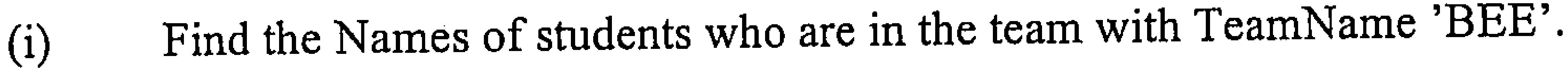
More Review Questions

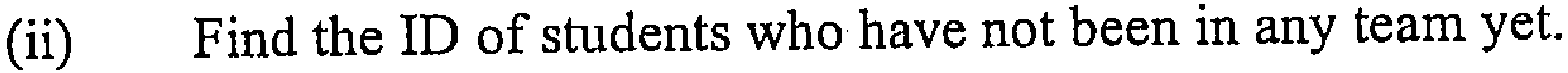
RA Review

- Student (ID, Name, Deparment)
- ProjectTeam(TeamName,ID)
- Midterm(ID, Grade)



(i) Find the Names of students who are in the team with TeamName 'BEE'.

$$\pi_{name} \left(\sigma_{TeamName='Bee'} (Project\ Team) \bowtie Student \right)$$



(ii) Find the ID of students who have not been in any team yet.

$$\pi_{ID} (Student) - \pi_{ID} (Project\ Team)$$

RA Review

- Student (ID, Name, Deparment)
- ProjectTeam(TeamName,ID)
- Midterm(ID, Grade)

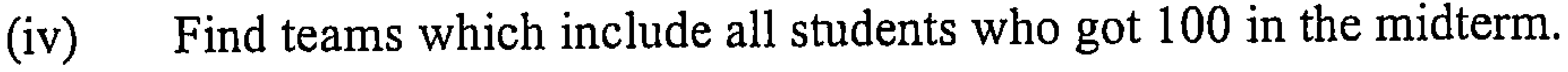
Find the TeamNames of teams whose member students are all from Department ’CS’

$$\rho(\text{UnQualifiedTeams},\ \pi_{\text{TeamName}}(\sigma_{\text{Department} \neq CS}(\text{ProjectTeam} \bowtie \text{Student})))$$

$$\pi_{\text{TeamName}}(\text{Project Team}) - \text{Unqualified Teams}.$$

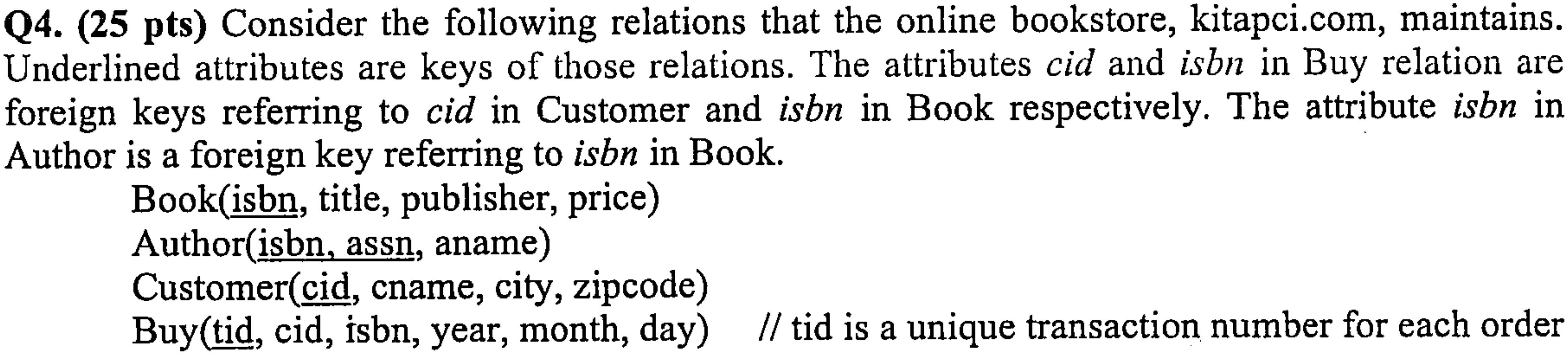
RA Review

- Student (ID, Name, Deparment)
- ProjectTeam(TeamName,ID)
- Midterm(ID, Grade)


(iv) Find teams which include all students who got 100 in the midterm.

$$\text{Project Team} \,/\, \pi_{ID}(\sigma_{grade=100}\ \text{Midterm})$$

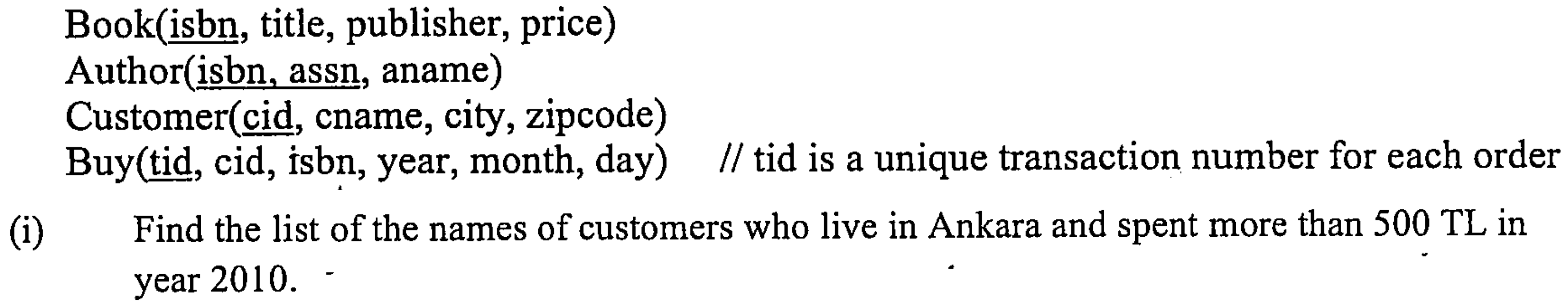
SQL Review



Q4. (25 pts) Consider the following relations that the online bookstore, kitapci.com, maintains. Underlined attributes are keys of those relations. The attributes *cid* and *isbn* in Buy relation are foreign keys referring to *cid* in Customer and *isbn* in Book respectively. The attribute *isbn* in Author is a foreign key referring to *isbn* in Book.

Book(isbn, title, publisher, price)
Author(isbn, assn, aname)
Customer(cid, cname, city, zipcode)
Buy(tid, cid, isbn, year, month, day)   // tid is a unique transaction number for each order

SQL Review

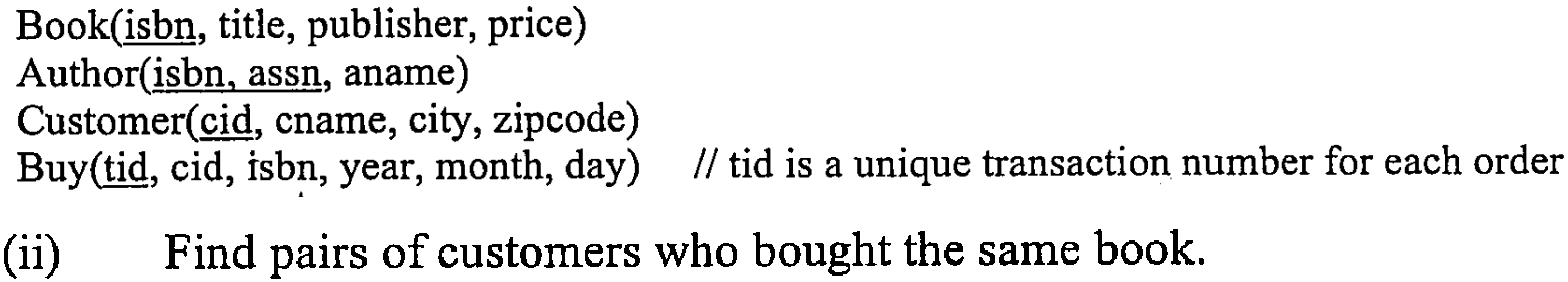


Book(isbn, title, publisher, price)
Author(isbn, assn, aname)
Customer(cid, cname, city, zipcode)
Buy(tid, cid, isbn, year, month, day) // tid is a unique transaction number for each order

(i) Find the list of the names of customers who live in Ankara and spent more than 500 TL in year 2010.

```
SELECT      C.cname
FROM        Customer C, Buy B, Book K
WHERE       C.cid = B.cid AND K.isbn = B.isbn
            AND C.city = ‘Ankara’ AND B.year = 2010
GROUP BY C.cid, C.cname
HAVING      SUM(price) > 500
```

SQL Review

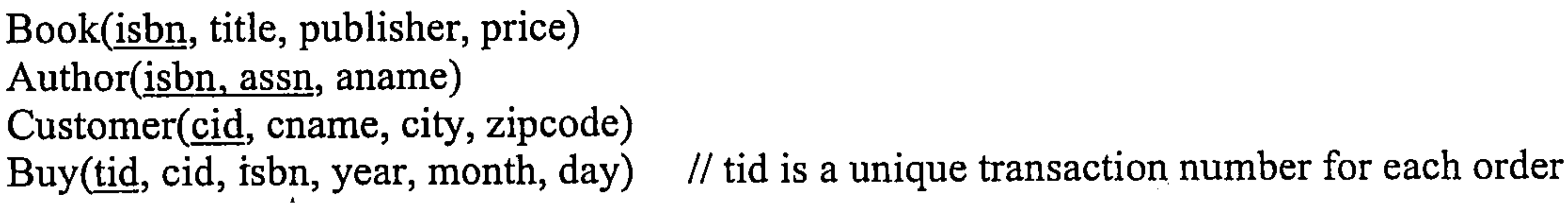


Book(isbn, title, publisher, price)
Author(isbn, assn, aname)
Customer(cid, cname, city, zipcode)
Buy(tid, cid, isbn, year, month, day)   // tid is a unique transaction number for each order

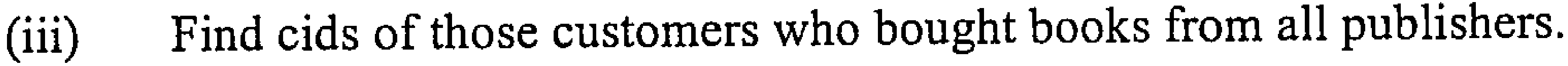
(ii) Find pairs of customers who bought the same book.

```
SELECT    B1.cid, B2.cid
FROM      Buy B1, Buy B2
WHERE     B1.isbn = B2.isbn AND  B1.cid < B2.cid
```

SQL Review



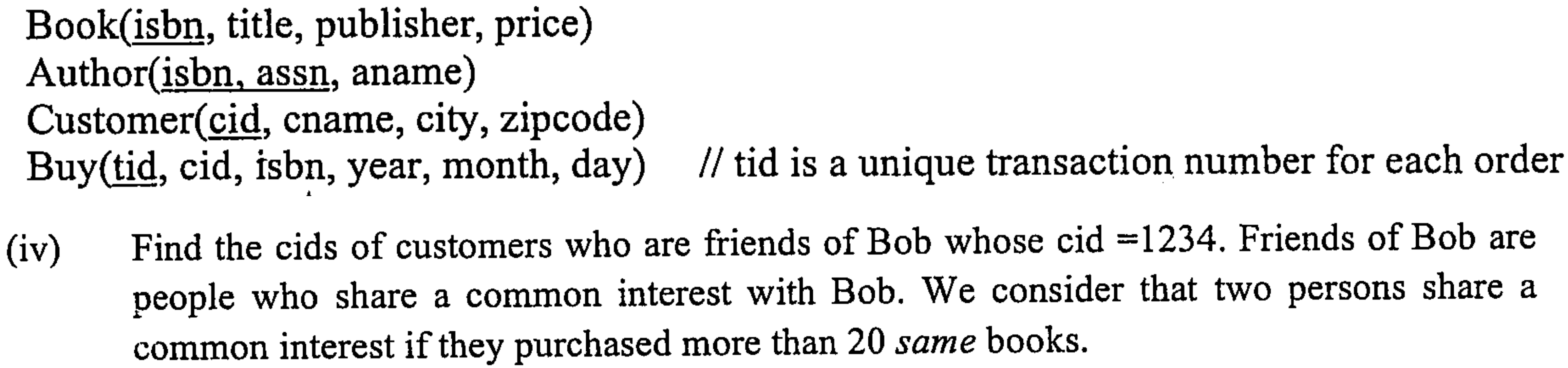
Book(isbn, title, publisher, price)
Author(isbn, assn, aname)
Customer(cid, cname, city, zipcode)
Buy(tid, cid, isbn, year, month, day)   // tid is a unique transaction number for each order



(iii) Find cids of those customers who bought books from all publishers.

```
SELECT    C.cid
FROM      Customers C
WHERE     NOT EXISTS (
                    (SELECT Publisher FROM Book)
                    EXCEPT
                    (SELECT Publisher
                     FROM Book K, Buy B
                     WHERE K.isbn = B.isbn AND B.cid = C.cid) )
```

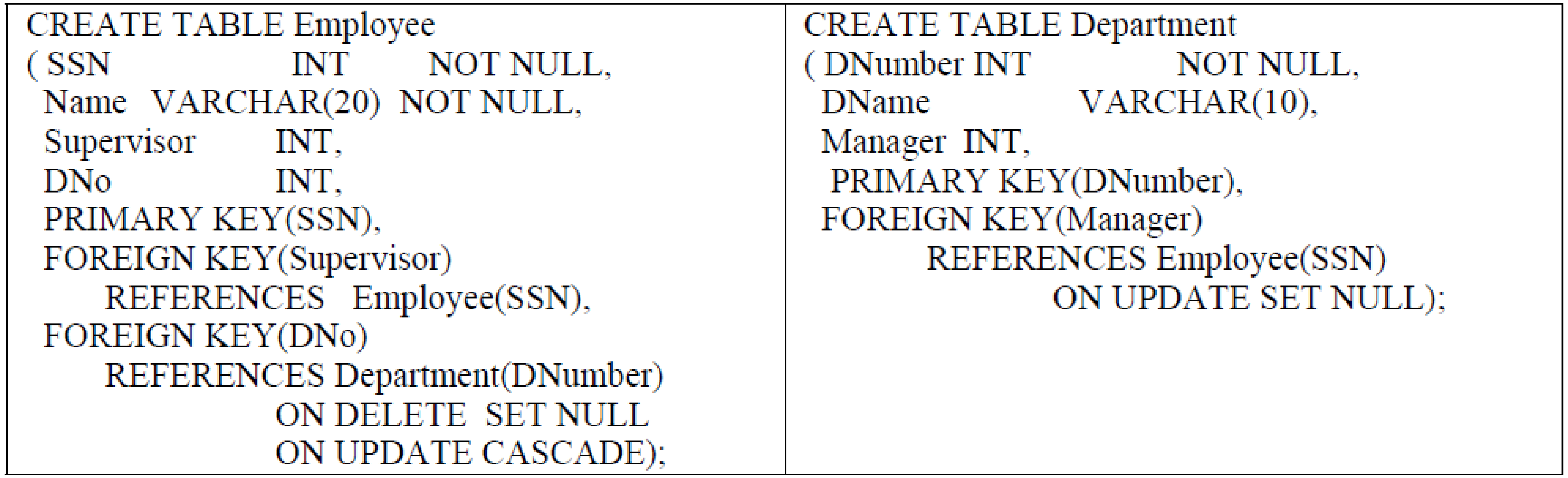
SQL Review



Book(isbn, title, publisher, price)
Author(isbn, assn, aname)
Customer(cid, cname, city, zipcode)
Buy(tid, cid, isbn, year, month, day) // tid is a unique transaction number for each order

(iv) Find the cids of customers who are friends of Bob whose cid =1234. Friends of Bob are people who share a common interest with Bob. We consider that two persons share a common interest if they purchased more than 20 *same* books.

```
SELECT    B2.cid
FROM      Buy B1, Buy B2
WHERE     B1.cid = 1234 AND B1.isbn = B2.isbn AND B1.cid <> B2.cid
GROUP BY B2.cid
HAVING    COUNT(*) > 20
```



```
CREATE TABLE Employee                    CREATE TABLE Department
( SSN             INT       NOT NULL,    ( DNumber INT            NOT NULL,
  Name  VARCHAR(20)  NOT NULL,             DName           VARCHAR(10),
  Supervisor      INT,                     Manager  INT,
  DNo             INT,                      PRIMARY KEY(DNumber),
  PRIMARY KEY(SSN),                        FOREIGN KEY(Manager)
  FOREIGN KEY(Supervisor)                          REFERENCES Employee(SSN)
       REFERENCES   Employee(SSN),                        ON UPDATE SET NULL);
  FOREIGN KEY(DNo)
       REFERENCES Department(DNumber)
                      ON DELETE  SET NULL
                      ON UPDATE CASCADE);
```

a. **INSERT INTO Employee VALUES (12, “Can”, NULL, NULL);**

b. **INSERT INTO Department VALUES (2, “Eng”, 12);**

c. **INSERT INTO Employee VALUES (15, “Oya”, 12, 2);**

d. **INSERT INTO Department VALUES (5, “Admin”, 15);**

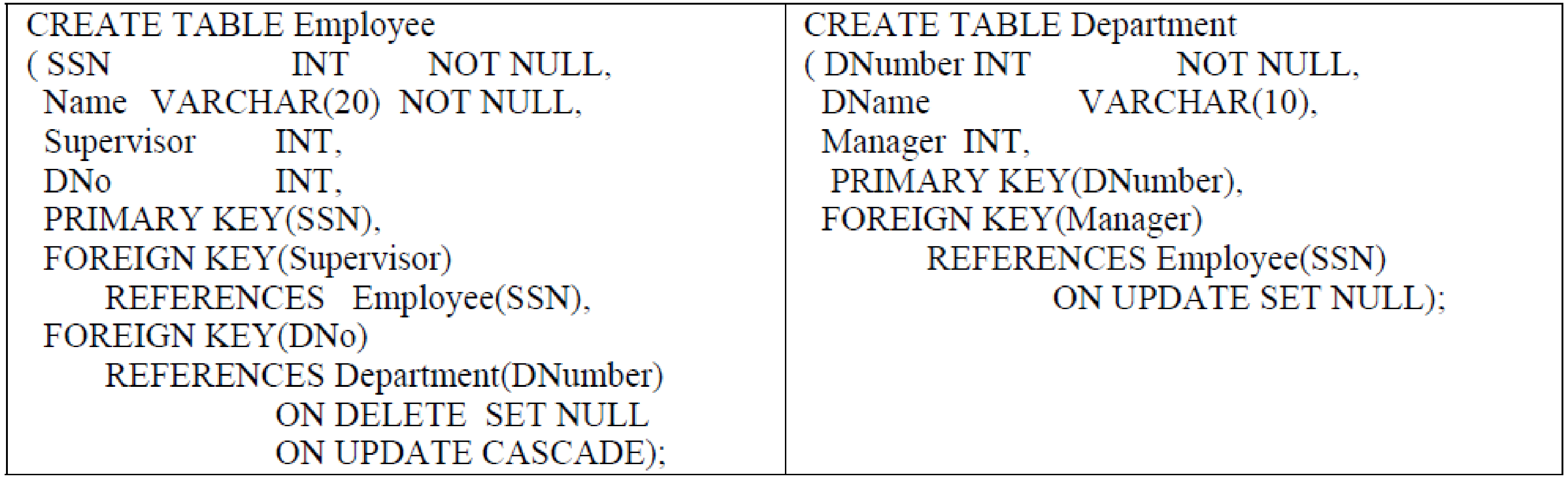
e. **INSERT INTO Employee VALUES (12, “Hikmet”, 12, 5);** **REJECTED: PK constraint**

f. **INSERT INTO Employee VALUES (14, “Oya”, 12, 5);**

g. **INSERT INTO Employee VALUES (16, NULL, 12, 2);** **REJECTED: NOT NULL constraint**

SSN	Name	Supervisor	DNo
12	Can	NULL	NULL
15	Oya	12	2
14	Oya	12	5

DNumber	DName	Manager
2	Eng	12
5	Admin	15



```
CREATE TABLE Employee
( SSN              INT       NOT NULL,
  Name  VARCHAR(20)  NOT NULL,
  Supervisor       INT,
  DNo              INT,
  PRIMARY KEY(SSN),
  FOREIGN KEY(Supervisor)
       REFERENCES   Employee(SSN),
  FOREIGN KEY(DNo)
       REFERENCES Department(DNumber)
                    ON DELETE  SET NULL
                    ON UPDATE CASCADE);
```

```
CREATE TABLE Department
( DNumber INT              NOT NULL,
  DName              VARCHAR(10),
  Manager  INT,
   PRIMARY KEY(DNumber),
  FOREIGN KEY(Manager)
          REFERENCES Employee(SSN)
                    ON UPDATE SET NULL);
```

h. **UPDATE Employee SET SSN = 18 WHERE SSN = 15;**

SSN	Name	Supervisor	DNo
12	Can	NULL	NULL
18	Oya	12	NULL
14	Oya	12	6

i. **UPDATE Department SET DNumber = DNumber+1 WHERE DNumber = 5;**

j. **DELETE FROM Department WHERE DNumber = 2;**

DNumber	DName	Manager
DNumber	DName	Manager
6	Admin	NULL

k. **DELETE FROM Employee WHERE SSN = 12;**

REJECTED: ON DELETE NO ACTION for Supervisor field