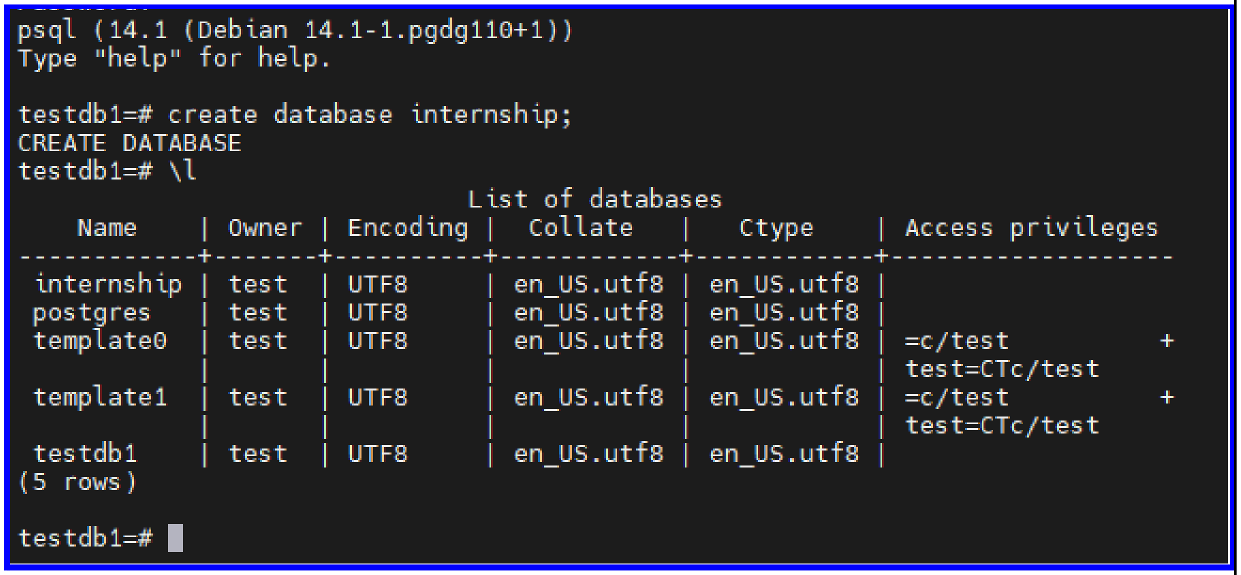
4. Create a database(internship) and a few tables in the database.

Creating database "***internship***"through **client connectoin to Original postgresDB**

create database internship;



```
psql (14.1 (Debian 14.1-1.pgdg110+1))
Type "help" for help.

testdb1=# create database internship;
CREATE DATABASE
testdb1=# \l
                              List of databases
    Name    | Owner | Encoding |  Collate   |   Ctype    | Access privileges
------------+-------+----------+------------+------------+-------------------
 internship | test  | UTF8     | en_US.utf8 | en_US.utf8 |
 postgres   | test  | UTF8     | en_US.utf8 | en_US.utf8 |
 template0  | test  | UTF8     | en_US.utf8 | en_US.utf8 | =c/test          +
            |       |          |            |            | test=CTc/test
 template1  | test  | UTF8     | en_US.utf8 | en_US.utf8 | =c/test          +
            |       |          |            |            | test=CTc/test
 testdb1    | test  | UTF8     | en_US.utf8 | en_US.utf8 |
(5 rows)

testdb1=#
```

To verify client make changes in Older postgresDB

Getting inside **older postgresDB in default Namespace**

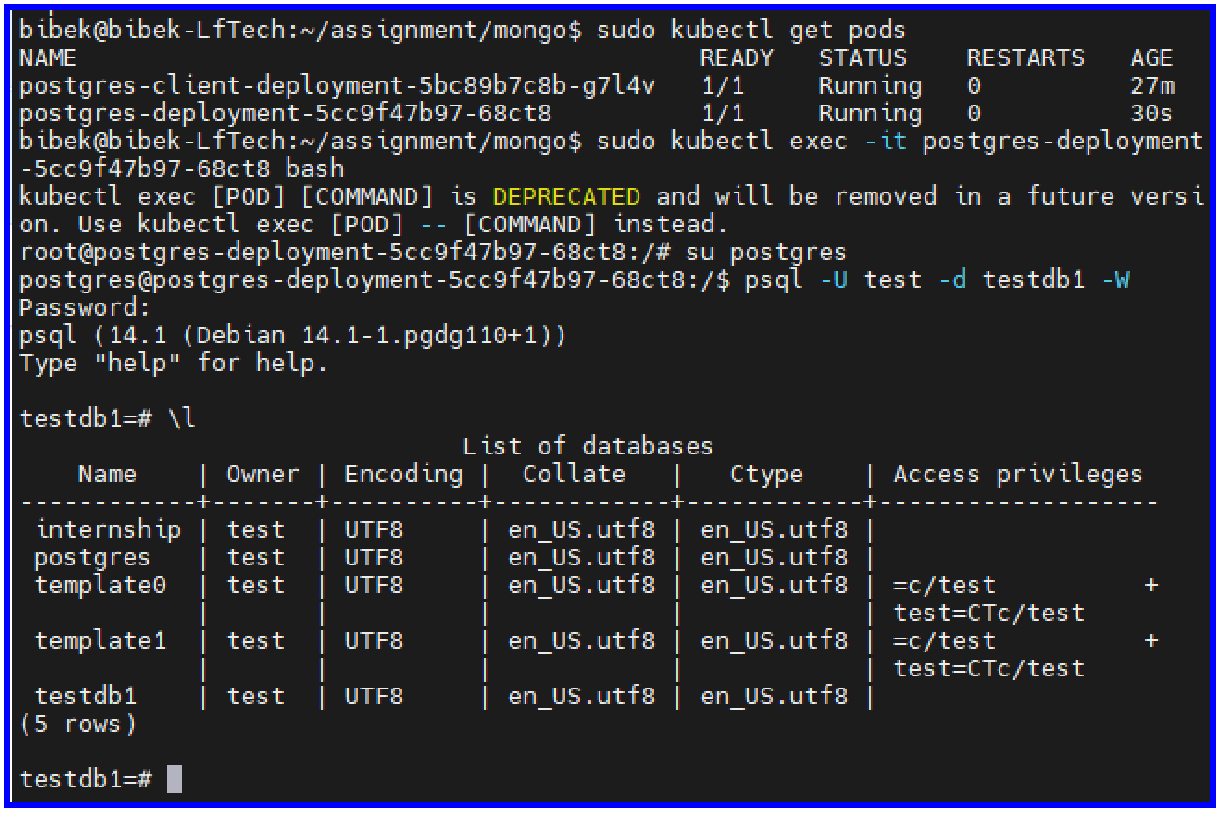
sudo kubectl exec -it postgres-deployment-5cc9f47b97-68ct8 bash

And checking the internship database has been created or not

su postgres

psql -U test -d testdb1 -W

And view all databases



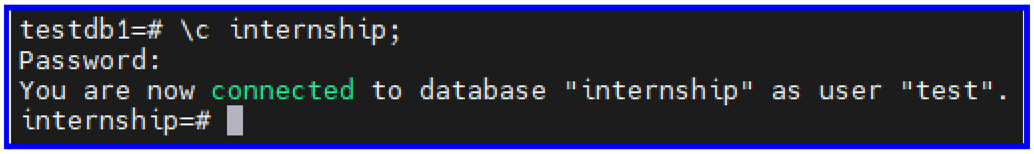
```
bibek@bibek-LfTech:~/assignment/mongo$ sudo kubectl get pods
NAME                                         READY   STATUS    RESTARTS   AGE
postgres-client-deployment-5bc89b7c8b-g7l4v   1/1     Running   0          27m
postgres-deployment-5cc9f47b97-68ct8          1/1     Running   0          30s
bibek@bibek-LfTech:~/assignment/mongo$ sudo kubectl exec -it postgres-deployment
-5cc9f47b97-68ct8 bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future versi
on. Use kubectl exec [POD] -- [COMMAND] instead.
root@postgres-deployment-5cc9f47b97-68ct8:/# su postgres
postgres@postgres-deployment-5cc9f47b97-68ct8:/$ psql -U test -d testdb1 -W
Password:
psql (14.1 (Debian 14.1-1.pgdg110+1))
Type "help" for help.

testdb1=# \l
                              List of databases
    Name    | Owner | Encoding |  Collate   |   Ctype    | Access privileges
------------+-------+----------+------------+------------+-------------------
 internship | test  | UTF8     | en_US.utf8 | en_US.utf8 |
 postgres   | test  | UTF8     | en_US.utf8 | en_US.utf8 |
 template0  | test  | UTF8     | en_US.utf8 | en_US.utf8 | =c/test          +
            |       |          |            |            | test=CTc/test
 template1  | test  | UTF8     | en_US.utf8 | en_US.utf8 | =c/test          +
            |       |          |            |            | test=CTc/test
 testdb1    | test  | UTF8     | en_US.utf8 | en_US.utf8 |
(5 rows)

testdb1=# 
```

Here we can see that **internship database** *<u>created through client connection</u>* ***can be seen in original database(*****Older PostgresDB*****)***

<u>Creating tables through client(PSQL-NameSpace) connection to Older PostgresDB</u>

<u>Connecting to database **Internship**</u>

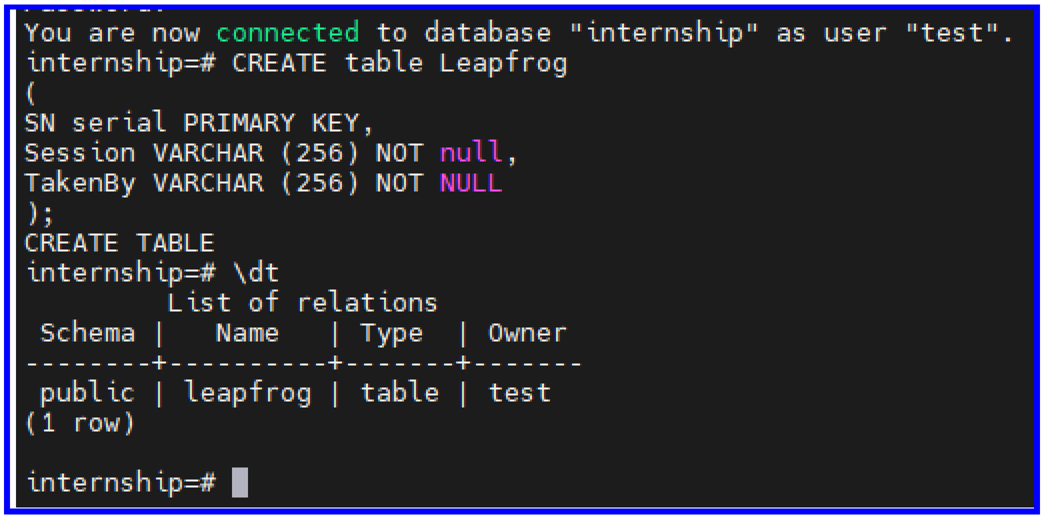
\c internship;



```
testdb1=# \c internship;
Password:
You are now connected to database "internship" as user "test".
internship=# 
```

<u>Creating table “**Leapfrog**”</u>

CREATE table Leapfrog
(
SN serial PRIMARY KEY,
Session VARCHAR (256) NOT null,
TakenBy VARCHAR (256) NOT NULL
);



```
You are now connected to database "internship" as user "test".
internship=# CREATE table Leapfrog
(
SN serial PRIMARY KEY,
Session VARCHAR (256) NOT null,
TakenBy VARCHAR (256) NOT NULL
);
CREATE TABLE
internship=# \dt
         List of relations
 Schema |   Name   | Type  | Owner
--------+----------+-------+-------
 public | leapfrog | table | test
(1 row)

internship=# 
```

*<u>Name of table</u> is "***Leapfrog***"*

Three columns are added - **SN (***serial type***), Session (***VARCHAR type***)** *and* **TakenBy (***VARCHAR***).**

<u>Serial type</u> *will assign numeric value itself starting from 1*

<u>VARCHAR</u> type *is used to give character as value* ***i.e. string***

<u>NOT NULL</u> *means the input should not be empty while inserting data in to the table*

<u>PRIMARY KEY</u> *defines the unique ID for the data in that column to be identified while querying.*

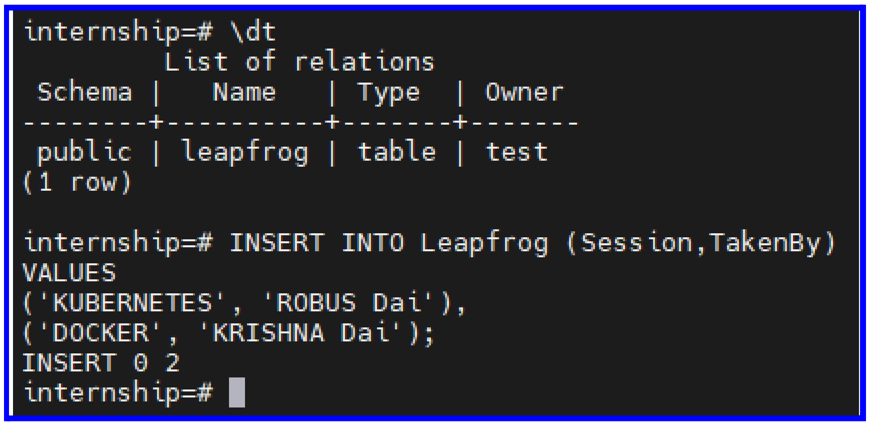
<u>Inserting value into the table Leapfrog</u>

INSERT INTO Leapfrog (Session,TakenBy)

VALUES

('KUBERNETES', 'ROBUS Dai'),

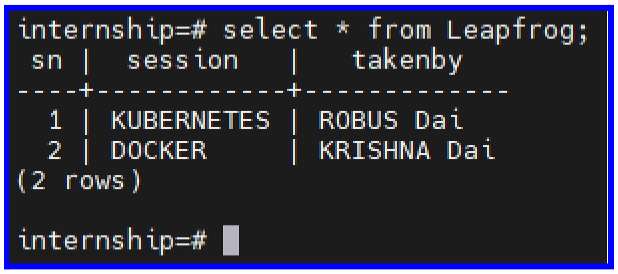
('DOCKER', 'KRISHNA Dai');



```
internship=# \dt
         List of relations
 Schema |   Name   | Type  | Owner
--------+----------+-------+-------
 public | leapfrog | table | test
(1 row)

internship=# INSERT INTO Leapfrog (Session,TakenBy)
VALUES
('KUBERNETES', 'ROBUS Dai'),
('DOCKER', 'KRISHNA Dai');
INSERT 0 2
internship=# 
```

To view data from table Leapfrog

select * from Leapfrog;



```
internship=# select * from Leapfrog;
 sn |   session   |   takenby
----+-------------+-------------
  1 | KUBERNETES  | ROBUS Dai
  2 | DOCKER      | KRISHNA Dai
(2 rows)

internship=# 
```

To delete table

drop table Leapfrog;

To delete database

drop database internship;

In this way in Kubernetes(Minikube) Cluster

- Postgres Pod was created using PV and PVC for persisting the data of DB
- Postgres-Client pod was created in new NameSpace
- Made connection to the database using client pod to Postgres-Pod (via CoreDNS hostname)
- And few practices were done (creating database, creating table, inserting values, deleting table, deleting database, etc)

Thank you !!