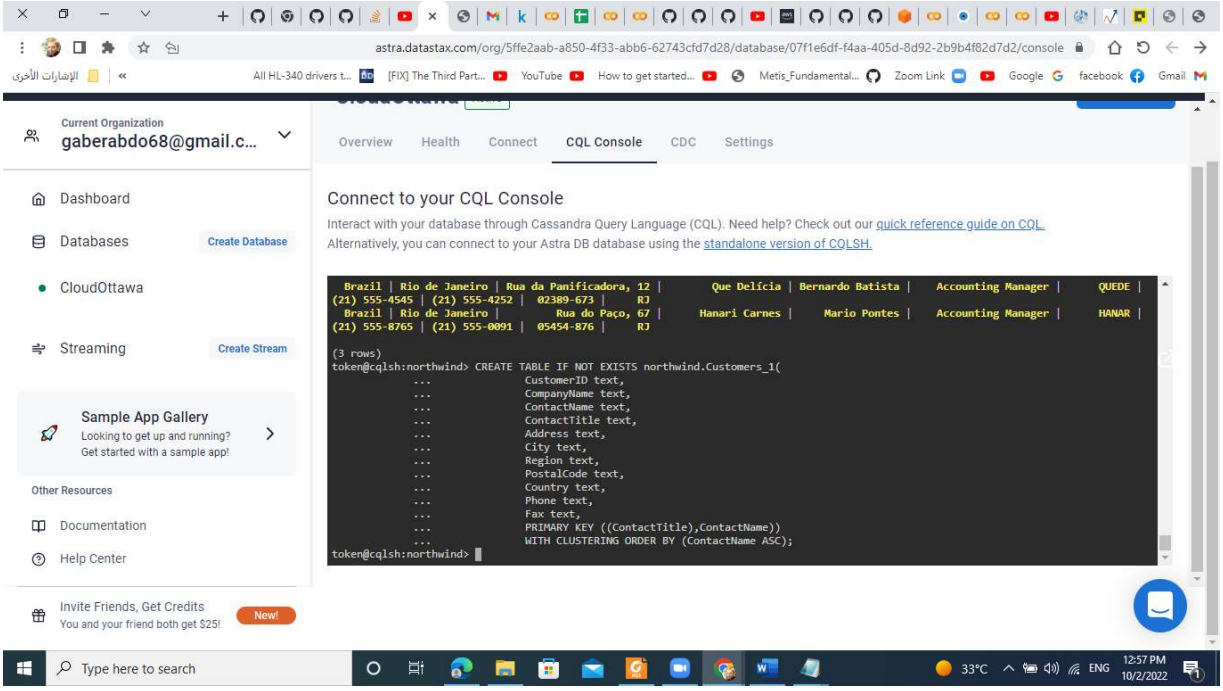
SELECT * FROM customers WHERE Country = 'Brazil' and City = 'Rio de Janeiro';



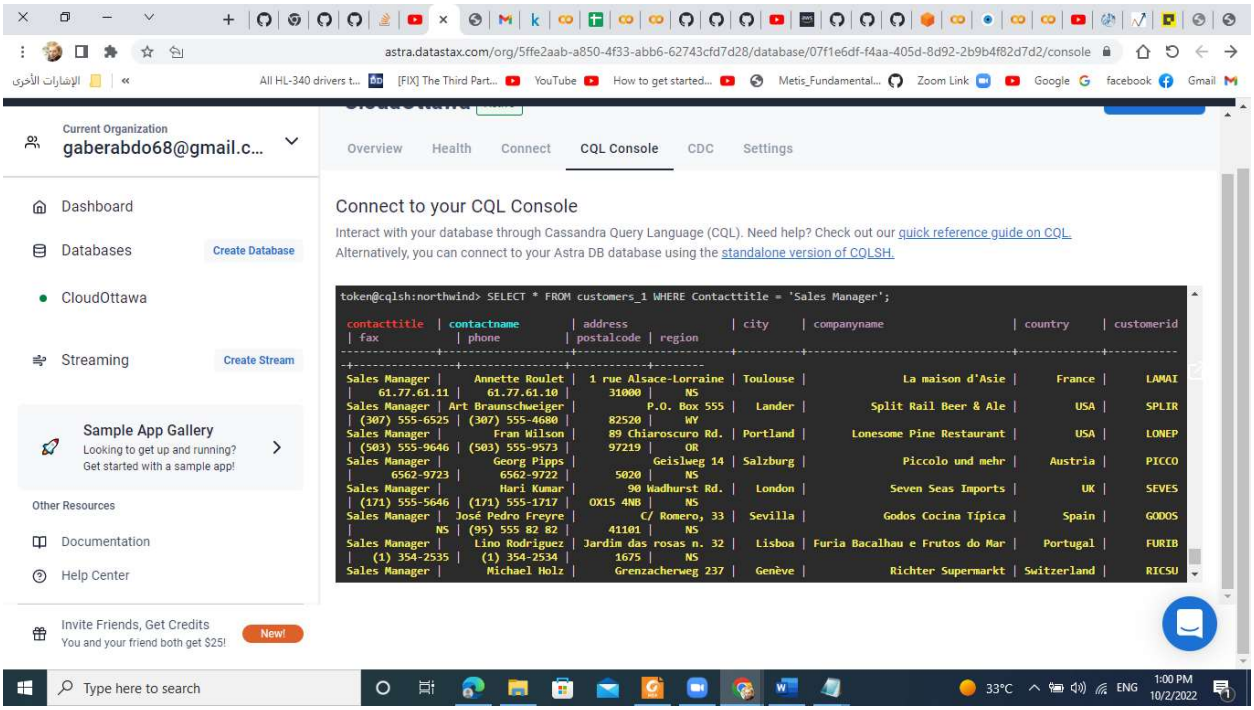
2. **Provide a list of customers that are in the Sales Manager role without forcing the scan of all partitions across all databases. The result should be ordered by their names.**

```
CREATE TABLE IF NOT EXISTS northwind.Customers_1(
        CustomerID text,
        CompanyName text,
        ContactName text,
        ContactTitle text,
        Address text,
        City text,
        Region text,
        PostalCode text,
        Country text,
        Phone text,
```

Fax text,
PRIMARY KEY ((ContactTitle),ContactName))
WITH CLUSTERING ORDER BY (ContactName ASC);



SELECT * FROM customers_1 WHERE Contacttitle = 'Sales Manager';



References:

Anon. (2022). *microsoft - Search*. Www.bing.com. https://www.bing.com/ck/a?

IBM Cloud Education. (2019, August 6). *nosql-databases*. Ibm.com. https://www.ibm.com/cloud/learn/nosql-databases

Introduction | Apache Cassandra Documentation. (n.d.). Cassandra.apache.org. https://cassandra.apache.org/doc/latest/cassandra/data_modeling/intro.html

seesharprun. (n.d.). *Introduction to Azure Cosmos DB*. Learn.microsoft.com. https://learn.microsoft.com/en-us/azure/cosmos-db/introduction

Top 10 Big Data Applications Across Industries. (n.d.). Simplilearn.com. https://www.simplilearn.com/tutorials/big-data-tutorial/big-data-applications

Zharovskikh, A. (2022, March 24). *IoT Big Data: Differences, Similarities & Use Cases*. InData Labs. https://indatalabs.com/blog/iot-big-data