SELECT sname,sid+year s12 FROM student;

SELECT sname s12 FROM student;

SELECT sname AS bros,sid AS age FROM student;

SELECT sname AS "ITBK" FROM student;

SELECT sname,sid AS "ITBK" FROM student;

SELECT \* FROM student AS boys;

SELECT sname,age AS "The Boys" FROM student WHERE sid >121;

SELECT sname,age AS "The Boys" FROM student WHERE sid BETWEEN 100 AND 121;

SELECT rollno,total,avg FROM marks FETCH FIRST 10 ROWS ONLY;

SELECT rollno FROM marks GROUP BY rollno;

SELECT rollno,s1 FROM marks GROUP BY rollno,s1;

SELECT min(rollno),max(s1) FROM marks GROUP BY rollno,s1;

SELECT p1.age,s1.crowd FROM population p1 JOIN security s1 ON p1.people = s1.crowd;

SELECT \* FROM officers f LEFT JOIN ministers m ON m.location = f.location;

SELECT \* FROM officers f RIGHT JOIN ministers m ON m.location = f.location;

SELECT \* FROM officers f INNER JOIN ministers m ON m.location = f.location;

SELECT \* FROM population CROSS JOIN security;

SELECT p1.age,s1.crowd FROM population p1 JOIN security s1 ON p1.people = s1.crowd;

SELECT \* FROM MINISTERS NATURAL JOIN OFFICERS;

SELECT location FROM ministers UNION SELECT location FROM officers;

SELECT location FROM ministers UNION ALL SELECT location FROM officers;

SELECT location FROM ministers UNION DISTINCT SELECT location FROM officers;

SELECT location FROM ministers INTERSECT SELECT location FROM officers;

SELECT location FROM ministers INTERSECT ALL SELECT location FROM officers;

SELECT location FROM ministers EXCEPT SELECT location FROM officers;

SELECT location FROM ministers EXCEPT ALL SELECT location FROM officers;

SELECT location FROM officers EXCEPT ALL SELECT location FROM ministers;

SELECT location FROM officers WHERE EXISTS ( SELECT location FROM ministers WHERE ministers.location = officers.location);

SELECT location FROM officers WHERE EXISTS ( SELECT location FROM ministers WHERE ministers.mid > officers.id);

SELECT location FROM officers WHERE NOT EXISTS ( SELECT location FROM ministers WHERE ministers.mid > officers.id);

SELECT location FROM officers WHERE NOT EXISTS ( SELECT location FROM ministers WHERE ministers.mid < officers.id);

SELECT \* FROM ministers m WHERE m.location IN (SELECT location FROM officers) ;

SELECT m.name,f.count FROM ministers m INNER JOIN officers f ON m.location IN (SELECT location FROM officers f) ;

SELECT m.name,f.count FROM ministers m INNER JOIN officers f ON m.location IN (SELECT location FROM officers f) ;

SELECT p.age,p.location FROM population p WHERE p.people > (SELECT SUM(crowd) FROM security s);

SELECT p.age,p.location FROM population p INNER JOIN security s ON p.people > (SELECT SUM(crowd) FROM security s);

SELECT p.age,p.location,s.name FROM population p INNER JOIN security s ON p.people > (SELECT SUM(crowd) FROM security s);

SELECT p.age,p.location,s.name FROM security s INNER JOIN population p ON s.crowd > (SELECT AVG(people) FROM population p);

SELECT p.age,p.location,s.name FROM security s LEFT JOIN population p ON s.crowd > (SELECT AVG(people) FROM population p);

SELECT p.age,p.location,s.name FROM security s RIGHT JOIN population p ON s.crowd > (SELECT AVG(people) FROM population p);

SELECT p.age,p.location,s.name FROM security s JOIN population p ON s.crowd > (SELECT AVG(people) FROM population p);

CREATE TABLE departments(did int PRIMARY KEY,dname varchar(100) UNIQUE NOT NULL);

CREATE TABLE employees(eid int PRIMARY KEY,fname varchar(30),lname varchar(30),did int REFERENCES departments(did), salary NUMERIC(10,2), hire\_date DATE DEFAULT CURRENT\_DATE);

1.CREATE PROCEDURE insertdept(p\_did IN INT,p\_dname IN VARCHAR(100)) LANGUAGE plpgsql AS $$ BEGIN INSERT INTO DEPARTMENTS (did ,dname) VALUES (p\_did , p\_dname);

END;

$$;

CALL insertdept(108,'civil');

CREATE OR REPLACE FUNCTION getempbydname(deptname TEXT)

RETURNS TABLE (eid INT,fname TEXT, lname TEXT, did int)AS $$

BEGIN

RETURN QUERY

SELECT e.eid,e.fname,e.lname,d.dname

FROM employees e

JOIN departments d ON e.did = d.did

WHERE d.dname = deptname;

END;

$$ LANGUAGE plpgsql;

2.CREATE OR REPLACE FUNCTION getempbydname(deptname VARCHAR(30))

RETURNS TABLE (

eid INT,

fname VARCHAR(50),

lname VARCHAR(50)

)

LANGUAGE plpgsql

AS $$

BEGIN

RETURN QUERY

SELECT e.eid, e.fname, e.lname

FROM employees e

JOIN departments d ON e.did = d.did

WHERE d.dname = deptname;

END;

$$;

3.CREATE OR REPLACE PROCEDURE salaryhike10(deptname VARCHAR(30))

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE employees e SET salary = 1.1 \* salary FROM departments d WHERE d.did = e.did AND d.dname = deptname;

END;

$$;

CREATE OR REPLACE FUNCTION newemp(empid INT , f\_name VARCHAR(30), l\_name VARCHAR(30),deptid INT ,esalary INT , edate date)

LANGUAGE plpgsql

AS $$

BEGIN

INSERT INTO employees(eid,fname, lname, did, hire\_date)VALUES (empid, f\_name, l\_name, deptid,esalary,edate);

END;

$$;

CREATE OR REPLACE FUNCTION newemp(empid INT , f\_name VARCHAR(30), l\_name VARCHAR(30),deptid INT ,esalary int , edate date) RETURNS VOID AS $$

BEGIN

INSERT INTO employees(eid,fname, lname, did,salary, hire\_date)VALUES (empid, f\_name, l\_name, deptid,esalary,edate);

END;

$$ LANGUAGE plpgsql;