CSCI 3287 Design and Analysis of Data Systems
Homework # 6 -- Cassandra Lab
Fall 2022

## Overview

This Project is worth 100 points (out of 1000) toward your final grade. It is due on Thursday, December 8 at 11:59 p.m. Your assignment submission should be a document saved and submitted as a PDF file via the link found in the assignment section of "Week 15" in Canvas which is the same place where you found this file.

This assignment will give you hands-on practice in working with the Apache Cassandra "NoSQL" database software.

## Objectives

- Students will deploy and run the Apache Cassandra NoSQL database engine
- Students will gain exposure to CQL – Cassandra Query Language
- Students will experience deployment of Apache Cassandra in a Docker container
- Students will launch and use the Cassandra CQL Shell
- Students will run CQL queries

## Submission

Create a document and save it as a PDF file. Submit your document via the link in Canvas in the Week 15 materials for Homework # 6.

Your submission should contain SCREEN SHOTS from your console to verify that you have completed each step. Please number your screen shots to match each step and problem within each step if applicable.

You may work with a partner on this assignment. Each partner must submit their own submission. Be sure to include your partner's name on your submission.

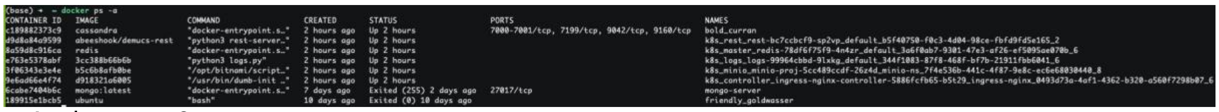
## Instructions to Get Started:

- Ensure you have docker up and running on your systems.
- Install Cassandra via docker on your systems:

```
docker run cassandra
```

- Get the container id of Cassandra from docker

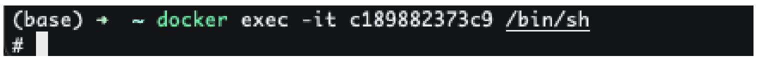
```
docker ps -a
```



```
(base) → ~ docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS                     PORTS                                         NAMES
c189882373c9   cassandra              "docker-entrypoint.s…"   2 hours ago    Up 2 hours                 7000-7001/tcp, 7199/tcp, 9042/tcp, 9160/tcp   bold_curran
d9d8a84a9599   abeeshook/demucs-rest  "python3 rest-server…"   2 hours ago    Up 2 hours                                                               k8s_rest_rest-bc7ccbcf9-sp2vp_default_b5f40750-f0c3-4d04-98ce-fbfd9fd5e165_2
8a59d8c916ca   redis                  "docker-entrypoint.s…"   2 hours ago    Up 2 hours                                                               k8s_master_redis-78df6f75f9-4n4zr_default_3a6f0ab7-9301-47e3-af26-ef5095ae070b_6
e763e5378abf   3cc388b66b6b           "python3 logs.py"        2 hours ago    Up 2 hours                                                               k8s_logs_logs-99964cbbd-9lxkg_default_344f1083-87f8-468f-bf7b-21911fbb6041_6
3f06343e3e4e   b5c6b8afb0be           "/opt/bitnami/script…"   2 hours ago    Up 2 hours                                                               k8s_minio_minio-proj-5cc489ccdf-26z4d_minio-ns_7f4e536b-441c-4f87-9e8c-ec6e68030440_8
9e6ad66e4f74   d918321a6005           "/usr/bin/dumb-init …"   2 hours ago    Up 2 hours                                                               k8s_controller_ingress-nginx-controller-5886fcfb65-b5t29_ingress-nginx_0493d73a-4af1-4362-b320-a560f7298b07_6
6cabe7404b6c   mongo:latest           "docker-entrypoint.s…"   7 days ago     Exited (255) 2 days ago    27017/tcp                                     mongo-server
189915e1bcb5   ubuntu                 "bash"                   10 days ago    Exited (0) 10 days ago                                                   friendly_goldwasser
```

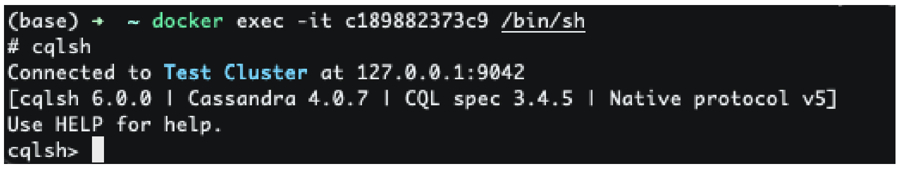
- Using the container id, open sh inside the image

```
docker exec -it <container-id> /bin/sh
```



```
(base) → ~ docker exec -it c189882373c9 /bin/sh
#
```

- Run `cqlsh`



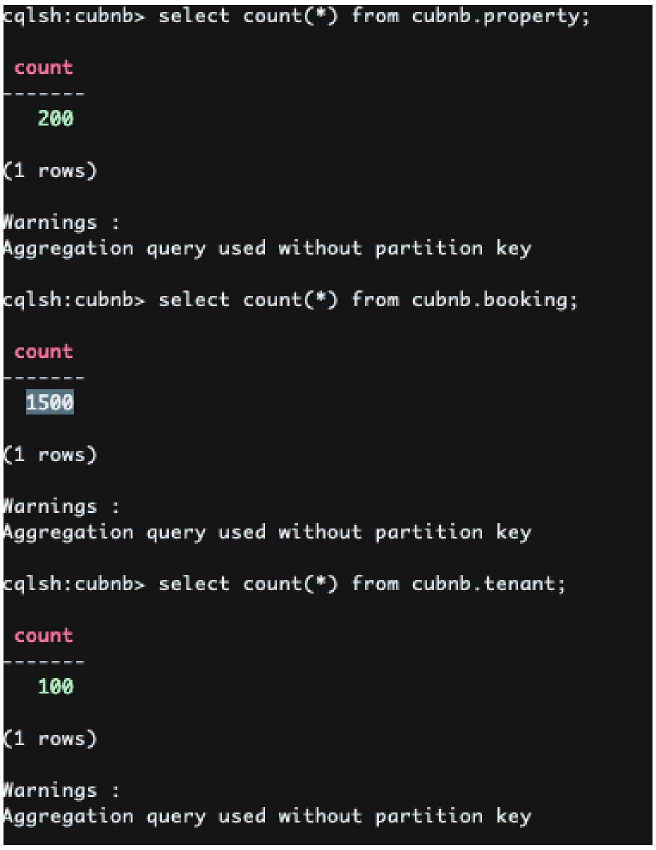
```
(base) → ~ docker exec -it c189882373c9 /bin/sh
# cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.0.0 | Cassandra 4.0.7 | CQL spec 3.4.5 | Native protocol v5]
Use HELP for help.
cqlsh>
```

- We are now connected to the Cassandra image inside docker.

## Using Real-world dataset to get started with the assignment:

1. Download the `cubnb-dataset.txt` from Canvas.
2. Once we have the cassandra image running and are connected to the `sh terminal` inside the image, open the `cubnb-dataset.txt` in a notepad/textpad and copy all the contents (Ctrl + C) and paste the contents in the `sh terminal` (Ctrl + V)

3. Ensure after the completion of all the commands, you get the following output.



```
cqlsh:cubnb> select count(*) from cubnb.property;

 count
-------
   200

(1 rows)

Warnings :
Aggregation query used without partition key

cqlsh:cubnb> select count(*) from cubnb.booking;

 count
-------
  1500

(1 rows)

Warnings :
Aggregation query used without partition key

cqlsh:cubnb> select count(*) from cubnb.tenant;

 count
-------
   100

(1 rows)

Warnings :
Aggregation query used without partition key
```

Note: Incase of an error, please ensure you have spelt the keyspace name and the table names correctly. Keyspace Name: `cubnb`
Table Names: `tenant`, `property`, `booking`

## Assignment Submission Instructions:

- Go through below basic operations of Cassandra before starting with Sections 1 and 2.
- Create operation commands and queries for Section 1 & 2. Please attach the screenshots (of query and output) of both the sections.
- Submit the document as PDF file on Canvas.

  **Note:** You may work with a partner on completing this assignment. However, this is an individual assignment; each one must submit your own final deliverable for this assignment.

### Basic Operations of Cassandra:

1. **Creating a KeySpace:** A keyspace in Cassandra is a namespace that defines data replication on nodes.

```
Syntax:
    CREATE KEYSPACE "KeySpace Name"
    WITH replication = {'class': 'Strategy name', 'replication_factor' : 'Number of replicas'};

Example:
```

```
CREATE KEYSPACE cubnb
WITH replication = {'class':'SimpleStrategy', 'replication_factor' : 3};
```

2. **Using a Keyspace :** You can use a created KeySpace using the keyword **USE**

```
Syntax:
USE <keyspace-name>

Example:
cqlsh> USE cubnb;
```

3. **Creating tables in keyspace :**

```
Syntax:
CREATE (TABLE | COLUMNFAMILY) <tablename>
('<column-definition>' , '<column-definition>') (WITH <option> AND
<option>)

Example:
cqlsh:cubnb>; CREATE TABLE cubnb.emp( emp_id int PRIMARY
     KEY,
     emp_name text, emp_city
     text, emp_sal varint,
     emp_phone varint
     );
```

4. **Describe a KeySpace:** Verify the keyspaces and its tables.

```
Syntax:
desc <keyspace-name>

Example:
cqlsh:cubnb> desc cubnb;
```

5. **Drop a KeySpace:** Delete the keyspaces and its tables.

```
Syntax:
drop keyspace <keyspace-name>

Example:
cqlsh:cubnb> drop keyspace cubnb;
```

## Section 1 :

> Create commands for below operations as mentioned above and attach screenshots in the assignment.

1. **Create** a keyspace **cubnb** for the application having class as SimpleStrategy and replication_factor as 1.

Keyspace name: cubnb.

2. **Use** the keyspace cubnb created above.

3. **Create** a table for **tenant** if not exists. Table name: tenant.
   Columns: tenant_id (int, primary key), mobile (bigint), password (string), tenant_name (string).

4. **Create** a table for **property** if not exists. Table name: property.
   Columns: property_id (int, primary key), property_name (string), host_id (int, primary key), location (string), type (string), price (int), number_of_rooms (int), max_occupancy (int), number_of_reviews (int), amenities (set<string> type), status (string).

5. **Create** a table for **booking** if not exists. Table name: booking.
   Columns: booking_id (int, primary key), property_id (int, primary key), total_occupancy (int), tenant_id (int, primary key), status (string), no_of_days (int), start_date (date), end_date (date).

## Section 2 :

> Create queries for below questions and attach screenshots of the query and output in the assignment.
>
> Note: WE HAVE ALREADY IMPORTED DATASET IN THE ABOVE INSTRUCTIONS SECTION

1. Get the booking_id, property_id, tenant_id and number of days a property is booked in the month of Feb '2022 for tenant-id 80855.
2. Get the count of properties whose reviews are more than 200.
3. List all properties from “Brooklyn” location with type as apartment and have status as available.
4. a) Get all the properties which are handled by host_id 1029.

   b) Add new amenity ‘Heater’ to the above results (in a single query).

   c) Show the updated results from part b query.
5. a) Add your details in tenant table with tenant_id as 3287

   b) Change the password of the tenant_id 3287 from the existing to ‘password@123’ and display the change
6. Remove an amenity ‘GardenView’ for host_id 1012 and property_id 110 and display the results.
7. List the properties which have a ‘BeachView’ in its amenities and whose rent price is more than 1000.
8. Get the number of highest occupancy of properties for 8 rooms.