AS201

Astra DB for Cloud Native Applications

Lab Guide

Lab 01: Create an Astra DB Account

1. Connect to https://astra.datastax.com
2. Create a free account (use any email you choose)
3. Create a database called *AS201*
4. Create a keyspace called *class*
5. Choose an available region
6. Copy or download the *secure token* and save it for use later
7. Explore your new database and the Astra UI

END OF LAB

Lab 02: Create a Table

1. Connect to the Astra Dashboard
2. Access your database
3. Open the CQL Console
4. Use the keyspace (class)
5. Create a table called *cars* with the following fields

Field	Type	Key
id	int	primary key
make	text	
model	text	
year	int	

6. Insert some data

```
INSERT INTO cars(id, make, model, year)
  values(1001, 'Dodge', 'Challenger', 1971);
INSERT INTO cars(id, make, model, year)
  values(1002, 'Ford', 'Mustang', 1968);
INSERT INTO cars(id, make, model, year)
  values(1003, 'Chevy', 'Camaro', 1969);
INSERT INTO cars(id, make, model, year)
  values(1004, 'Dodge', 'Daytona', 1969);
INSERT INTO cars(id, make, model, year)
  values(1005, 'Dodge', 'Challenger', 1972);
INSERT INTO cars(id, make, model, year)
  values(1006, 'Ford', 'Mustang', 1971);
INSERT INTO cars(id, make, model, year)
  values(1007, 'Dodge', 'Charger', 1969);
```

7. Run a query and select all cars

END OF LAB

Lab 03: INSERTs, UPSERTs and UPDATEs

1. In this lab make sure your CQL commands only contain the necessary information to complete the task
2. In the *cars* table from the previous lab, use an INSERT to change car, id 1007, form a Dodge Charger to a Dodge Dart
3. Verify that the values has changed
4. Use an UPDATE to change it back
5. Verify that the value has changed back

END OF LAB

LAB 04: Partition Keys

1. Use the *DROP TABLE* command in CQL Shell to delete the cars table
2. Recreate the cars table with:

   a composite partition key: *make & model*
   a clustering column: *id*

3. Populate the table with the same data as before
4. Run a query to retrieve the Camaro form the table
5. Write a query to retrieve all the Dodges from the table

END OF LAB

Lab 05: Clustering Columns

1. Use the table from the previous lab for this exercise because it already has clustering column
2. Write a query to return all Dodge Challengers and all Ford Mustangs with ids less than 1005
3. Write a query to return all Chevy Camaros
4. Write a query to return all cars in descending id order

END OF LAB

Lab 06: Multiple Clustering Columns

1. Use the *DROP TABLE* command in CQL Shell to delete the cars table
2. Create a new cars table

Field	Type	Key
make	text	partition key
model	text	partition key
miles	int	1st clustering column
year	int	2nd clustering column
color	text	

3. Insert the following data into the table

```
INSERT INTO cars(make, model, miles, year, color)
    values('Ford', 'Mustang', 34000, 1969, 'red');
INSERT INTO cars(make, model, miles, year, color)
    values('Ford', 'Mustang', 40000, 1969, 'green');
INSERT INTO cars(make, model, miles, year, color)
    values('Ford', 'Mustang', 45000, 1968, 'blue');
INSERT INTO cars(make, model, miles, year, color)
    values('Chevy', 'Camaro', 13000, 1969, 'red');
INSERT INTO cars(make, model, miles, year, color)
    values('Chevy', 'Camaro', 31000, 1969, 'yellow');
INSERT INTO cars(make, model, miles, year, color)
    values('Chevy', 'Camaro', 31000, 1971, 'red');
INSERT INTO cars(make, model, miles, year, color)
    values('Chevy', 'Camaro', 60000, 1970, 'blue');
```

4. Write a query to list all Ford Mustangs in order by miles (descending)
5. Write a query to list all Ford Mustangs in order by year (ascending)
6. Write a query to list all Ford Mustangs in order by miles (descending)
7. Write a query to list all Chevy Camaros with more than 15000 miles
8. Write a query to list all Chevy Camaros

END OF LAB

Lab 07: Consistency Level

9. Using the cars table from the previous lab
10. Execute a query to list all cars at Consistency Level ALL
11. Execute a query to list all cars at Consistency Level LOCAL_QUORUM
12. Execute a query to list all cars at Consistency Level THREE
13. Execute a query to list all cars at Consistency Level TWO
14. Execute a query to list all cars at Consistency Level ONE
15. Insert a new car (Chevy ,Camaro, 4000 miles, 2022, yellow) at Consistency Level ALL
16. Insert the same car at Consistency Level LOCAL_QUORUM
17. Insert the same car at Consistency Level THREE
18. Insert the same car at Consistency Level TWO
19. Insert the same car at Consistency Level ONE

END OF LAB

Lab 08: Denormalization

20. Using the data from Lab 06, create a new table that allows the following queries (do not delete the cars table)

    all cars from a specific year

    all cars from 1969 with miles between 25000 and 35000

END OF LAB

Lab 9: Storage Attached Indexes

1. Use the cars table from the *Multiple Clustering Columns* lab
2. Create an index that allows querying all cars from a given year
3. Create another index that allows querying all yellow and red cars
4. If you needed to query all *red* cars from *1969* would the best solution be an SAI index or a denormalized table?