```
-- data for Assignment3


DROP TABLE IF EXISTS person;

DROP TABLE IF EXISTS knows;

DROP TABLE IF EXISTS company;

DROP TABLE IF EXISTS worksfor;

DROP TABLE IF EXISTS jobskill;

DROP TABLE IF EXISTS personskill;

create table person (pid  integer,
                    name text,
                    city text,
                    birthYear integer,
              primary key(pid));

insert into person values
 (1,'Nick','NewYork',1990),
 (2,'Deepa','Indianapolis',1985),
 (3,'Eric','NewYork',1990),
 (4,'Ryan','Indianapolis',1995),
 (5,'Hasan','Indianapolis',1990),
 (6,'Arif','Indianapolis',1980),
 (7,'Ryan','Chicago',1980),
 (8,'Jean','SanFransisco',2000),
 (9,'Aya','SanFransisco',1985),
 (10,'Lisa','NewYork',2000),
 (11,'Arif','Chicago',1990),
 (12,'Deepa','Bloomington',1990),
 (13,'Nick','SanFransisco',1980),
 (14,'Ryan','Indianapolis',1990),
 (15,'Nick','Indianapolis',1990),
 (16,'Anna','Chicago',1980),
 (17,'Lisa','Bloomington',1990),
 (18,'Ryan','Bloomington',1995),
 (19,'Lisa','Chicago',1980),
 (20,'Danielle','Indianapolis',1985),
 (21,'Eric','Chicago',1980),
 (22,'Anna','Indianapolis',1985),
  (23,'Chris','Bloomington',1990),
  (24,'Aya','NewYork',1995),
  (25,'Arif','SanFransisco',1990),
  (26,'Anna','Bloomington',2000),
  (27,'Latha','SanFransisco',2000),
  (28,'Eric','Bloomington',2000),
  (29,'Linda','Bloomington',1990),
  (30,'Aya','NewYork',1995);


 create table knows (pid1 integer,
                    pid2 integer,
                    primary key(pid1, pid2),
                    foreign key(pid1) references person(pid),
                    foreign key(pid2) references person(pid));

 insert into knows values
  (5,22),
  (15,28),
  (10,27),
  (11,27),
  (13,14),
  (11,14),
  (5,28),
  (1,26),
  (18,24),
  (24,5),
  (6,26),
  (15,7),
  (15,25),
  (19,27),
  (10,5),
  (11,19),
  (20,22),
  (27,23),
  (24,29),
  (4,10),
  (26,12),
  (13,15),
  (19,4),
  (20,10),
  (10,6),
  (1,7),
  (17,23),
  (9,26),
 (3,10),
 (21,29),
 (27,15),
 (12,13),
 (16,3),
 (14,24),
 (14,28),
 (12,4),
 (15,8),
 (4,28),
 (18,11),
 (12,16),
 (30,12),
 (4,9),
 (4,8),
 (29,13),
 (29,20),
 (24,18),
 (16,13),
 (30,17),
 (23,22),
 (7,16),
 (29,22),
 (26,3),
 (28,30),
 (25,10),
 (3,22),
 (22,21),
 (30,3),
 (1,20),
 (19,11),
 (29,15),
 (13,30),
 (11,12),
 (1,5),
 (13,18),
 (24,19),
 (30,10),
 (4,12),
 (24,11),
 (18,22),
 (3,2),
 (4,3),
 (12,23),
 (25,24),
  (17,20),
  (28,10),
  (8,17),
  (15,13),
  (1,9),
  (6,18),
  (3,4),
  (4,19),
  (24,23),
  (27,3),
  (12,5),
  (12,2),
  (26,22),
  (30,15),
  (20,13),
  (28,14),
  (14,5),
  (1,10),
  (7,9),
  (27,22),
  (12,11),
  (16,20),
  (12,3),
  (17,7),
  (2,14),
  (18,25),
  (16,24);


 create table company(cname text,
                     city  text,
                     primary key(cname, city));


 insert into company values
  ('Amazon','NewYork'),
  ('IBM','NewYork'),
  ('Amazon','Indianapolis'),
  ('Amazon','Bloomington'),
  ('Intel','NewYork'),
  ('Netflix','Indianapolis'),
  ('Yahoo','Indianapolis'),
  ('Google','Bloomington'),
  ('Apple','Indianapolis'),
  ('Hulu','Chicago'),
 ('Hulu','NewYork'),
 ('Yahoo','Chicago'),
 ('Intel','Bloomington'),
 ('Google','Chicago'),
 ('Zoom','Chicago'),
 ('Yahoo','NewYork'),
 ('Yahoo','Bloomington'),
 ('Netflix','Bloomington'),
 ('Microsoft','Chicago'),
 ('Netflix','NewYork'),
 ('Microsoft','Indianapolis'),
 ('Zoom','SanFransisco'),
 ('Netflix','SanFrancisco'),
 ('Yahoo','SanFrancisco'),
 ('IBM','SanFrancisco');



create table worksfor(pid    integer,
                      cname  text,
                      salary integer,
                      primary key(pid));


insert into worksfor values
 (1,'IBM',60000),
 (2,'Hulu',50000),
 (3,'Amazon',45000),
 (4,'Microsoft',60000),
 (5,'Amazon',40000),
 (6,'IBM',50000),
 (7,'IBM',50000),
 (8,'Netflix',45000),
 (9,'Yahoo',50000),
 (10,'Hulu',40000),
 (11,'Apple',40000),
 (12,'Netflix',55000),
 (13,'Apple',40000),
 (14,'IBM',50000),
 (15,'IBM',40000),
 (16,'Apple',55000),
 (17,'Google',45000),
 (18,'Amazon',45000),
 (19,'Zoom',45000),
 (20,'Microsoft',55000),
  (21,'Intel',55000),
  (22,'IBM',40000),
  (23,'Apple',40000),
  (24,'Google',45000),
  (25,'Hulu',50000),
  (26,'Intel',55000),
  (27,'Intel',50000),
  (28,'Intel',50000),
  (29,'Google',60000),
  (30,'Intel',60000);



 create table jobskill(skill text,
                      primary key(skill));

 insert into jobskill values
  ('Programming'),
  ('Databases'),
  ('AI'),
  ('Networks'),
  ('Mathematics');


 create table personskill(pid integer,
                         skill text,
                         primary key(pid, skill),
                         foreign key(pid) references person(pid),
                         foreign key(skill) references
 jobskill(skill));

 insert into personskill values
  (27,'Programming'),
  (18,'Mathematics'),
  (10,'AI'),
  (29,'Networks'),
  (23,'AI'),
  (4,'AI'),
  (1,'Databases'),
  (10,'Networks'),
  (9,'Programming'),
  (13,'Networks'),
  (9,'AI'),
  (27,'Mathematics'),
  (20,'AI'),
 (29,'Databases'),
 (5,'Programming'),
 (26,'Databases'),
 (1,'Networks'),
 (28,'AI'),
 (15,'Programming'),
 (16,'Mathematics'),
 (12,'Databases'),
 (15,'Databases'),
 (24,'Programming'),
 (14,'AI'),
 (25,'Networks'),
 (13,'AI'),
 (12,'Programming'),
 (22,'Programming'),
 (7,'Mathematics'),
 (10,'Programming'),
 (16,'Databases'),
 (19,'Programming'),
 (7,'Programming'),
 (22,'AI'),
 (5,'Databases'),
 (2,'Mathematics'),
 (14,'Programming'),
 (26,'Networks'),
 (19,'Networks'),
 (21,'Programming'),
 (14,'Mathematics'),
 (19,'AI'),
 (2,'Networks'),
 (8,'Databases'),
 (13,'Mathematics'),
 (29,'Programming'),
 (3,'AI'),
 (16,'Networks'),
 (5,'Networks'),
 (17,'AI'),
 (24,'Databases'),
 (2,'Databases'),
 (27,'Networks'),
 (28,'Databases'),
 (30,'Databases'),
 (4,'Networks'),
 (6,'Networks'),
 (17,'Networks'),
  (23,'Programming'),
  (20,'Programming');

 TABLE person;

 TABLE knows;

 TABLE company;

 TABLE worksfor;

 TABLE jobskill;

 table personskill;

 \qecho "Question 1"

 \qecho "Question 1.1"

 ---For question 1
 CREATE TABLE P(coefficient integer, degree integer);

 INSERT INTO P VALUES
 (3, 4),
  (4, 3),
 (2,2),
 (-5, 1),
 (5, 0);

 CREATE TABLE Q(coefficient integer, degree integer);

 INSERT INTO Q VALUES
 (1,5),
 (2, 4),
 (10, 3),
 (-1, 1),
 (9, 0);

 TABLE P;
 TABLE Q;

 create or replace function multiplicationPandQ()
 returns table(coefficient bigint, degree integer) as
 $$
 SELECT SUM(p.coefficient*q.coefficient), (q.degree+p.degree) FROM P
```

```
p, Q q GROUP BY q.degree + p.degree ORDER BY q.degree + p.degree
DESC;
$$ LANGUAGE SQL;

SELECT * FROM  multiplicationPandQ();

\qecho "Question 1.2"

---- For Question 2
CREATE TABLE X(index integer, value integer);

INSERT INTO X VALUES
(1, -8),
(2, -3),
(3, 4),
(4, 9);

CREATE TABLE Y(index integer, value integer);

INSERT INTO Y VALUES
(1, 3),
(2, -1),
(3, 9),
(4, -3);

TABLE X;
TABLE Y;

create or replace function dotProductXandY() returns bigint as
$$
SELECT SUM(x.value * y.value) FROM X x, Y y WHERE x.index =
y.index;
$$ LANGUAGE SQL;

SELECT * FROM dotProductXandY();


\qecho "Question 2"

/* Formulate the following queries in SQL.
You should use aggregate functions to solve these queries.
You can use views, including temporary views as well as
parameterized views defined by user-defined functions that return
relations (i.e., tables). */
```

```
\qecho "Question 2.3"

-- Find the pid and name of each person who lives in  and who
-- knows at least one person who has at least 3 job skills.


CREATE VIEW min3skills AS
SELECT DISTINCT p.pid, p.name FROM person p, person p2, knows k,
personskill ps WHERE p.pid = k.pid1 AND p2.pid = k.pid2 AND
p.city = 'Chicago' AND p2.pid = ps.pid AND (SELECT COUNT(skill)
FROM personskill) >= 3 ORDER BY pid ASC;

SELECT * FROM min3skills;


\qecho "Question 2.4"

-- Find the pid and name of each person who has all but four job
skills. I.e.,
-- such a person lacks precisely four job skills from the
possible job skills that
-- are stored in the relation jobSkill and lives in
'Indianapolis'.

CREATE VIEW skills AS

SELECT DISTINCT p.pid, p.name FROM person p WHERE(SELECT COUNT(1)
FROM (SELECT skill FROM jobskill EXCEPT (SELECT skill FROM
personskill.pid)) q) = 4;


\qecho "3 Queries with quantifiers"

\qecho "Question 3.5"


CREATE VIEW p1Knowsp2 AS
SELECT DISTINCT p.pid, p.name from  person p, person p2, knows k,
worksfor w WHERE p.pid = k.pid1 AND k.pid2 = p2.pid AND k.pid2 =
w.pid AND w.cname = 'Apple' AND w.salary < 60000;

CREATE VIEW birthYear AS
SELECT DISTINCT p.pid, p.name FROM person p WHERE p.birthYear <
2000;
```

```
SELECT p.pid, p.name FROM p1Knowsp2 p INTERSECT SELECT o.pid,
o.name FROM birthYear o;


\qecho "Question 3.6"

SELECT DISTINCT w.cname FROM worksfor w GROUP BY w.cname HAVING
MAX(w.salary) < 50000;


\qecho "Question 4.7"

-- I think I got half of it but I just don't know how to connect
them together with the quantifiers to where it only outputs the
company names that have even number of employees

CREATE VIEW evenSalaryCname AS
SELECT COUNT(pid) % 2 = 0, cname from worksfor WHERE salary <=
60000 GROUP BY cname;
SELECT * FROM evenSalaryCname;

CREATE VIEW oddSalaryCname AS
SELECT COUNT(pid) % 2 <> 0, cname FROM worksfor WHERE salary <=
60000 GROUP BY cname;
SELECT * FROM oddSalaryCname;

\qecho "Question 4.8"

CREATE VIEW pKnowsp AS
SELECT DISTINCT p.pid FROM person p, person p2, knows k WHERE
p.pid = k.pid1 AND p2.pid = k.pid2;

CREATE OR REPLACE FUNCTION twoJS(pid integer) RETURNS TABLE(pid
integer) AS
$$
SELECT DISTINCT w.pid FROM personskill w, knows k WHERE k.pid2 =
w.pid AND w.pid = pid AND (SELECT COUNT(skill) FROM personskill)
>= 2;
$$ language sql;

SELECT p.pid, p.name FROM person p WHERE EXISTS (SELECT pid FROM
pKnowsp WHERE pid NOT IN (SELECT pid FROM twoJS(pid))) GROUP BY
p.pid;
```

```
\qecho "Question 4.9"

CREATE OR REPLACE FUNCTION person1(pid integer) RETURNS TABLE(pid
integer) AS
$$
SELECT DISTINCT p.pid FROM person p, person p2, knows k WHERE
p.pid = k.pid1 AND p2.pid = k.pid2 AND (SELECT COUNT(k.pid2) FROM
knows k) >= 1;
$$ LANGUAGE SQL;

CREATE OR REPLACE FUNCTION person2(pid integer) RETURNS TABLE(pid
integer) AS
$$
SELECT DISTINCT p.pid FROM person p, person p2, knows k WHERE
p.pid = k.pid1 AND p2.pid = k.pid2 AND (SELECT COUNT(k.pid2) FROM
knows k) >= 1;
$$ LANGUAGE SQL;

SELECT p.pid, p2.pid FROM person p, person p2 WHERE EXISTS
(SELECT pid FROM person1(p.pid) WHERE pid IN (SELECT pid FROM
person2(p.pid) WHERE (SELECT COUNT(person1(p.pid)) FROM
person1(pid) WHERE (SELECT COUNT(person2(p.pid)) FROM
person2(p.pid)))));
```