Homework 2 Solutions

Discussion Session 5

QUESTION 1 Solutions

(RA, RC and SQL queries corresponding to a single question are written together)

Question 1

List the ISBN of all books written by “Frank Herbert”

Q1-SQL

List the ISBN of all books written by “Frank Herbert”

```
SELECT E.isbn
FROM editions E, books B, authors A
WHERE B.author_id=A.author_id AND E.
book_id=B.book_id AND A.first_name='Frank'
AND A.last_name='Herbert';
```

Q1-RA

List the ISBN of all books written by "Frank Herbert"

$\Pi_{\text{editions.isbn}}(\sigma_{\text{authors.first\_name='Frank'} \wedge \text{authors.last\_name='Herbert'}}(\text{editions} \bowtie \text{authors} \bowtie \text{books}))$

Q1-RC

List the ISBN of all books written by "Frank Herbert"

{T | ∃ A E authors, **∃ B** E books **∃ E** E Editions

(A.author_id=B.author_id

^ E.book_id=B.book_id

^ A.first_name='Frank' ^

A.last_name= 'Herbert' ^

T.isbn=E.isbn)}

Q2

- List last name and first name of authors who have written both Short Story and Horror books.
- Be careful here. In general, there could be two different authors with the same name, one who has written a horror book and another who has written short stories.

Q2-SQL

List last name and first name of authors who have written both Short Story and Horror books.

```
SELECT A.first_name, A.last_name FROM
(
    SELECT A2.author_id, A2.first_name, A2.last_name
    FROM Authors A2, Books B2, Subjects S2 WHERE
    S2.subject = ‘Horror’ AND S2.subject_id = B2.subject_id
    AND B2.author_id = A2.author_id

    INTERSECT

    SELECT A3.author_id, A3.first_name, A3.last_name
    FROM Authors A3, Books B3, Subjects S3 WHERE
    S3.subject = ‘Short Story’ AND S3.subject_id = B3.subject_id
    AND B3.author_id = A3.author_id
) A;
```

Q2-SQL (Alternative)

List last name and first name of authors who have written both Short Story and Horror books.

```
SELECT A1.first_name, A1.last_name
    FROM Authors A1, Books B1, Subjects S1, Books B2, Subjects S2
WHERE
    S1.subject = ‘Short Story’ AND B1.subject_id = S1.subject_id AND
B1.author_id = A1.author_id AND S2.subject = ‘Horror’ AND
S2.subject_id = B2.subject_id AND B2.author_id = A1.author_id
```

Q2-SQL (Alternative)

List last name and first name of authors who have written both Short Story and Horror books.

```
CREATE VIEW sshauthorid AS
SELECT A.author_id
FROM authors A, books B, subjects S
WHERE A.author_id=B.author_id AND B.subject_id=S.subject_id AND S.subject='Short Story'
INTERSECT
SELECT A.author_id
FROM authors A, books B, subjects S
WHERE A.author_id=B.author_id AND B.subject_id=S.subject_id AND S.subject='Horror';

SELECT A.last_name, A.first_name
FROM authors A, sshauthorid T
WHERE A.author_id=T.author_id;

DROP VIEW sshauthorid;
```

Q2-RA

List last name and first name of authors who have written both Short Story and Horror books.

$$\pi_{last\_name,first\_name}(\pi_{id,last\_name,first\_name}$$
$$\sigma_{subject='ShortStory'}(Authors \bowtie Subjects \bowtie Books)$$
$$\cap$$
$$\pi_{id,last\_name,first\_name}$$
$$\sigma_{subject='Horror'}(Authors \bowtie Subjects \bowtie Books))$$

Q3

- List titles, subjects, author’s id, author's last name, and author's first name of all books by authors who have written Short Story book(s)

- Note that this may require a nested query. The answer can include non-Short Story books

- You can also use views. But DROP any views at the end of your query. Using a single query is likely to be more efficient in practice.

Q3-SQL (with View)

List titles, subjects, author's id, author's last name, and author's firs name of all books by authors who have written Short Story book(s)

```
CREATE VIEW ShortStoryAuthors AS
SELECT A.author_id, A.last_name, A.first_name
FROM Authors A, Books B, Subjects S
WHERE A.author_id = B.author_id AND B.subject_id = S.subject_id
AND S.subject = ‘Short Story’;

SELECT B.title, H.last_name, H.first_name, S.subject, H.author_id
FROM ShortStoryAuthors H, Books B, Subjects S
WHERE H.author_id = B.author_id AND B.subject_id = S.subject_id;
```

Q3-SQL (without View)

List titles, subjects, author’s id, author's last name, and author's firs name of all books by authors who have written Short Story book(s)

```
SELECT DISTINCT B.title, S.subject, A.author_id, A.last_name, A.
first_name
FROM books B, authors A, subjects S
WHERE B.author_id=A.author_id AND B.subject_id=S.subject_id AND
B.author_id IN
(SELECT B1.author_id
FROM books B1, subjects S
WHERE B1.subject_id=S.subject_id AND S.subject='Short Story');
```

Q3-SQL (without View)

List titles, subjects, author’s id, author's last name, and author's firs name of all books by authors who have written Short Story book(s)

```
SELECT B.title, H.last_name, H.first_name, S.subject, H.author_id
FROM
    (SELECT A.author_id, A.first_name, A.last_name
    FROM Authors A, Books B, Subjects S
    WHERE A.author_id = B.author_id
    AND B.subject_id = S.subject_id
    AND S.subject = ‘Short Story’) H,
    Books B, Subjects S
WHERE H.author_id = B.author_id AND B.subject_id = S.subject_id;
```

Q3-RA

List titles, subjects, author’s id, author's last name, and author's firs name of all books by authors who have written Short Story book(s)

$\rho(\text{ShortStoryAuthors},\ \Pi_{\text{author.author\_id,author.last\_name,author.first\_name}}(\sigma_{\text{subjects.subject='Short Story''}}(\text{subjects} \bowtie \text{authors} \bowtie \text{books})))$

$\Pi_{\text{last\_name,first\_name,author\_id,book.title,subject}}(\text{ShortStoryAuthors} \bowtie \text{books} \bowtie \text{Subjects})$

Q4

- Find id, first name, and last names of authors who wrote books for all the subjects of books written by author 'Edgar Allen Poe'

Q4 - SQL

Find id, first name, and last names of authors who wrote books for all the subjects of books written by author ‘Edgar Allen Poe’

```
CREATE view edgar_subjects AS
SELECT B.subject_id from books B,authors A where
B.author_id=A.author_id AND
A.first_name='Edgar Allen' AND A.last_name='Poe';
```

```
select A.first_name,A.last_name
from Authors A
where not exists (select E.subject_id
                  from edgar_subjects E

                  except

                  select B.subject_id
                  from Books B
                  where B.author_id=A.author_id);
```

Q4 - RA

Find id, first name, and last names of authors who wrote books for all the subjects of books written by author 'Edgar Allen Poe'

$\rho(\text{Subjects\_Edgar}, \Pi_{\text{subject\_id}} (\sigma_{\text{author.first\_name} = \text{'Edgar Allen'}} (\text{subjects} \bowtie \text{authors} \bowtie \text{books})))$

$\Pi_{\text{first\_name, last\_name}} (\text{authors} \bowtie (\Pi_{\text{author\_id,subject\_id}} (\text{books})/\text{Subjects\_Edgar})$

Q5

- Find the name and id of all publishers who have published books for authors who have written more than one book, order by ascending publisher id;

Q5-SQL

Find the name and id of all publishers who have published books for authors who have written more than one book, order by ascending publisher id;

```
SELECT DISTINCT P.publisher_id, P.name
FROM publishers P, editions E, books B
WHERE P.publisher_id=E.publisher_id AND E.book_id=B.book_id
AND B.author_id IN
           (SELECT B1.author_id
            FROM books B1
            GROUP BY B1.author_id
            HAVING COUNT(*)>1)
ORDER BY P.publisher_id;
```

Q6 - SQL

Find the last name and first name of authors who haven't written any book

Q6

Find the last name and first name of authors who haven't written any book

```
SELECT A.last_name, A.first_name
FROM Authors A
WHERE A.author_id NOT IN ( SELECT B.author_id
FROM Books B);
```

Q7

Find id of authors who have written exactly 1 book. Name the column as id. Order the id in ascending order

Q7-SQL

Find id of authors who have written exactly 1 book. Name the column as id. Order the id in ascending order

```
SELECT A.author_id as id
FROM Books B
GROUP BY B.author_id
HAVING COUNT(*) = 1
ORDER BY B.author_id;
```

QUESTION 2 Solutions

Q1-SQL

Student (<u>sid</u>, name, major) : student id, name, and majors of students
Project (<u>pid</u>, ptitle) : project ID and title of projects
Course (<u>cid,</u> title) : course ID and title of courses
Member (<u>pid, sid</u>) : Relationship, Student sid is a member of project pid
Enrolled (<u>sid, cid</u>) : Relationship. Student sid is enrolled in course cid.

Find the name of all students who have a project partner who is enrolled in both ‘EECS484’ AND ‘EECS482’. For example, student A will be output if students A and B are members of the same project and B is enrolled in the above two courses.

Find the name of all students who have a project partner who is enrolled in both ‘EECS484’ AND ‘EECS482’. For example, student A will be output if students A and B are members of the same project and B is enrolled in the above two courses.

```
SELECT S1.name
FROM Student S1, Student S2, Member M1, Member M2, Enrolled E1,
Enrolled E2, Course C1, Course C2
WHERE
        S1.sid = M1.sid AND S2.sid = M2.sid AND M1.pid = M2.pid AND
S1.sid<>S2.sid      -- (partners)
        AND
        S2.sid = E1.sid and S2.sid = E2.sid AND E1.cid = C1.cid and E2.
cid = C2.cid
        AND
        C1.title = ‘EECS 484’ AND C2.title = ‘EECS 482’;
```

Key idea: Need to do self-joins on Members, Enrolled, and Courses.

Q2-SQL

Student (sid, name, major) : student id, name, and majors of students
Project (pid, ptitle) : project ID and title of projects
Course (cid, title) : course ID and title of courses
Member (pid, sid) : Relationship, Student sid is a member of project pid
Enrolled (sid, cid) : Relationship. Student sid is enrolled in course cid.

Find the student id (sid) and name of all students who are enrolled in a class with more than 50 ‘CS’ majors enrolled. For example, if a course has over 50 CS students, then all its students (whether CS or non-CS) will be included in the output. The result of the query should be output in increasing order by student id and each student must appear only once. Use of views or nested queries is OK for this part.

Find the student id (sid) and name of all students who ar enrolled in a class with more than 50 ‘CS’ majors enrolle

```
SELECT DISTINCT S1.sid, S1.name
FROM Student S1, Enrolled E1
WHERE S1.sid = E1.sid AND E1.cid
IN
(SELECT E.cid
FROM Enrolled E, Student S2
WHERE E.sid = S2.sid and S2.major = ‘CS’
GROUP BY E.cid
HAVING COUNT(*) > 50)
ORDER BY S.sid ASC;
```

Q3-SQL

Student (sid, name, major) : student id, name, and majors of students
Project (pid, ptitle) : project ID and title of projects
Course (cid, title) : course ID and title of courses
Member (pid, sid) : Relationship, Student sid is a member of project pid
Enrolled (sid, cid) : Relationship. Student sid is enrolled in course cid.

Define a VIEW that contains all student id pairs (sid1, sid2) with the following property: the students are members of the same project but are not enrolled in any common course; i.e., the student pair do not share any courses. List each pair of students only once. For example, if you list (23, 42), do not list (42, 23). Student ids can be assumed to be integers. Use of views or nested queries is OK for this part.

Define a VIEW that contains all student id pairs (sid1, sid2) with the following property: the students are members of the same project but are not enrolled in any common course; i.e., the student pair do not share any courses. List each pair of students only once.

```
CREATE VIEW StudentPairs(sid1,sid2) AS
(
      --student pairs who are partners
      (SELECT S1.sid, S2.sid
      FROM Student S1, Student S2, Member M1, Member M2
      WHERE S1.sid = M1.sid AND S2.sid = M2.sid AND M1.pid =
M2.pid AND S1.sid<S2.sid)
      MINUS
      --student pairs who share courses
      (SELECT S1.sid, S2.sid
      FROM
      Student S1, Student S2, Enrolled E1, Enrolled E2
      WHERE
      S1.sid = E1.sid AND S2.sid = E2.sid AND E1.cid = E2.cid  AND
S1.sid < S2.sid
));
```

- **Define a VIEW that contains all student id pairs (sid1, sid2) with the following property: the students are members of the same project but are not enrolled in any common course; i.e., the student pair do not share any courses. List each pair of students only once.**

```
CREATE VIEW pair AS
SELECT M1.sid, M2.sid
FROM Member M1, Member M2
WHERE M1.pid=M2.pid AND M1.sid<M2.sid
AND NOT EXISTS
    ( SELECT E1.cid
    FROM Enrolled E1
    WHERE E1.sid=M1.sid
    INTERSECT
    SELECT E2.cid
    FROM Enrolled E2
    WHERE E2.sid=M2.sid
    );
```