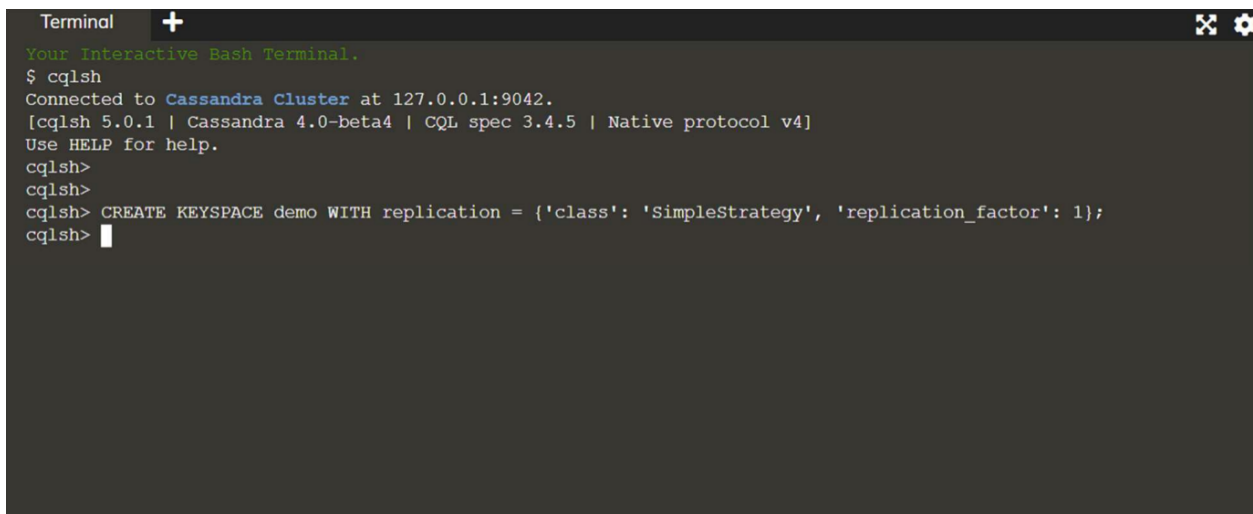


Practical - 9: Setup Cassandra environment in your system and perform CRUD operation.

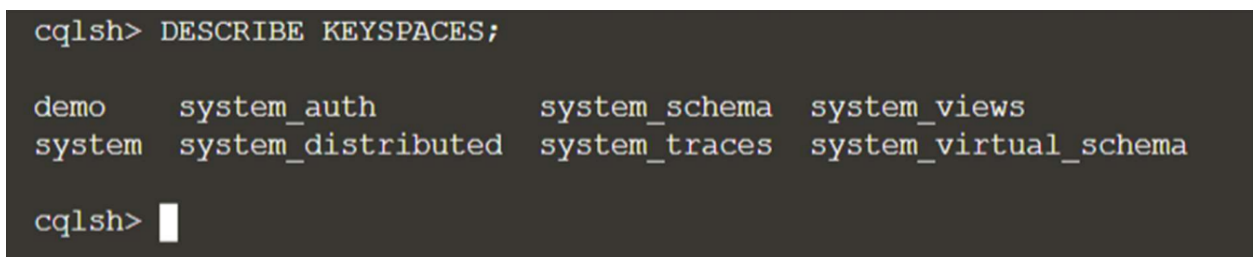
By Divya Mahur (18bce106)

→ **Create Keyspace:** Keyspace is similar to Database in mysql.

A terminal window titled 'Terminal' with a '+' icon and window controls. It shows the command 'cqlsh' being executed, connecting to a Cassandra cluster at 127.0.0.1:9042. The terminal displays the version information: '[cqlsh 5.0.1 | Cassandra 4.0-beta4 | CQL spec 3.4.5 | Native protocol v4]'. The user is prompted to use HELP for help. Then, the user enters the command 'CREATE KEYSPACE demo WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};' and the prompt returns to 'cqlsh>'.

```
Terminal +
Your Interactive Bash Terminal.
$ cqlsh
Connected to Cassandra Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 4.0-beta4 | CQL spec 3.4.5 | Native protocol v4]
Use HELP for help.
cqlsh>
cqlsh>
cqlsh> CREATE KEYSPACE demo WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh> 
```

→ **Describe Keyspace:** From this we can view all the keyspaces which have been created in Cassandra

A terminal window showing the command 'DESCRIBE KEYSPACES;' being executed in cqlsh. The output lists the keyspaces: 'demo', 'system_auth', 'system_schema', 'system_views', 'system', 'system_distributed', 'system_traces', and 'system_virtual_schema'. The prompt returns to 'cqlsh>'.

```
cqlsh> DESCRIBE KEYSPACES;

demo      system_auth      system_schema    system_views
system    system_distributed system_traces     system_virtual_schema

cqlsh> 
```

-> Create Table: Creating table called users and declaring "lastname" as PRIMARY KEY.

```
cqlsh> CREATE TABLE demo.users (lastname text PRIMARY KEY, firstname text, email text);  
CREATE TABLE demo.users (lastname text PRIMARY KEY, firstname text, email text);
```

-> Insert User: First selecting the keyspace, inserting the column and column values.

```
cqlsh> USE demo;  
cqlsh:demo> INSERT INTO users (lastname,firstname,email) VALUES ('Vakil', 'Jayraj','jayraj@gmail.com');  
cqlsh:demo> INSERT INTO users (lastname, firstname, email) VALUES ('Pratico', 'Cassi', 'cassi@example.com');  
cqlsh:demo> INSERT INTO users (lastname, firstname, email) VALUES ('Polson', 'Lino', 'lino@example.com');  
cqlsh:demo> □
```

-> Select User:

```
cqlsh:demo> SELECT * FROM users;
```

lastname	email	firstname
Polson	lino@example.com	Lino
Pratico	cassi@example.com	Cassi
Vakil	jayraj@gmail.com	Jayraj

(3 rows)

```
cqlsh:demo> □
```

-> Updating User:

```
cqlsh:demo> SELECT * FROM users;
```

lastname	email	firstname
Polson	lino@example.com	Lino
Pratico	cassi@example.com	Cassi
Vakil	jayraj@gmail.com	Jayraj

(3 rows)

```
cqlsh:demo> SELECT * FROM users WHERE lastname = 'Vakil';
```

lastname	email	firstname
Vakil	jayraj@gmail.com	Jayraj

(1 rows)

```
cqlsh:demo> UPDATE users SET email = 'jvakil@gmail.com' WHERE lastname = 'Vakil';  
cqlsh:demo> SELECT * FROM users WHERE lastname = 'Vakil';
```

lastname	email	firstname
Vakil	jvakil@gmail.com	Jayraj

(1 rows)

```
cqlsh:demo>
```

-> Deleting User:

```
cqlsh:demo> DELETE FROM users WHERE lastname = 'Vakil';  
cqlsh:demo> SELECT * FROM users;
```

lastname	email	firstname
Polson	lino@example.com	Lino
Pratico	cassi@example.com	Cassi

(2 rows)

```
cqlsh:demo>
```