CS4350 Database Systems Spring2024 Test#1

**Name: Loc Nguyen**

Q1 (70) Given the following schema of a video rental business:

**VIDEO**= ( Title, RentalFee, Rating) // The Title is the primary key. The Rating is either ‘PG’ or ‘R’, or ‘PG13’.

**CUSTOMER**= ( FullName , Address, CreditCardN) //The FullName is the primary key

**RENTALTRANSCATION**= ( FullName, Title,  DateCheckOut, DateReturn,)  // If the video has not been returned, the DateReturn = Null. The FullName, Title and DateCheckOut are the primary key.

Assume all dates are before the date of today and you may use <, >, =, and <> to compare any two dates. Express the following queries by SQL statements using a minimum number of tables and operations.

1. List the Title and Rating of every video which has been rented by the same customer more than once

SELECT V.Title, V.Rating

FROM Video V

JOIN RentalTransaction T ON V.Title = T.Title

GROUP BY T.FullName, T.Title

HAVING COUNT(T.Title) > 1;

**VIDEO**= ( Title, RentalFee, Rating) // The Title is the primary key. The Rating is either ‘PG’ or ‘R’, or ‘PG13’.

**CUSTOMER**= ( FullName , Address, CreditCardN) //The FullName is the primary key

**RENTALTRANSCATION**= ( FullName, Title,  DateCheckOut, DateReturn,)  // If the video has not been returned, the DateReturn = Null. The FullName, Title and DateCheckOut are the primary key.

1. List the video titles with the lowest rental fee.

SELECT V.Title

FROM Video V

WHERE V.RentalFee = (SELECT MIN(RentalFee) FROM Video);

1. List the FullName of every customer who never rented any video.

SELECT C.FullName

FROM Customer C

WHERE C.FullName NOT IN (

SELECT FullName

FROM RentalTransaction

);

Explanation – Check if Customer name isn’t in the Transaction table, they never rented any video.

**VIDEO**= ( Title, RentalFee, NumberOfDays, Rating) // The Title is the primary key. The Rating is either ‘PG’ or ‘R’, or ‘PG13’.

**CUSTOMER**= ( FullName , Address, CreditCardN) //The FullName is the primary key

**RENTALTRANSCATION**= ( FullName, Title,  DateCheckOut, DateReturn, TotalNumberOf Days)  // If the video has not been returned, the DateReturn = Null. The FullName, Title , and DateCheckOut are the primary key.

1. List the name of each customer and Title of every video which was rented by the customer.

SELECT C.FullName, T.title

FROM Customer C   
JOIN RentalTransaction T ON C.Fullname = T.FullName

GROUP BY C.FullName;

1. List the titles of the movies which have been rented at least two times.

SELECT V.Title

FROM Video V

JOIN RentalTransaction T ON V.Title = T.Title

WHERE COUNT(T.Title) > 2;

**VIDEO**= ( Title, RentalFee, Rating) // The Title is the primary key. The Rating is either ‘PG’ or ‘R’, or ‘PG13’.

**CUSTOMER**= ( FullName , Address, CreditCardN) //The FullName is the primary key

**RENTALTRANSCATION**= ( FullName, Title,  DateCheckOut, DateReturn,)  // If the video has not been returned, the DateReturn = Null. The FullName, Title, and DateCheckOut are the primary key.

1. List the Fullname and Address of every customer who has only rented movies with rating ‘R’

SELECT C.FullName, C.Address

FROM Customer C

JOIN RentalTransaction T ON C.FullName = T.FullName

JOIN Video V ON T.Title = V.Title

WHERE V.Rating = ‘R’ AND T.FullName NOT IN (

SELECT R.FullName

FROM RentalTransaction R

JOIN Video V2 ON R.Title = V2.Title

WHERE V2.Rating <> ‘R’

);

Explanation: Check if Customer has rented R rated movies and make sure they are not in the sub-query for renting another movies that isn’t R rated.

1. List the title of every video which was rented by both ‘John Smith’ and ‘ Mike Lee’( at different times).

SELECT DISTINCT T1.Title

FROM RentalTransaction T1

JOIN RentalTransaction T2 ON T1.Fullname = T2.FullName

WHERE T1.Fullname = ‘John Smith’ AND T2.FullName = ‘Mike Lee’

**VIDEO**= ( Title, RentalFee, Rating) // The Title is the primary key. The Rating is either ‘PG’ or ‘R’, or ‘PG13’.

**CUSTOMER**= ( FullName , Address, CreditCardN) //The FullName is the primary key

**RENTALTRANSCATION**= ( FullName, Title,  DateCheckOut, DateReturn,)  // If the video has not been returned, the DateReturn = Null. The FullName, Title , and DateCheckOut are the primary key.

1. List the customer names who have rented videos and haven’t returned them yet

SELECT DISTINCT T.FullName

FROM RentalTransaction T

WHERE T.DateReturn IS NULL;

1. List the name of every customer who never rented any video with rating ’R’.

SELECT T.FullName

FROM RentalTransaction T   
WHERE T.FullName NOT IN (

SELECT T2.FullName

FROM RentalTransaction T2

JOIN Video V ON T2.Title = V.Title

WHERE V.Rating = ‘R’

);

Q2. (30 points) Consider the following relational schema, DDL statements and tables. Show the status of the above tables after each of the following operations.

1. Deleting the Employee whose ID=8 from the table Employee.

**Accepted. EmployeeID = 8 is deleted from Employee table and on DELETE for SupervisorID = 8, set to default of 9. For Project Table, set to NULL on DELETE**

CREATE TABLE EMPLOYEE (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50) NOT NULL,

SupervisorID INT DEFAULT 9,

FOREIGN KEY (SupervisorID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET DEFAULT,

DepartmentID INT. DEFAULT 6

FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

ON UPDATE SET CASCADE

);

CREATE TABLE PROJECT (

ProjectID INT PRIMARY KEY,

EmployeeID INT DEFAULT 9

FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET NULL

ON UPDATE SET DEFAULT);

CREATE TABLE DEPARTMENT(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(50)

);

EMPLOYEE

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | SupervisorID | DepartmentID |
| 6 | A | 7 | 6 |
| 7 | B | ~~8~~ **9** | 6 |
| 1 | C | 7 | 7 |
| ~~8~~ | ~~D~~ | ~~9~~ | ~~3~~ |
| 9 | E | NULL | 6 |

PROJECT

|  |  |
| --- | --- |
| ProjectID | EmployeeID |
| 1 | 6 |
| 2 | 7 |
| 3 | 1 |
| 4 | ~~8~~ **NULL** |
| 5 | 7 |
| 6 | 1 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartmentID | DepartmentName |
| 3 | X |
| 6 | Y |
| 7 | Z |

1. Inserting a new employee: 30, F, NULL,10 into the Table Employee. If it is rejected, explain.

**Rejected. There is no DepartmentID = 10 in the DEPARTMENT table. Table is unchanged.**

CREATE TABLE EMPLOYEE (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50) NOT NULL,

SupervisorID INT DEFAULT 9,

FOREIGN KEY (SupervisorID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET DEFAULT,

DepartmentID INT. DEFAULT 6

FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

ON UPDATE SET CASCADE

);

CREATE TABLE PROJECT (

ProjectID INT PRIMARY KEY,

EmployeeID INT DEFAULT 9

FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET NULL

ON UPDATE SET DEFAULT);

CREATE TABLE DEPARTMENT(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(50)

);

EMPLOYEE

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | SupervisorID | DepartmentID |
| 6 | A | 7 | 6 |
| 7 | B | 8 | 6 |
| 1 | C | 7 | 7 |
| 8 | D | 9 | 3 |
| 9 | E | NULL | 6 |

PROJECT

|  |  |
| --- | --- |
| ProjectID | EmployeeID |
| 1 | 6 |
| 2 | 7 |
| 3 | 1 |
| 4 | 8 |
| 5 | 7 |
| 6 | 1 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartmentID | DepartmentName |
| 3 | X |
| 6 | Y |
| 7 | Z |

1. Change the Employee ID= 8 to 10 in the table Employee.

**Rejected because EmployeeID = 8 appear in the column SupervisorID and not having override rule for update prevents the change. Table is unchanged**

CREATE TABLE EMPLOYEE (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50) NOT NULL,

SupervisorID INT DEFAULT 9,

FOREIGN KEY (SupervisorID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET DEFAULT,

DepartmentID INT. DEFAULT 6

FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

ON UPDATE SET CASCADE

);

CREATE TABLE PROJECT (

ProjectID INT PRIMARY KEY,

EmployeeID INT DEFAULT 9

FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET NULL

ON UPDATE SET DEFAULT);

CREATE TABLE DEPARTMENT(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(50)

);

EMPLOYEE

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | SupervisorID | DepartmentID |
| 6 | A | 7 | 6 |
| 7 | B | 8 | 6 |
| 1 | C | 7 | 7 |
| 8 | D | 9 | 3 |
| 9 | E | NULL | 6 |

PROJECT

|  |  |
| --- | --- |
| ProjectID | EmployeeID |
| 1 | 6 |
| 2 | 7 |
| 3 | 1 |
| 4 | 8 |
| 5 | 7 |
| 6 | 1 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartmentID | DepartmentName |
| 3 | X |
| 6 | Y |
| 7 | Z |

1. Changing the DepartmentID= 3 to 34 in the table DEPARTMENT. If it is rejected, explain.

**Accepted. In the Department table, DepartmentID changed from 3 to 34. In the Employee table, change is cascaded on update. Updated table below**

CREATE TABLE EMPLOYEE (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50) NOT NULL,

SupervisorID INT DEFAULT 9,

FOREIGN KEY (SupervisorID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET DEFAULT,

DepartmentID INT. DEFAULT 6

FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

ON UPDATE SET CASCADE

);

CREATE TABLE PROJECT (

ProjectID INT PRIMARY KEY,

EmployeeID INT DEFAULT 9

FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET NULL

ON UPDATE SET DEFAULT);

CREATE TABLE DEPARTMENT(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(50)

);

EMPLOYEE

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | SupervisorID | DepartmentID |
| 6 | A | 7 | 6 |
| 7 | B | 8 | 6 |
| 1 | C | 7 | 7 |
| 8 | D | 9 | ~~3~~ 34 |
| 9 | E | NULL | 6 |

PROJECT

|  |  |
| --- | --- |
| ProjectID | EmployeeID |
| 1 | 6 |
| 2 | 7 |
| 3 | 1 |
| 4 | 8 |
| 5 | 7 |
| 6 | 1 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartmentID | DepartmentName |
| ~~3~~ 34 | X |
| 6 | Y |
| 7 | Z |

1. Delete the employee whose ID= 6 from the table Employee. If it is rejected, explain.

**Accepted. EmployeeID = 6 is deleted from Employee table. In the Project table, EmployeeID = 6 is set to NULL on DELETE.**

CREATE TABLE EMPLOYEE (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50) NOT NULL,

SupervisorID INT DEFAULT 9,

FOREIGN KEY (SupervisorID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET DEFAULT,

DepartmentID INT. DEFAULT 6

FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

ON UPDATE SET CASCADE

);

CREATE TABLE PROJECT (

ProjectID INT PRIMARY KEY,

EmployeeID INT DEFAULT 9

FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET NULL

ON UPDATE SET DEFAULT);

CREATE TABLE DEPARTMENT(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(50)

);

EMPLOYEE

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | SupervisorID | DepartmentID |
| ~~6~~ | ~~A~~ | ~~7~~ | ~~6~~ |
| 7 | B | 8 | 6 |
| 1 | C | 7 | 7 |
| 8 | D | 9 | 3 |
| 9 | E | NULL | 6 |

PROJECT

|  |  |
| --- | --- |
| ProjectID | EmployeeID |
| 1 | ~~6~~ NULL |
| 2 | 7 |
| 3 | 1 |
| 4 | 8 |
| 5 | 7 |
| 6 | 1 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartmentID | DepartmentName |
| 3 | X |
| 6 | Y |
| 7 | Z |

1. Changing the. EmployeeID = 7 in the table Employee to 10. If it is rejected, explain.

**Rejected because EmployeeID = 7 appear in the column SupervisorID and not having override rule for update prevents the change. Table is unchanged**

ACCEPT because the EmployeeID=6 does not appear in the column SupervisorID  
So not having override rule for update does not prevent the change .In the table PROJECT , the  
EmployeeID=6 changes to 10

CREATE TABLE EMPLOYEE (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(50) NOT NULL,

SupervisorID INT DEFAULT 9,

FOREIGN KEY (SupervisorID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET DEFAULT,

DepartmentID INT. DEFAULT 6

FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

ON UPDATE SET CASCADE

);

CREATE TABLE PROJECT (

ProjectID INT PRIMARY KEY,

EmployeeID INT DEFAULT 9

FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE (EmployeeID)

ON DELETE SET NULL

ON UPDATE SET DEFAULT);

CREATE TABLE DEPARTMENT(

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(50)

);

EMPLOYEE

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | SupervisorID | DepartmentID |
| 6 | A | 7 | 6 |
| 7 | B | 8 | 6 |
| 1 | C | 7 | 7 |
| 8 | D | 9 | 3 |
| 9 | E | NULL | 6 |

PROJECT

|  |  |
| --- | --- |
| ProjectID | EmployeeID |
| 1 | 6 |
| 2 | 7 |
| 3 | 1 |
| 4 | 8 |
| 5 | 7 |
| 6 | 1 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartmentID | DepartmentName |
| 3 | X |
| 6 | Y |
| 7 | Z |