```
/*
To install postgresql in ubuntu
        sudo apt install postgresql postgresql-contrib

It will install postgres and create a user postgres. To log into that user
        sudo -iu postgres

Then to run psql
        psql

List available databases
        \l

Switch connection to new database
        \c database username

List available table
        \dt

Describe a table
        \d tablename

List available schema
        \dn

List available functions
        \df

List available views
        \dv

List users and their roles
        \du

Retrieve current version of postgreSQL server
        SELECT VERSION();

Execute the previous command
        \g

command history
        \s

To save command history to a file
        \s filename

Execute psql commands from a file
        \i filename

Get help on psql commands
        \?

To get help on specific psql commands
        \h ALTER TABLE
```

```
Turn on query execution time
	\timing
Use the same command to turn it off.

Quit psql
	\q

*/




CREATE TABLE branch(
	b_id SERIAL PRIMARY KEY,
	bname VARCHAR(30) NOT NULL,
	city VARCHAR(30) NOT NULL
);

CREATE TABLE customer(
	c_id SERIAL PRIMARY KEY,
	cname VARCHAR(30) NOT NULL,
	city VARCHAR(30) NOT NULL
);

CREATE TABLE borrow(
	loan_no SERIAL PRIMARY KEY,
	b_id INT REFERENCES branch(b_id),
	c_id INT REFERENCES customer(c_id),
	amount NUMERIC NOT NULL CHECK (amount >= 1000),
	dat date NOT NULL CHECK (dat <= date(CURRENT_TIMESTAMP))
);


CREATE TABLE borrow(
	loan_no SERIAL,
	b_id INT,
	c_id INT,
	amount NUMERIC NOT NULL,
	dat date NOT NULL,
	PRIMARY KEY(loan_no),
	FOREIGN KEY(b_id) REFERENCES branch(b_id),
	FOREIGN KEY(c_id) REFERENCES customer(c_id),
	CHECK (amount > 1000),
	CHECK (dat <= date(CURRENT_TIMESTAMP))
);

CREATE TABLE borrow(
	loan_no SERIAL,
	b_id INT,
	c_id INT,
	amount NUMERIC NOT NULL,
	dat date NOT NULL,
	CONSTRAINT borro_pkey PRIMARY KEY(loan_no),
	CONSTRAINT borrow_b_id_fkey FOREIGN KEY(b_id) REFERENCES branch(b_id),
	CONSTRAINT borrow_c_id_fkey FOREIGN KEY(c_id) REFERENCES customer(c_id),
	CONSTRAINT borrow_amount_check CHECK (amount > 1000),
	CONSTRAINT borrow_dat_check CHECK (dat <= date(CURRENT_TIMESTAMP))
);
```

```
do $$
BEGIN
	IF (5< 0) THEN RAISE NOTICE 'Enter a non negative number';
	ELSE RAISE NOTICE 'The number is non negative';
	end if;
END $$
LANGUAGE plpgsql;
```

```
-- student database
CREATE TABLE class_(
	class_id SERIAL PRIMARY KEY,
	class_name VARCHAR(30) NOT NULL,
	division VARCHAR(2) NOT NULL DEFAULT 'A',
	st_cnt INT NOT NULL CHECK(st_cnt >= 0),
	UNIQUE(class_name, division)
);

INSERT INTO class_ (class_name, st_cnt)
VALUES ('Ten', 65);

CREATE TABLE student(
	st_id SERIAL PRIMARY KEY,
	st_fname VARCHAR(30) NOT NULL,
	st_lname VARCHAR(30) NOT NULL,
	addr VARCHAR(30) NOT NULL,
	phone VARCHAR(13),
	email VARCHAR(30),
	class_id INT REFERENCES class_(class_id)
);


INSERT INTO student (st_fname, st_lname, addr, phone, email, class_id)
VALUES('Abijith', 'R', 'Address', '+910000000000', 'email', 1),
('Abirami', 'S Pillai', 'Address', '+910000000000', 'email', 1),
('Adish', 'A P', 'Address', '+910000000000', 'email', 1),
('Aiswarya', 'A S', 'Address', '+910000000000', 'email', 1),
('Aiswarya', 'Ramdas', 'Address', '+910000000000', 'email', 1),
('Anagha', 'Sree', 'Address', '+910000000000', 'email', 1),
('Anagha', 'V', 'Address', '+910000000000', 'email', 1),
('Anamika', 'Sudheesh', 'Address', '+910000000000', 'email', 1),
('Ansaf', 'Muhammed P T', 'Address', '+910000000000', 'email', 1),
('Anushamol', 'S', 'Address', '+910000000000', 'email', 1),
('Arjun', 'T B', 'Address', '+910000000000', 'email', 1),
('Aromal', 'V Ashokan', 'Address', '+910000000000', 'email', 1),
('Ashwati', 'Ashok Kumar', 'Address', '+910000000000', 'email', 1),
('Ashwin', 'Das', 'Address', '+910000000000', 'email', 1),
('Ashwin', 'Titus', 'Address', '+910000000000', 'email', 1),
('Ashwin', 'Pradeep', 'Address', '+910000000000', 'email', 1),
('Athira', 'S Pillai', 'Address', '+910000000000', 'email', 1),
('BalaSubhramani', '', 'Address', '+910000000000', 'email', 1),
('Bhagyalekshmi', 'Jaya Prakash', 'Address', '+910000000000', 'email', 1),
('Deepu', 'Krishnan U', 'Address', '+910000000000', 'email', 1),
('Delna', 'K Bijo', 'Address', '+910000000000', 'email', 1),
('Devi', 'R Jayan', 'Address', '+910000000000', 'email', 1),
('Fathima', 'Heera F', 'Address', '+910000000000', 'email', 1),
('Fathimath', 'Shifana E S', 'Address', '+910000000000', 'email', 1),
('Gokul', 'S Krishnan', 'Address', '+910000000000', 'email', 1),
('Jishma', 'Shareena', 'Address', '+910000000000', 'email', 1),
('Jithu', 'P', 'Address', '+910000000000', 'email', 1),
('Kavya', 'T Kunjumon', 'Address', '+910000000000', 'email', 1),
('Manikantan', 'K', 'Address', '+910000000000', 'email', 1),
('Midhun', 'K R', 'Address', '+910000000000', 'email', 1),
('Karupaswamy', 'M', 'Address', '+910000000000', 'email', 1),
('Mohamed', 'Saif Muthanikkatt', 'Address', '+910000000000', 'email', 1),
('Muhsina', 'Binth Abdulla C', 'Address', '+910000000000', 'email', 1),
('Nanditha', 'Reji Krishnan', 'Address', '+910000000000', 'email', 1),
('Navaneeth', 'K', 'Address', '+910000000000', 'email', 1),
('Navaneeth', 'Mohan', 'Address', '+910000000000', 'email', 1),
('Nidhilal', 'M S', 'Address', '+910000000000', 'email', 1),
('Nidhi', 'P', 'Address', '+910000000000', 'email', 1),
('Phebe', 'Joshua', 'Address', '+910000000000', 'email', 1),
('Pooja', 'R', 'Address', '+910000000000', 'email', 1),
('Rahul', '', 'Address', '+910000000000', 'email', 1),
('Rahul', 'R', 'Address', '+910000000000', 'email', 1),
('Sangeeth', 'V S', 'Address', '+910000000000', 'email', 1),
('Sarath', 'A R', 'Address', '+910000000000', 'email', 1),
('Shibina', 'Nazar', 'Address', '+910000000000', 'email', 1),
('Shyam', 'Lal A S', 'Address', '+910000000000', 'email', 1),
('Soumya', 'K', 'Address', '+910000000000', 'email', 1),
('Sreenath', 'B S', 'Address', '+910000000000', 'email', 1),
('Surya', 'Suresh', 'Address', '+910000000000', 'email', 1),
('Vaisakh', 'G', 'Address', '+910000000000', 'email', 1),
('Varsha', 'Salil', 'Address', '+910000000000', 'email', 1),
('Vineetha', 'M Mathews', 'Address', '+910000000000', 'email', 1),
('Vishnudas', 'T K', 'Address', '+910000000000', 'email', 1),
('Vishnujith', 'B', 'Address', '+910000000000', 'email', 1),
('Vysakh', 'R J', 'Address', '+910000000000', 'email', 1),
('Ahsan', 'S', 'Address', '+910000000000', 'email', 1),
('Sajub', 'Ur Rehman K', 'Address', '+910000000000', 'email', 1),
('Abhilash', '', 'Address', '+910000000000', 'email', 1),
('Ajin', 'Thampi', 'Address', '+910000000000', 'email', 1),
('Anjali', 'K V', 'Address', '+910000000000', 'email', 1),
('Aravind', '', 'Address', '+910000000000', 'email', 1),
('Arya', 'Rajan', 'Address', '+910000000000', 'email', 1),
('Lekshmi', 'Mohan', 'Address', '+910000000000', 'email', 1),
('Parvathy', 'P Menon', 'Address', '+910000000000', 'email', 1),
('Vaishakh', 'K', 'Address', '+910000000000', 'email', 1);

SELECT st_id AS "Roll No", CONCAT(st_fname, ' ', st_lname) AS "name" FROM Student;

SELECT class_name, division FROM class_
WHERE (st_cnt > 50);

SELECT s1.st_id AS "Roll No" , CONCAT(s1.st_fname, ' ', s1.st_lname) AS "Name" FROM student
AS s1, student AS s2
WHERE s1.st_id != s2.st_id AND s1.st_fname LIKE s2.st_fname;

-- For selecting distinct fnames from the self join table and displaying it in order
SELECT DISTINCT s1.st_id AS "Roll No" , CONCAT(s1.st_fname, ' ', s1.st_lname) AS "Name" FROM
student AS s1, student AS s2
WHERE s1.st_id != s2.st_id AND s1.st_fname LIKE s2.st_fname;
ORDER BY s1.st_id;

CREATE TABLE teacher(
	tr_id SERIAL PRIMARY KEY,
	tr_fname VARCHAR(30) NOT NULL,
	tr_lname VARCHAR(30) NOT NULL,
	phone VARCHAR(13),
	subject VARCHAR(30) NOT NULL
);

CREATE TABLE stud_class(
	st_id INT REFERENCES student(st_id),
	class_id INT REFERENCES class_(class_id),
	tr_id INT REFERENCES teacher(tr_id),
	UNIQUE(st_id, class_id, tr_id)
);





-- TO DEMONSTRATE JOINS
CREATE TABLE company(
    emp_id INT PRIMARY KEY        NOT NULL,
    name                  TEXT       NOT NULL,
    age                   SMALLINT        NOT NULL CHECK (age>=18),
    address               VARCHAR(50),
    salary               REAL,
    join_date        DATE
);
CREATE TABLE department(
    id INT PRIMARY KEY          NOT NULL,
    dept                  VARCHAR(50) NOT NULL,
    emp_id                INT          NOT NULL
);
```

```
INSERT INTO company(emp_id,name,age,address,salary,join_date) VALUES (1, 'Paul', 32,
'California', 20000.00,'2001-07-13');
INSERT INTO company (emp_id,name,age,address,join_date) VALUES (2, 'Allen', 25, 'Texas', '2007-
12-13');
INSERT INTO company (emp_id,name,age,address,salary,join_date) VALUES (3, 'Teddy', 23,
'Norway', 20000.00, DEFAULT );
INSERT INTO company (emp_id,name,age,address,salary,join_date) VALUES (4, 'Mark', 25, 'Rich-
Mond ', 65000.00, '2007-12-13' ), (5, 'David', 27, 'Texas', 85000.00, '2007-12-13');
INSERT INTO company (emp_id,name,age,address,salary,join_date) VALUES (8, 'Paul', 24,
'Houston ', 20000.00, '2005-07-13' ), (9, 'James', 44, 'Norway ', 5000.00, '2005-07-13' ),
(10, 'James', 45, 'Texas ', 5000.00, '2005-07-13' );

INSERT INTO department (id, dept, emp_id)
VALUES (1, 'IT Billing', 1 );

INSERT INTO department (id, dept, emp_id)
VALUES (2, 'Engineering', 2 );

INSERT INTO department (id, dept, emp_id)
VALUES (3, 'Finance', 8 );

-- CROSS JOIN
SELECT company.emp_id, name, dept FROM company CROSS JOIN department;

-- INNER JOIN
SELECT company.emp_id, name, dept FROM company INNER JOIN department
ON company.emp_id = department.emp_id;

-- LEFT OUTER JOIN or LEFT JOIN
SELECT company.emp_id, name, dept FROM company LEFT OUTER JOIN department
ON company.emp_id = department.emp_id;

-- RIGHT OUTER JOIN or RIGHT JOIN
SELECT company.emp_id, name, dept FROM company RIGHT OUTER JOIN department
ON company.emp_id = department.emp_id;

-- FULL OUTER JOIN or FULL JOIN
SELECT company.emp_id, name, dept FROM company FULL OUTER JOIN department
ON company.emp_id = department.emp_id;

SELECT company.emp_id, name, dept FROM company NATURAL JOIN department;
```