DBMS LAB WEEK-1

PART -A

1.Familiar Online Application: **Instagram**

a. Total Number of Daily Active Users who engage with augmented reality on Instagram
→ Instagram has 500+ million daily active users (DAUs) globally, ranking it second behind Facebook for the social network with the highest daily logins.
→The U.S. has the most snapchat users (120 million), followed by India (80 million).

I used Google chrome for my estimation process.
Website: https://backlinko.com/instagram-users

b. Three main kind of data for the application:
i. Geolocational data
ii. Transactional data from Facebook products and services.
iii. Data that links users to the photos they took, tagged or liked

c. Estimation
i. For location and all Instagram uses 1 to 2 MB per time.
ii. The data is used for transaction is approximately 10 to 20 MB.
iii. Uploading photos uses quite a lot more data. You should expect to use about 2 to 4 MB for every photo you post on Instagram. Posting videos take twice of that of the photos.
Liking or tagging approximately 40 photos on an Instagram feed uses up 1MB of data.

The basis for my estimate is Instagram, Google and Youtube.

d. The total amount of data managed by the application provider is nearly 17500 + MB

e. Suggestions:
- Get a stronger CPU
  The better your CPU, the faster and more efficient your database will be.
  The more powerful your CPU is, the less strain it will be under when tasked with multiple applications and requests.
- Data defragmentation
  If you're having trouble with a slow database, another possible solution is data defragmentation.

When many records are written to the database and time goes by, the records are fragmented in MySQL’s internal data files and on the disk itself.

2.How Library Works…

- Library management systems help libraries keep track of the books and their checkouts, as well as members’ subscriptions and profiles.
- Library management systems also involve maintaining the database for entering new books and recording books that have been borrowed with their respective due dates.

List of data that are needed to store with respect to books, patrons:

- **Library:** The central part of the organization for which this software has been designed. It has attributes like ‘Name’ to distinguish it from any other libraries and ‘Address’ to describe its location.

- **Book:** The basic building block of the system. Every book will have ISBN, Title, Subject, Publishers, etc.

- **BookItem:** Any book can have multiple copies, each copy will be considered a book item in our system. Each book item will have a unique barcode.

- **Account:** We will have two types of accounts in the system, one will be a general member, and the other will be a librarian.

- **LibraryCard:** Each library user will be issued a library card, which will be used to identify users while issuing or returning books.

- **BookReservation:** Responsible for managing reservations against book items.

- **BookLending:** Manage the checking-out of book items.

- **Catalog:** Catalogs contain list of books sorted on certain criteria. Our system will support searching through four catalogs: Title, Author, Subject, and Publish-date.

- **Fine:** This class will be responsible for calculating and collecting fines from library members.

- **Author:** This class will encapsulate a book author.
- **Rack:** Books will be placed on racks. Each rack will be identified by a rack number and will have a location identifier to describe the physical location of the rack in the library.
- **Notification:** This class will take care of sending notifications to library members.

3.5 different database management systems:

i. Redis
ii. SQLite
iii. Apache HBase
iv. MongoDB
v. DynamoDB

a. **Market Share:**
   - Redis → 1.25 percent
   - SQLite → 0.24 percent
   - Apache HBase → 0.96 percent
   - MongoDB → 3.42 percent
   - DynamoDB → 0.99 percent

b. **Type of DBMS:**
   - Redis → Open Source DBMS
   - SQLite → Relational DBMS
   - Apache HBase → Non-Relational DBMS
   - MongoDB → NoSQL DBMS
   - DynamoDB → NoSQL DBMS

c. **An Application that uses the DBMS:**
   - Redis → Twitter
   - SQLite → Adobe Systems
   - Apache HBase → Medical
   - MongoDB → Zendesk

DynamoDB → Amazon

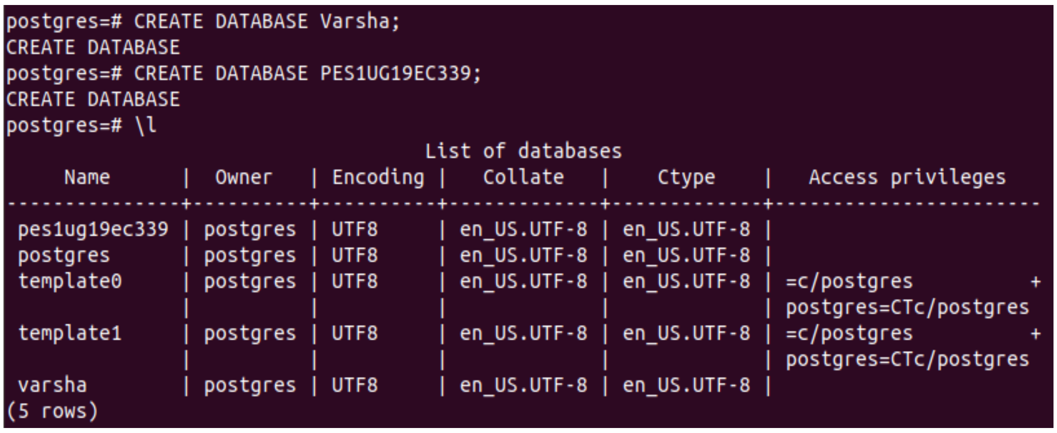
<u>**PART -B**</u>

BASIC SQL COMMANDS:

- **CREATE DATABASE:**

  Create Database creates a new PostgreSQL database.

  Syntax: CREATE DATABASE <name>;

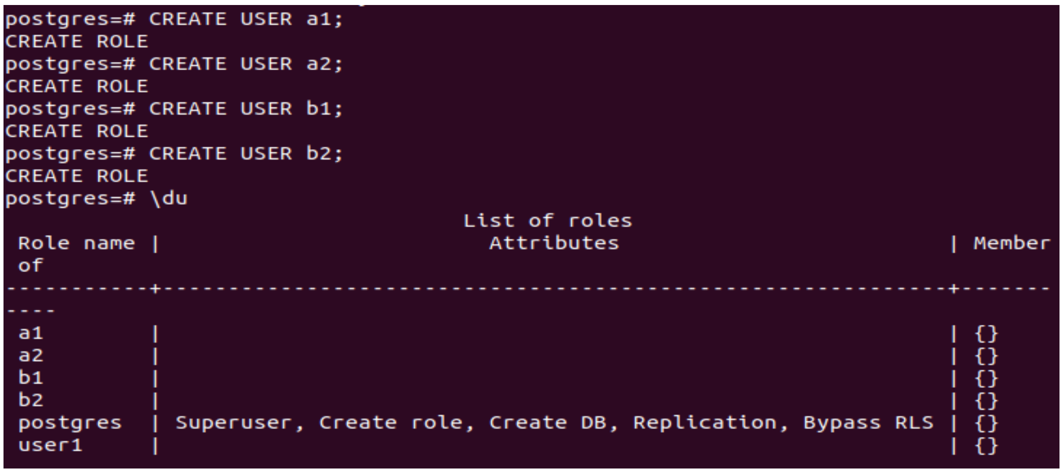


```
postgres=# CREATE DATABASE Varsha;
CREATE DATABASE
postgres=# CREATE DATABASE PES1UG19EC339;
CREATE DATABASE
postgres=# \l
                                  List of databases
     Name      |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
---------------+----------+----------+-------------+-------------+-----------------------
 pes1ug19ec339 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
 template1     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
 varsha        | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
(5 rows)
```

- **CREATE USER**:

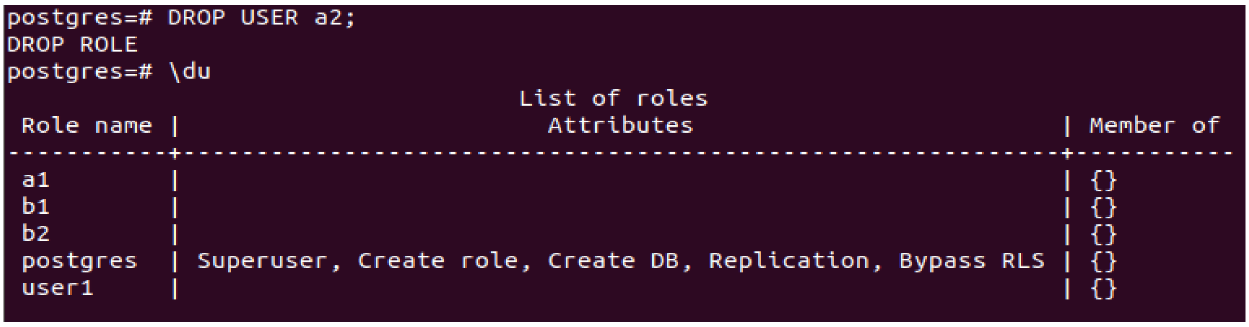
  Create User adds a new user to a PostgreSQL database cluster.

  Syntax: CREATE USER <name>;



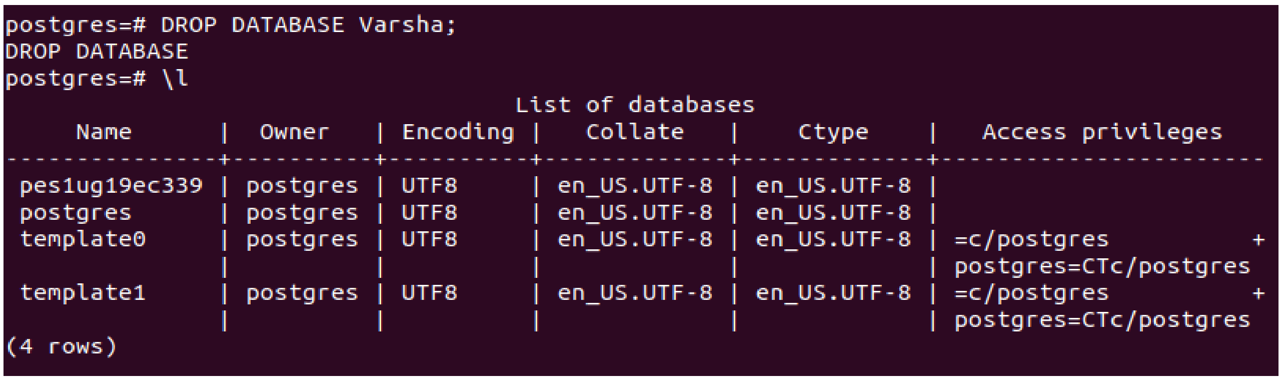
```
postgres=# CREATE USER a1;
CREATE ROLE
postgres=# CREATE USER a2;
CREATE ROLE
postgres=# CREATE USER b1;
CREATE ROLE
postgres=# CREATE USER b2;
CREATE ROLE
postgres=# \du
                                   List of roles
 Role name |                         Attributes                         | Member
 of
-----------+------------------------------------------------------------+-------
----
 a1        |                                                            | {}
 a2        |                                                            | {}
 b1        |                                                            | {}
 b2        |                                                            | {}
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 user1     |                                                            | {}
```

- **DROP USER:**
  Drop user remove a database user.It is simply an another name of DROP ROLE.
  Syntax: DROP USER <username>;



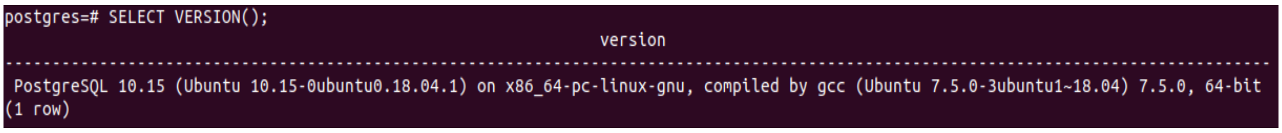
```
postgres=# DROP USER a2;
DROP ROLE
postgres=# \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 a1        |                                                            | {}
 b1        |                                                            | {}
 b2        |                                                            | {}
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 user1     |                                                            | {}
```

- **DROP DATABASE**:
  Drop Database removes the catalog entries for the database and deletes the directory containing the data.
  Syntax: DROP DATABASE <databasename>;



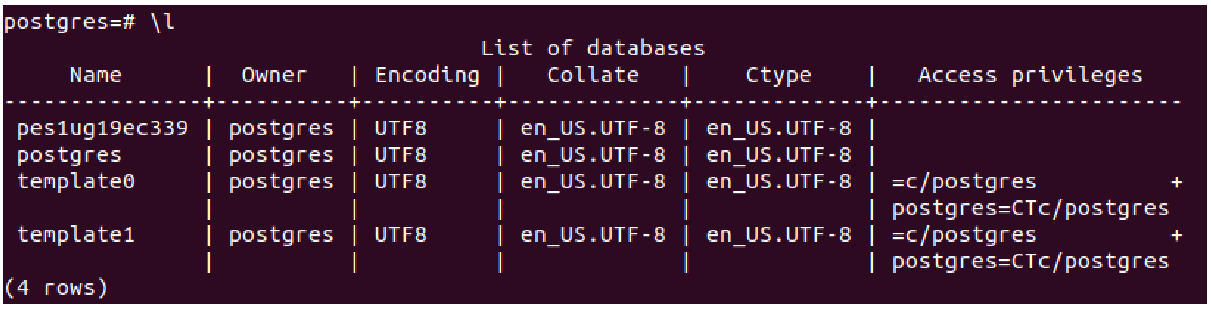
```
postgres=# DROP DATABASE Varsha;
DROP DATABASE
postgres=# \l
                                  List of databases
     Name      |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
---------------+----------+----------+-------------+-------------+-----------------------
 pes1ug19ec339 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
 template1     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
(4 rows)
```

- **CHECK VERSION:**
  It gives current version of PostgreSQL.
  Syntax: SELECT VERSION();



```
postgres=# SELECT VERSION();
                                                             version
-----------------------------------------------------------------------------------------------------------------------------------
 PostgreSQL 10.15 (Ubuntu 10.15-0ubuntu0.18.04.1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0, 64-bit
(1 row)
```

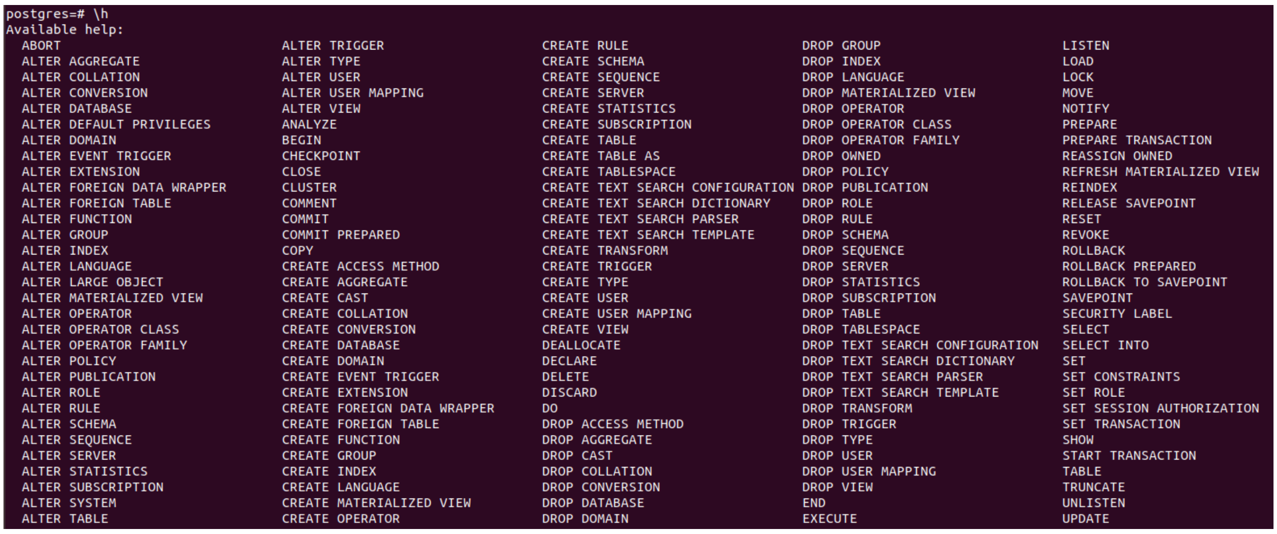
- **LIST DATABSE**:
  \l command is to list Databases.



```
postgres=# \l
                                  List of databases
     Name      |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
---------------+----------+----------+-------------+-------------+-----------------------
 pes1ug19ec339 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
 template1     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
(4 rows)
```

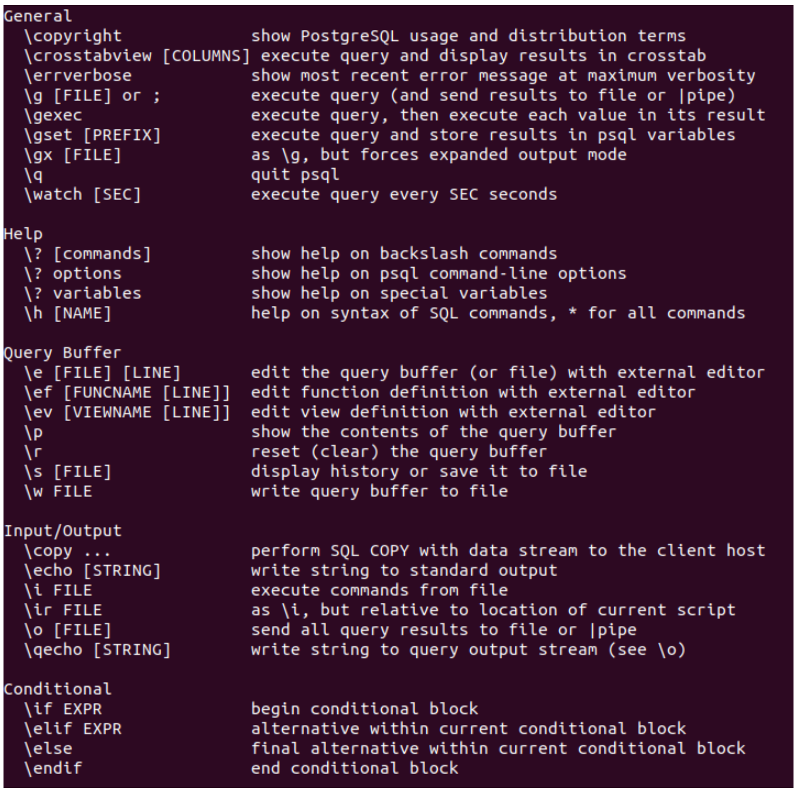
- **\s**
  To display command history.

- **\h**
  This command provides help on syntax of SQL commands.



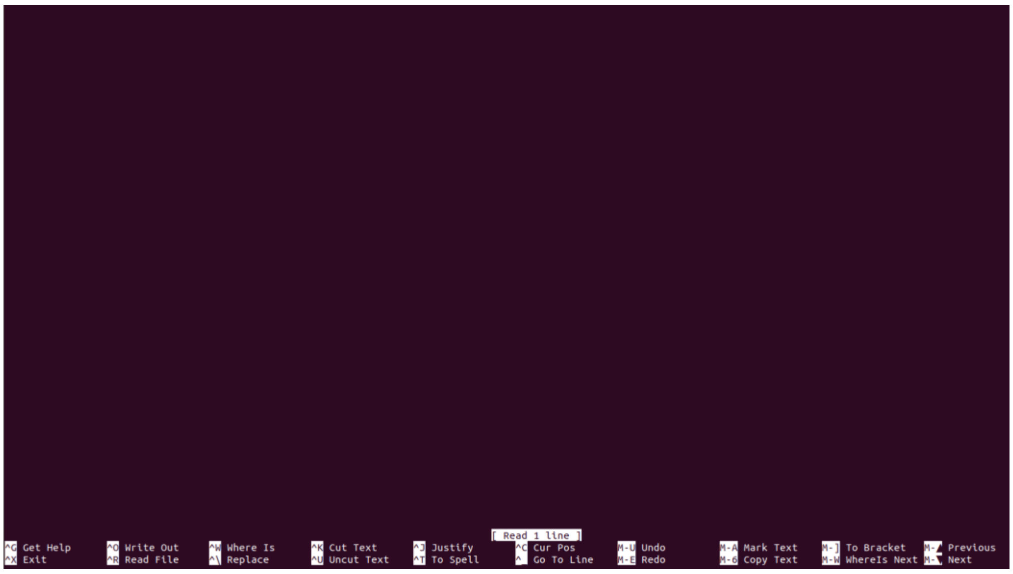
```
postgres=# \h
Available help:
  ABORT                            ALTER TRIGGER                    CREATE RULE                      DROP GROUP                       LISTEN
  ALTER AGGREGATE                  ALTER TYPE                       CREATE SCHEMA                    DROP INDEX                       LOAD
  ALTER COLLATION                  ALTER USER                       CREATE SEQUENCE                  DROP LANGUAGE                    LOCK
  ALTER CONVERSION                 ALTER USER MAPPING               CREATE SERVER                    DROP MATERIALIZED VIEW           MOVE
  ALTER DATABASE                   ALTER VIEW                       CREATE STATISTICS                DROP OPERATOR                    NOTIFY
  ALTER DEFAULT PRIVILEGES         ANALYZE                          CREATE SUBSCRIPTION              DROP OPERATOR CLASS              PREPARE
  ALTER DOMAIN                     BEGIN                            CREATE TABLE                     DROP OPERATOR FAMILY             PREPARE TRANSACTION
  ALTER EVENT TRIGGER              CHECKPOINT                       CREATE TABLE AS                  DROP OWNED                       REASSIGN OWNED
  ALTER EXTENSION                  CLOSE                            CREATE TABLESPACE                DROP POLICY                      REFRESH MATERIALIZED VIEW
  ALTER FOREIGN DATA WRAPPER       CLUSTER                          CREATE TEXT SEARCH CONFIGURATION DROP PUBLICATION                 REINDEX
  ALTER FOREIGN TABLE              COMMENT                          CREATE TEXT SEARCH DICTIONARY    DROP ROLE                        RELEASE SAVEPOINT
  ALTER FUNCTION                   COMMIT                           CREATE TEXT SEARCH PARSER        DROP RULE                        RESET
  ALTER GROUP                      COMMIT PREPARED                  CREATE TEXT SEARCH TEMPLATE      DROP SCHEMA                      REVOKE
  ALTER INDEX                      COPY                             CREATE TRANSFORM                 DROP SEQUENCE                    ROLLBACK
  ALTER LANGUAGE                   CREATE ACCESS METHOD             CREATE TRIGGER                   DROP SERVER                      ROLLBACK PREPARED
  ALTER LARGE OBJECT               CREATE AGGREGATE                 CREATE TYPE                      DROP STATISTICS                  ROLLBACK TO SAVEPOINT
  ALTER MATERIALIZED VIEW          CREATE CAST                      CREATE USER                      DROP SUBSCRIPTION                SAVEPOINT
  ALTER OPERATOR                   CREATE COLLATION                 CREATE USER MAPPING              DROP TABLE                       SECURITY LABEL
  ALTER OPERATOR CLASS             CREATE CONVERSION                CREATE VIEW                      DROP TABLESPACE                  SELECT
  ALTER OPERATOR FAMILY            CREATE DATABASE                  DEALLOCATE                       DROP TEXT SEARCH CONFIGURATION   SELECT INTO
  ALTER POLICY                     CREATE DOMAIN                    DECLARE                          DROP TEXT SEARCH DICTIONARY      SET
  ALTER PUBLICATION                CREATE EVENT TRIGGER             DELETE                           DROP TEXT SEARCH PARSER          SET CONSTRAINTS
  ALTER ROLE                       CREATE EXTENSION                 DISCARD                          DROP TEXT SEARCH TEMPLATE        SET ROLE
  ALTER RULE                       CREATE FOREIGN DATA WRAPPER      DO                               DROP TRANSFORM                   SET SESSION AUTHORIZATION
  ALTER SCHEMA                     CREATE FOREIGN TABLE             DROP ACCESS METHOD               DROP TRIGGER                     SET TRANSACTION
  ALTER SEQUENCE                   CREATE FUNCTION                  DROP AGGREGATE                   DROP TYPE                        SHOW
  ALTER SERVER                     CREATE GROUP                     DROP CAST                        DROP USER                        START TRANSACTION
  ALTER STATISTICS                 CREATE INDEX                     DROP COLLATION                   DROP USER MAPPING                TABLE
  ALTER SUBSCRIPTION               CREATE LANGUAGE                  DROP CONVERSION                  DROP VIEW                        TRUNCATE
  ALTER SYSTEM                     CREATE MATERIALIZED VIEW         DROP DATABASE                    END                              UNLISTEN
  ALTER TABLE                      CREATE OPERATOR                  DROP DOMAIN                      EXECUTE                          UPDATE
```

- **\?**
  This is used to know all available commands.



```
General
  \copyright             show PostgreSQL usage and distribution terms
  \crosstabview [COLUMNS] execute query and display results in crosstab
  \errverbose            show most recent error message at maximum verbosity
  \g [FILE] or ;         execute query (and send results to file or |pipe)
  \gexec                 execute query, then execute each value in its result
  \gset [PREFIX]         execute query and store results in psql variables
  \gx [FILE]             as \g, but forces expanded output mode
  \q                     quit psql
  \watch [SEC]           execute query every SEC seconds

Help
  \? [commands]          show help on backslash commands
  \? options             show help on psql command-line options
  \? variables           show help on special variables
  \h [NAME]              help on syntax of SQL commands, * for all commands

Query Buffer
  \e [FILE] [LINE]       edit the query buffer (or file) with external editor
  \ef [FUNCNAME [LINE]]  edit function definition with external editor
  \ev [VIEWNAME [LINE]]  edit view definition with external editor
  \p                     show the contents of the query buffer
  \r                     reset (clear) the query buffer
  \s [FILE]              display history or save it to file
  \w FILE                write query buffer to file

Input/Output
  \copy ...              perform SQL COPY with data stream to the client host
  \echo [STRING]         write string to standard output
  \i FILE                execute commands from file
  \ir FILE               as \i, but relative to location of current script
  \o [FILE]              send all query results to file or |pipe
  \qecho [STRING]        write string to query output stream (see \o)

Conditional
  \if EXPR               begin conditional block
  \elif EXPR             alternative within current conditional block
  \else                  final alternative within current conditional block
  \endif                 end conditional block
```

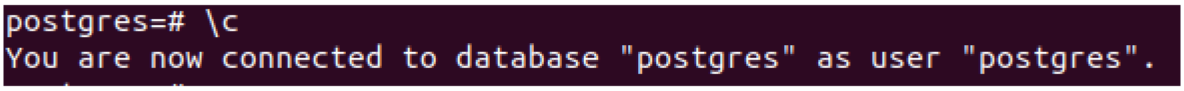
- **\e**
  This is used to open your text editor to write SQL commands.



```
                                          [ Read 1 line ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo       M-A Mark Text  M-] To Bracket    M-▲ Previous
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line  M-E Redo       M-6 Copy Text  M-W WhereIs Next  M-▼ Next
```

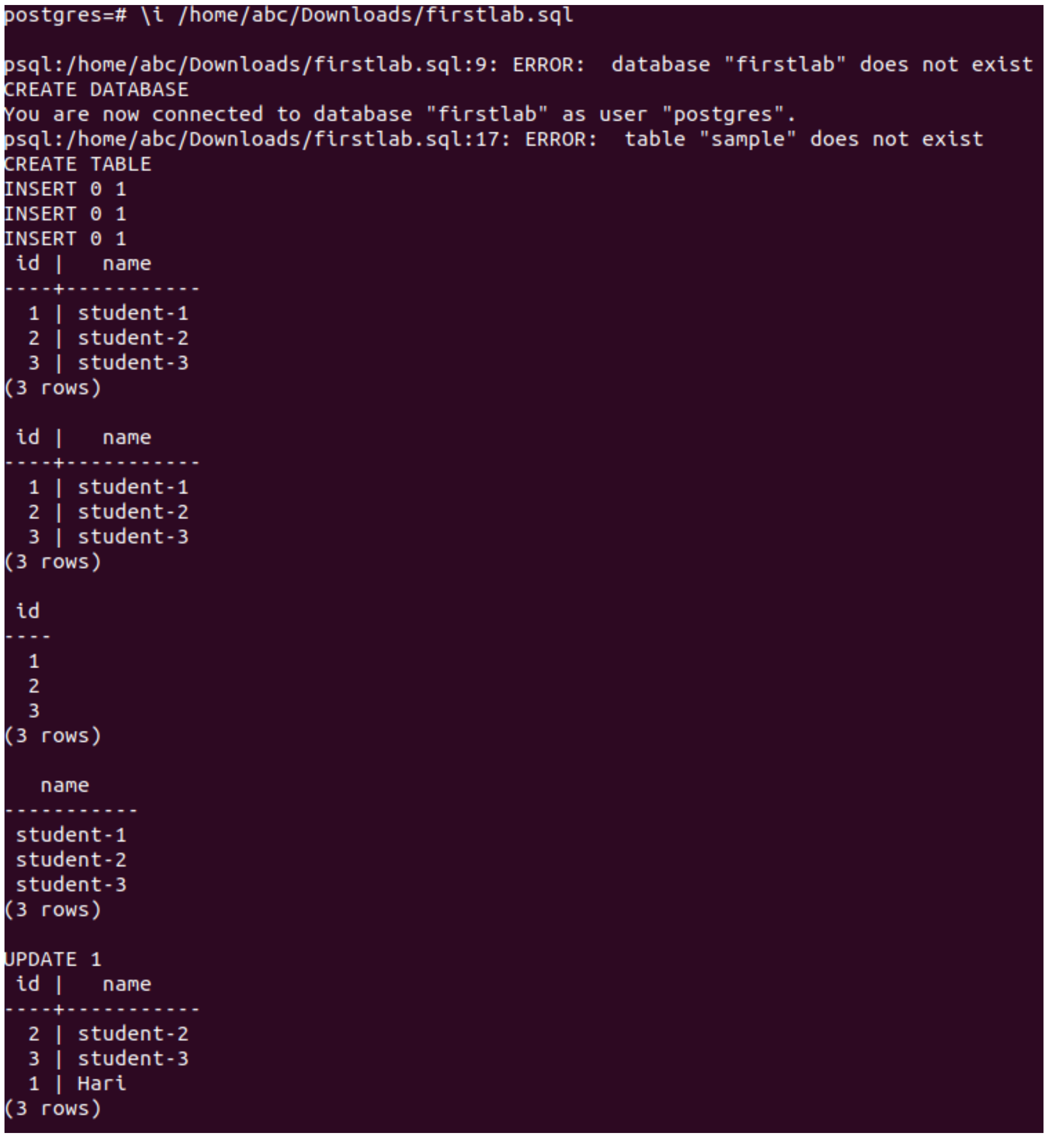
- **\q**
  This is used to quit psql terminal.

- **\c**
  Tis command is used to connect a file to the database.



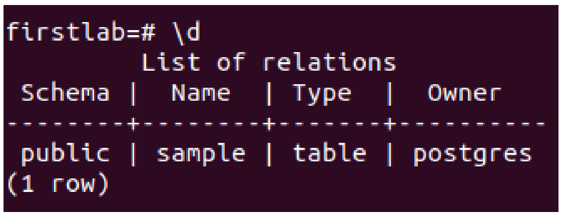
```
postgres=# \c
You are now connected to database "postgres" as user "postgres".
```

- **\i**
  This command is used to execute psql command from a file.



```
postgres=# \i /home/abc/Downloads/firstlab.sql

psql:/home/abc/Downloads/firstlab.sql:9: ERROR:  database "firstlab" does not exist
CREATE DATABASE
You are now connected to database "firstlab" as user "postgres".
psql:/home/abc/Downloads/firstlab.sql:17: ERROR:  table "sample" does not exist
CREATE TABLE
INSERT 0 1
INSERT 0 1
INSERT 0 1
 id |   name
----+----------
  1 | student-1
  2 | student-2
  3 | student-3
(3 rows)

 id |   name
----+----------
  1 | student-1
  2 | student-2
  3 | student-3
(3 rows)

 id
----
  1
  2
  3
(3 rows)

   name
----------
 student-1
 student-2
 student-3
(3 rows)

UPDATE 1
 id |   name
----+----------
  2 | student-2
  3 | student-3
  1 | Hari
(3 rows)
```

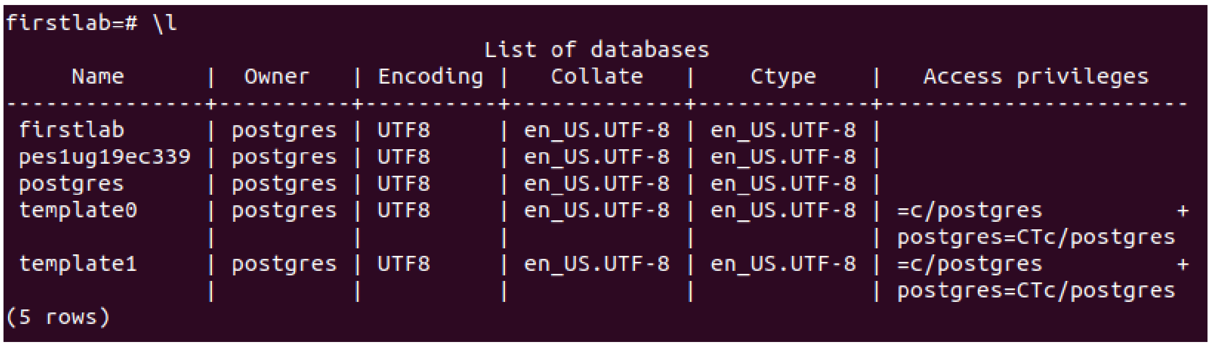
- **\d**
  To describe a table.



```
firstlab=# \d
          List of relations
 Schema |  Name  | Type  |  Owner
--------+--------+-------+----------
 public | sample | table | postgres
(1 row)
```

- **\l**

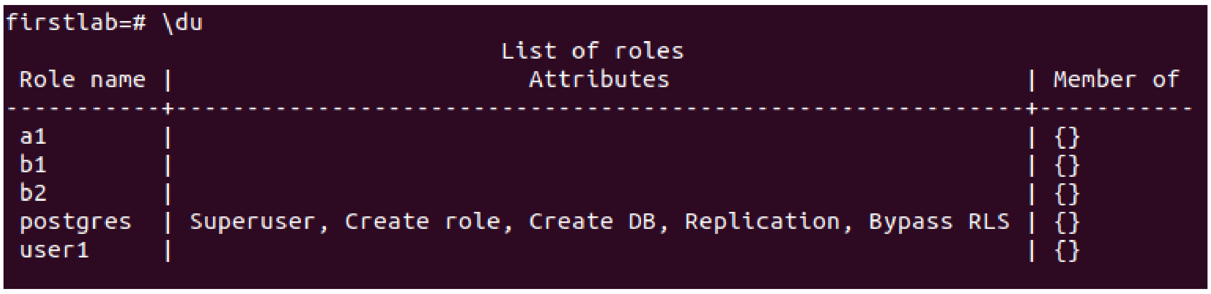
  To list all the databases in the current psql database server.



```
firstlab=# \l
                                    List of databases
     Name      |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
---------------+----------+----------+-------------+-------------+-----------------------
 firstlab      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 pes1ug19ec339 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
 template1     | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
               |          |          |             |             | postgres=CTc/postgres
(5 rows)
```

- **\du**

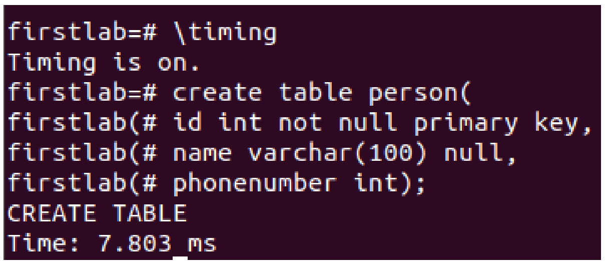
  To list all the users and their assign roles.



```
firstlab=# \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 a1        |                                                            | {}
 b1        |                                                            | {}
 b2        |                                                            | {}
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 user1     |                                                            | {}
```

- **\timing**

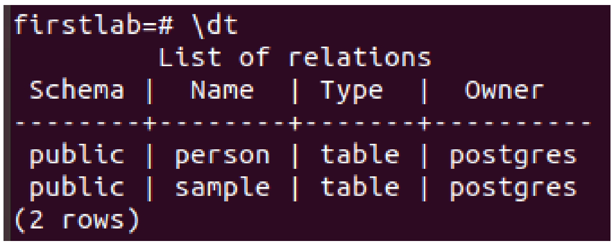
  You can ask psql to print the tie taken to execute every command or query.



```
firstlab=# \timing
Timing is on.
firstlab=# create table person(
firstlab(# id int not null primary key,
firstlab(# name varchar(100) null,
firstlab(# phonenumber int);
CREATE TABLE
Time: 7.803 ms
```

- **\dt**

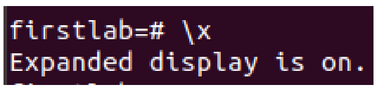
  To list all tables in current database.



```
firstlab=# \dt
         List of relations
 Schema |  Name  | Type  |  Owner
--------+--------+-------+----------
 public | person | table | postgres
 public | sample | table | postgres
(2 rows)
```

- **\x**

  To turn on expanded display.



```
firstlab=# \x
Expanded display is on.
```