Міністерство освіти і науки, молоді та спорту України
Національний технічний університет України
“Київський політехнічний інститут”
Фізико-Технічний інститут

Лабораторна робота № 7
з семестрового курсу
“Проектування високонавантажених систем”

Виконала:
Студентка групи ФІ-03
Швець Катерина

Київ, 2023

Task 7 - Реплікация у Cassandra

1. Сконфігурувати кластер з 3-х нод:

- https://hub.docker.com/_/cassandra
- https://gokhanatil.com/2018/02/build-a-cassandra-cluster-on-docker.html
- https://www.jamescoyle.net/how-to/2448-create-a-simple-cassandra-cluster-with-3-nodes
- https://www.digitalocean.com/community/tutorials/how-to-run-a-multi-node-cluster-database-with-cassandra-on-ubuntu-14-04

```
┌──(kali㉿kali)-[~]
└─$ docker run --name cassandra-node1 -d cassandra:latest
bbf02c6f5b1a372d95da51efb26e90f66e7ce5f3deee0f57c60e1a7909d0f1a6

┌──(kali㉿kali)-[~]
└─$ docker run --name cassandra-node2 -d --link cassandra-node1:cassandra cassandra:latest
c6e8a19d71a9e34b53d0c49b756f273497dcca274eb800b298ed399bccd5309e

┌──(kali㉿kali)-[~]
└─$ docker run --name cassandra-node3 -d --link cassandra-node1:cassandra cassandra:latest
d0e6c7d245ea8d4469d6130e0c6808d8c8d41e8bca9271ed1a4c47a6cb60d999
```

2. Перевірити правильність конфігурації за допомогою *nodetool status*

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool status
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address     Load       Tokens  Owns (effective)  Host ID                               Rack
UN  172.17.0.4  109.32 KiB  16      59.3%             fd7c433c-1595-403c-8a2c-3b237c9bbe79  rack1
UN  172.17.0.3  109.39 KiB  16      64.7%             921a5362-771e-4f38-a8d5-76d52b43a522  rack1
UN  172.17.0.2  180.74 KiB  16      76.0%             085334f5-545b-4d9c-a8be-d40084b6e877  rack1
```

3. Використовуючи *cqlsh*, створити три *Keyspace* з replication factor 1, 2, 3

https://www.tutorialspoint.com/cassandra/cassandra_create_keyspace.htm
https://docs.datastax.com/en/cql/3.1/cql/cql_reference/create_keyspace_r.html

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> CREATE KEYSPACE keyspace_rf1 WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh> CREATE KEYSPACE keyspace_rf2 WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 2};
cqlsh> CREATE KEYSPACE keyspace_rf3 WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 3};
```

4. В кожному з кейспейсів створити таблиці

https://docs.datastax.com/en/cql/3.1/cql/cql_reference/create_table_r.html
https://www.tutorialspoint.com/cassandra/cassandra_create_table.htm

```
cqlsh> USE keyspace_rf1;
cqlsh:keyspace_rf1> CREATE TABLE example_table (id int PRIMARY KEY, data text);
cqlsh:keyspace_rf1> USE keyspace_rf2;
cqlsh:keyspace_rf2> CREATE TABLE example_table (id int PRIMARY KEY, data text);
cqlsh:keyspace_rf2> USE keyspace_rf3;
cqlsh:keyspace_rf3> CREATE TABLE example_table (id int PRIMARY KEY, data text);
```

5. Спробуйте писати і читати на / та з різних нод.

Заповнення першої таблиці:

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> INSERT INTO keyspace_rf1.example_table (id, data) VALUES (1, 'test1');
cqlsh> INSERT INTO keyspace_rf1.example_table (id, data) VALUES (2, 'test2');
cqlsh> SELECT * FROM keyspace_rf1.example_table;

 id | data
----+-------
  1 | test1
  2 | test2
```

перевірка з іншиї нод

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node2 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf1.example_table;

 id | data
----+-------
  1 | test1
  2 | test2

(2 rows)
cqlsh> exit

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node3 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf1.example_table;

 id | data
----+-------
  1 | test1
  2 | test2
```

Заповнення другої таблиці і тут ми з 3 ноди записуємо в 2 таблицю

```
cqlsh> INSERT INTO keyspace_rf2.example_table (id, data) VALUES (1, '123');
cqlsh> INSERT INTO keyspace_rf2.example_table (id, data) VALUES (2, '456');
cqlsh> SELECT * FROM keyspace_rf2.example_table;

 id | data
----+------
  1 |  123
  2 |  456

(2 rows)
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf2.example_table;

 id | data
----+------
  1 |  123
  2 |  456

(2 rows)
cqlsh>

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node3 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf2.example_table;

 id | data
----+------
  1 |  123
  2 |  456

(2 rows)
cqlsh>
```

Заповнення третьої таблиці:

```
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, '555');
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (2, '777');
cqlsh> SELECT * FROM keyspace_rf3.example_table;

 id | data
----+------
  1 |  555
  2 |  777

(2 rows)
```

Перевірка з інших нод

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf3.example_table;

 id | data
----+------
  1 |  555
  2 |  777

(2 rows)
cqlsh>

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node2 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf3.example_table;

 id | data
----+------
  1 |  555
  2 |  777
```

6. Вставте дані в створені таблиці і подивіться на їх розподіл по вузлах кластера (для кожного з кейспесов - *nodetool status*)

https://docs.datastax.com/en/cql/3.1/cql/cql_reference/insert_r.html
https://docs.datastax.com/en/cql/3.1/cql/cql_reference/select_r.html
https://www.tutorialspoint.com/cassandra/cassandra_create_data.htm
https://www.tutorialspoint.com/cassandra/cassandra_read_data.htm

Вставила дані в таблицю в минулому пункті

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool status keyspace_rf3
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address     Load        Tokens  Owns (effective)  Host ID                               Rack
UN  172.17.0.4  115.44 KiB  16      100.0%            fd7c433c-1595-403c-8a2c-3b237c9bbe79  rack1
UN  172.17.0.3  110.46 KiB  16      100.0%            921a5362-771e-4f38-a8d5-76d52b43a522  rack1
UN  172.17.0.2  138.22 KiB  16      100.0%            085334f5-545b-4d9c-a8be-d40084b6e877  rack1


┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool status keyspace_rf2
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address     Load        Tokens  Owns (effective)  Host ID                               Rack
UN  172.17.0.4  115.44 KiB  16      59.3%             fd7c433c-1595-403c-8a2c-3b237c9bbe79  rack1
UN  172.17.0.3  110.46 KiB  16      64.7%             921a5362-771e-4f38-a8d5-76d52b43a522  rack1
UN  172.17.0.2  138.22 KiB  16      76.0%             085334f5-545b-4d9c-a8be-d40084b6e877  rack1


┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool status keyspace_rf1
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address     Load        Tokens  Owns (effective)  Host ID                               Rack
UN  172.17.0.4  115.44 KiB  16      31.6%             fd7c433c-1595-403c-8a2c-3b237c9bbe79  rack1
UN  172.17.0.3  110.46 KiB  16      32.7%             921a5362-771e-4f38-a8d5-76d52b43a522  rack1
UN  172.17.0.2  138.22 KiB  16      35.7%             085334f5-545b-4d9c-a8be-d40084b6e877  rack1
```

7. Для якогось запису з кожного з кейспейсу виведіть ноди на яких зберігаються дані

https://docs.datastax.com/en/dse/5.1/dse-admin/datastax_enterprise/tools/nodetool/toolsGetEndPoints.html

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool getendpoints keyspace_rf1 example_table 1
172.17.0.4

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool getendpoints keyspace_rf2 example_table 2
172.17.0.2
172.17.0.3
```

8. Відключиіть одну з нод. Для кожного з кейспейсів визначить з якими рівнями *consistency* можемо читати та писати, і які з них забезпечують *strong consistency*

https://docs.datastax.com/en/cql/3.1/cql/cql_reference/consistency_r.html

```
┌──(kali㉿kali)-[~]
└─$ docker stop cassandra-node3
cassandra-node3

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> CONSISTENCY ONE;
Consistency level set to ONE.
cqlsh> SELECT * FROM keyspace_rf1.example_table WHERE id = 1;

 id | data
----+-------
  1 | test1
```

```
cqlsh> SELECT * FROM keyspace_rf1.example_table WHERE id = 1;

 id | data
----+-------
  1 | test1

(1 rows)
cqlsh> CONSISTENCY ALL;
Consistency level set to ALL.
cqlsh> SELECT * FROM keyspace_rf1.example_table WHERE id = 1;

 id | data
----+-------
  1 | test1
```

```
cqlsh> INSERT INTO keyspace_rf1.example_table (id, data) VALUES (1, 'text');
cqlsh> CONSISTENCY QUORUM;
Consistency level set to QUORUM.
cqlsh> INSERT INTO keyspace_rf1.example_table (id, data) VALUES (1, 'text');
cqlsh> CONSISTENCY ONE;
Consistency level set to ONE.
cqlsh> INSERT INTO keyspace_rf1.example_table (id, data) VALUES (1, 'text');
cqlsh>
```

Читає і записує успішно для першого кейпейсу на всіх рівнях

2)Для кейспейсу 2 читає та записує успішно оркім рівня CONSISTENCY QUORUM;

```
cqlsh> INSERT INTO keyspace_rf2.example_table (id, data) VALUES (1, 'text');
cqlsh> SELECT * FROM keyspace_rf2.example_table WHERE id = 1;

 id | data
----+-------
  1 | text

(1 rows)
cqlsh> CONSISTENCY QUORUM;
Consistency level set to QUORUM.
cqlsh> INSERT INTO keyspace_rf2.example_table (id, data) VALUES (2, 'text');
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0
t achieve consistency level QUORUM" info={\'consistency\': \'QUORUM\', \'required_repli
cqlsh> SELECT * FROM keyspace_rf2.example_table WHERE id = 2;
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0
t achieve consistency level QUORUM" info={\'consistency\': \'QUORUM\', \'required_repli
cqlsh> CONSISTENCY ALL;
Consistency level set to ALL.
cqlsh> INSERT INTO keyspace_rf2.example_table (id, data) VALUES (3, 'text');
cqlsh> SELECT * FROM keyspace_rf2.example_table WHERE id = 3;

 id | data
----+-------
  3 | text
```

3) Для кейспейсу 3 не читає на рівні CONSISTENCY QUORUM та не пише на рівні CONSISTENCY ALL

```
cqlsh> CONSISTENCY ONE;
Consistency level set to ONE.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, 'text');
cqlsh> SELECT * FROM keyspace_rf1.example_table WHERE id = 1;

 id | data
----+------
  1 | text

(1 rows)
cqlsh> CONSISTENCY QUORUM;
Consistency level set to QUORUM.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (2, 'text');
cqlsh> SELECT * FROM keyspace_rf1.example_table WHERE id = 2;
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0.1:9042 datacenter1>: Un
t achieve consistency level QUORUM" info={\'consistency\': \'QUORUM\', \'required_replicas\': 1, \'alive_replic
cqlsh> CONSISTENCY ALL;
Consistency level set to ALL.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (3, 'text');
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0.1:9042 datacenter1>: Un
t achieve consistency level ALL" info={\'consistency\': \'ALL\', \'required_replicas\': 3, \'alive_replicas\':
cqlsh> SELECT * FROM keyspace_rf1.example_table WHERE id = 3;

 id | data
----+------

(0 rows)
cqlsh>
```

9. Зробить так щоб три ноди працювали, але не бачили одна одну по мережі (відключити зв'язок між ними)

```
┌──(kali㉿kali)-[~]
└─$ docker network create --driver bridge network1
c7cf71df32d9b004e372d0147c190f394c058a66df717d22135714630098ca9e

┌──(kali㉿kali)-[~]
└─$ docker network create --driver bridge network2
76e5852154a551cfab5f5955d34246973c917e90a86aaae03dea4d60eb96779a

┌──(kali㉿kali)-[~]
└─$ docker network create --driver bridge network3
6eb2a209df66a4eda07f7e68dd7bb8ca2cd27c97c2d674d4065e87199af8939f

┌──(kali㉿kali)-[~]
└─$ docker network disconnect bridge cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network disconnect bridge cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network disconnect bridge cassandra-node3

┌──(kali㉿kali)-[~]
└─$ docker network connect network1 cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network connect network2 cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network connect network3 cassandra-node3
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool status
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address     Load        Tokens  Owns  Host ID                               Rack
DN  172.17.0.4  210.79 KiB  16      ?     fd7c433c-1595-403c-8a2c-3b237c9bbe79  rack1
UN  172.17.0.3  198.91 KiB  16      ?     921a5362-771e-4f38-a8d5-76d52b43a522  rack1
DN  172.17.0.2  138.22 KiB  16      ?     085334f5-545b-4d9c-a8be-d40084b6e877  rack1

Note: Non-system keyspaces don't have the same replication settings, effective ownership information is meaningless
```

10. Для кейспейсу з *replication factor* 3 задайте рівень consistency рівним 1. Виконайте запис одного й того самого значення, з однаковим primary key, але різними іншими значенням на кожну з нод (тобто створіть конфлікт)

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> CONSISTENCY ONE;
Consistency level set to ONE.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, 'random');
cqlsh>

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node2 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, 'random1');
cqlsh>
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node3 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, 'random3');
cqlsh>
```

11. Об’єднайте ноди в кластер і визначте яке значення було прийнято кластером та за яким принципом

```
┌──(kali㉿kali)-[~]
└─$ docker network disconnect network1 cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network disconnect network2 cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network disconnect network3 cassandra-node3

┌──(kali㉿kali)-[~]
└─$ docker network connect bridge cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network connect bridge cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network connect bridge cassandra-node3

┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 nodetool status
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address     Load        Tokens  Owns  Host ID                               Rack
DN  172.17.0.4  210.79 KiB  16      ?     fd7c433c-1595-403c-8a2c-3b237c9bbe79  rack1
UN  172.17.0.3  198.91 KiB  16      ?     921a5362-771e-4f38-a8d5-76d52b43a522  rack1
DN  172.17.0.2  138.22 KiB  16      ?     085334f5-545b-4d9c-a8be-d40084b6e877  rack1
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> SELECT * FROM keyspace_rf3.example_table
   ... ;

 id | data
----+--------
  1 | random
  2 |   text

(2 rows)
```

12. Перевірте поведінку *lightweight transactions* для попередніх пунктів у розділеному та не розділеному на три частини кластері https://docs.datastax.com/en/cql-oss/3.3/cql/cql_using/useInsertLWT.html

розділяємо

```
┌──(kali㉿kali)-[~]
└─$ docker network disconnect bridge cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network disconnect bridge cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network disconnect bridge cassandra-node3
```

```
┌──(kali㉿kali)-[~]
└─$ docker network connect network1 cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network connect network2 cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network connect network3 cassandra-node3
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it cassandra-node1 cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.3 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, 'random1') IF NOT EXISTS;
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0.1:9042 datacenter1>: Unavailable('Error from server: code=1000 [
t achieve consistency level SERIAL" info={\'consistency\': \'SERIAL\', \'required_replicas\': 2, \'alive_replicas\': 1}')})
```

```
cqlsh> UPDATE keyspace_rf3.example_table SET data='text' WHERE id=1 IF EXISTS;
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0.1:9042 datacenter1>: Unavailable('Error from server: code=1000 [
t achieve consistency level SERIAL" info={\'consistency\': \'SERIAL\', \'required_replicas\': 2, \'alive_replicas\': 1}')})
```

```
cqlsh> DELETE FROM keyspace_rf3.example_table WHERE id=1 IF EXISTS
   ... ;
NoHostAvailable: ('Unable to complete the operation against any hosts', {<Host: 127.0.0.1:9042 datacenter1>: Unavailable('Error from server: code=1000 [
t achieve consistency level SERIAL" info={\'consistency\': \'SERIAL\', \'required_replicas\': 2, \'alive_replicas\': 1}')})
cqlsh>
```

З’єднуємо

```
┌──(kali㉿kali)-[~]
└─$ docker network disconnect network1 cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network disconnect network2 cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network disconnect network3 cassandra-node3

┌──(kali㉿kali)-[~]
└─$ docker network connect bridge cassandra-node1

┌──(kali㉿kali)-[~]
└─$ docker network connect bridge cassandra-node2

┌──(kali㉿kali)-[~]
└─$ docker network connect bridge cassandra-node3
```

```
cqlsh> INSERT INTO keyspace_rf3.example_table (id, data) VALUES (1, 'random') IF NOT EXISTS;

 [applied] | id | data
-----------+----+---------
     False |  1 | random3

cqlsh> UPDATE keyspace_rf3.example_table SET data='text' WHERE id=1 IF EXISTS;

 [applied]
-----------
      True

cqlsh> DELETE FROM keyspace_rf3.example_table WHERE id=1 IF EXISTS ;

 [applied]
-----------
      True
```

Вимогу до оформлення протоколу:

Завдання здається особисто без протоколу, або надсилається протокол який має містити:

- команди та результати їх виконання