CSE4/560

Databases and Query Languages

Homework 2

Total Marks 100

Your Name: Challa Manasa Sai

Your email ID: manasasa@buffalo.edu

Your UB Person ID: 50356441

1.[15] What is a foreign key constraint? Why are such constraints important? What is referential integrity?

Ans:

|  |  |  |
| --- | --- | --- |
| OrderID | OrderNumber | PersonID |
| 1 | 77 | 346529 |
| 2 | 23 | 283654 |
| 3 | 64 | 187574 |

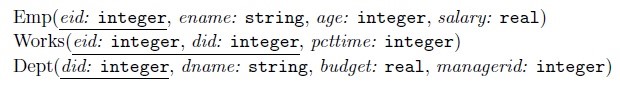
A foreign key is a field (or collection of fields) in one table, that refers to the [primary key](https://www.w3schools.com/sql/sql_primarykey.asp) in another table. The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table. A foreign key constraint specifies that the key can only contain values that are in the referenced primary key, and thus ensures the referential integrity of data that is joined on the two keys. Foreign key constraints are important because they provide safeguards for insuring the integrity of data. Users are alerted/thwarted when they try to do something that does not make sense. This can help minimize errors in application programs or in data-entry. The first advantage of the foreign key constraint is that the database constantly maintains referential integrity.  This means the database monitors the data inserted into the parent and child tables. One of the intangibles of the foreign key constraint is how much easier it is for DBA’s and database developers to determine how the database is designed.  When you use the ‘View Dependencies…’ option for either table, you will be greeted with a visible hierarchy showing how the tables are related. Another obvious advantage for using foreign key constraints is improved performance.  By including information on how tables are joined, SQL Server can easily determine how it’s going to retrieve data when using those joins. Referential integrity is a property of data stating that all its references are valid. In the context of relational databases, it requires that if a value of one attribute of a relation references a value of another attribute, then the referenced value must exist. For example, if we delete row number 15 in a primary table, we need to be sure that there’s no foreign key in any related table with the value of 15. We should only be able to delete a primary key if there are no associated rows. Otherwise, we would end up with an [orphaned record](https://database.guide/what-is-an-orphaned-record/).

|  |  |
| --- | --- |
| PersonID | Name |
| 346529 | Jack |
| 187574 | Mia |
| 283654 | Raj |

The "PersonID" column in the "Persons" table is the PRIMARY KEY in the "Persons" table.

The "PersonID" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.

2. [25] Given the following relational schema, you have already written the SQL statements required to create the preceding relations, including appropriate versions of all primary and foreign key integrity constraints. They are again added below for your reference.



CREATE TABLE Works ( eid INTEGER NOT NULL, did INTEGER NOT NULL, pettime INTEGER, PRIMARY KEY(eid, did),

UNIQUE (eid),

FOREIGN KEY (did) REFERENCES Dept

);

CREATE TABLE Dept ( did INTEGER NOT NULL, budget REAL, managerid INTEGER,

PRIMARY KEY(did),

FOREIGN KEY (managerid) REFERENCES Emp

);

1. Now modify the Dept relation in SQL so that every department is guaranteed to have a manager.

Ans:

ALTER TABLE Dept ALTER COLUMN managerid SET NOT NULL

1. Write an SQL statement to add John Doe as an employee with eid = 101, age = 32 andsalary = 15, 000

Ans:

INSERT

INTO Emp (eid, ename, age, salary)

VALUES (101, ’John Doe’, 32, 15000)

1. Suppose you have a view SeniorEmp defined as follows:

CREATE VIEW SeniorEmp (sname, sage, salary)

AS SELECT E.ename, E.age, E.salary

FROM Emp E

WHERE *E.age >* 50;

Explain what the system will do to process the following query:

SELECT S.sname

FROM SeniorEmp S

WHERE *S.salary >* 10000

Ans:

The system will do the following:  
SELECT S.name FROM ( SELECT E.ename AS name, E.age, E.salary FROM Emp E WHERE E.age > 50 ) AS S WHERE S.salary > 10000

1. Give an example of a view on Emp that could be automatically updated by updating Emp.

Ans:

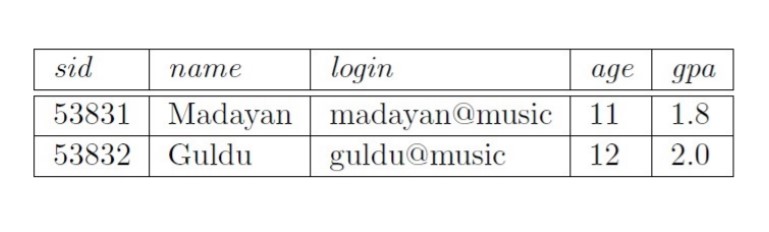
The following view on Emp can be updated automatically by updating Emp:  
CREATE VIEW SeniorEmp (eid, name, age, salary) AS SELECT E.eid, E.ename, E.age, E.salary FROM Emp E WHERE E.age > 50

1. Give an example of a view on Emp that would be impossible to update (automatically) and explain why your example presents the update problem that it does.

Ans:

The following view cannot be updated automatically because it is not clear which employee records will be affected by a given update:  
CREATE VIEW AvgSalaryByAge (age, avgSalary) AS SELECT E.eid, AVG (E.salary) FROM Emp E GROUP BY E.age

3.[10] Consider the SQL query whose answer is shown below.



1. Modify this query so that only the login column is included in the answer.

Ans:

SELECT S.login  
FROM Students S  
WHERE S.age < 18

1. If the clause WHERE *S.gpa*2 is added to the original query, what is the set of tuples in the answer?

Ans:

The answer tuple for Madayan is omitted then

4. [60] Consider the following relations: Student(snum: integer, sname: string, major: string, level: string, age: integer)

Class(name: string, meets at: string, room: string, fid: integer)

Enrolled(snum: integer, cname: string)

Faculty(fid: integer, fname: string, deptid: integer)

Write the following queries in SQL. No duplicates should be printed in any of the answers.

1. Find the names of all Juniors (level = JR) who are enrolled in a class taught by I;

Ans:

SELECT DISTINCT S.Sname  
FROM Student S, Class C, Enrolled E, Faculty F  
WHERE S.snum = E.snum AND E.cname = C.name AND C.fid = F.fid AND F.fname = ‘I’ AND S.level = ‘JR’

1. Find the age of the oldest student who is either a History major or enrolled in a coursetaught by I.

Ans:

SELECT MAX(S.age)  
FROM Student S  
WHERE (S.major = ‘History’)  
 OR S.snum IN (SELECT E.snum

FROM Class C, Enrolled E, Faculty F  
 WHERE E.cname = C.name AND C.fid = F.fid

AND F.fname = ‘I’ )

1. Find the names of all classes that either meet in room R128 or have five or more students enrolled.

Ans:

SELECT C.name  
FROM Class C  
WHERE C.room = ‘R128’  
 OR C.name IN (SELECT E.cname

FROM Enrolled E

GROUP BY E.cname  
 HAVING COUNT (\*) >= 5)

1. Find the names of all students who are enrolled in two classes that meet at the sametime.

Ans:

SELECT DISTINCT S.sname

FROM Student S

WHERE S.snum IN (SELECT E1.snum

FROM Enrolled E1, Enrolled E2, Class C1, Class C2

WHERE E1.snum = E2.snum AND E1.cname *<>* E2.cname

AND E1.cname = C1.name

AND E2.cname = C2.name AND C1.meets at = C2.meets at)

1. Find the names of faculty members who teach in every room in which some class is taught.

Ans:

SELECT DISTINCT F.fname

FROM Faculty F

WHERE NOT EXISTS (( SELECT \*

FROM Class C )

EXCEPT

(SELECTC1.room FROM Class C1

WHERE C1.fid = F.fid ))

1. Find the names of faculty members for whom the combined enrollment of the coursesthat they teach is less than five.

Ans:

SELECT DISTINCT F.fname

FROM Faculty F

WHERE 5 *>* (SELECT COUNT (E.snum)

FROM Class C, Enrolled E

WHERE C.name = E.cname

AND C.fid = F.fid)

1. For each level, print the level and the average age of students for that level.

Ans:

SELECT S.level, AVG(S.age)

FROM Student S

GROUP BY S.level

1. For all levels except JR, print the level and the average age of students for that level.

Ans:

SELECT S.level, AVG(S.age)

FROM Student S

WHERE S.level *<>* ‘JR’

GROUP BY S.level

1. For each faculty member that has taught classes only in room R128, print the facultymember’s name and the total number of classes she or he has taught.

Ans:

SELECT F.fname, COUNT(\*) AS CourseCount

FROM Faculty F, Class C

WHERE F.fid = C.fid

GROUP BY F.fid, F.fname

HAVING EVERY ( C.room = ‘R128’ )

1. Find the names of students enrolled in the maximum number of classes.

Ans:

SELECT DISTINCT S.sname

FROM Student S

WHERE S.snum IN (SELECT E.snum

FROM Enrolled E

GROUP BY E.snum

# HAVING COUNT (\*) *>*= ALL (SELECT COUNT (\*)

FROM Enrolled E2

GROUP BY E2.snum ))

1. Find the names of students not enrolled in any class.

Ans:

SELECT DISTINCT S.sname

FROM Student S

WHERE S.snum NOT IN (SELECT E.snum

FROM Enrolled E )

1. For each age value that appears in Students, find the level value that appears most often.For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

Ans:

SELECT S.age, S.level

FROM Student S

GROUP BY S.age, S.level,

HAVING S.level IN (SELECT S1.level

FROM Student S1

WHERE S1.age = S.age

GROUP BY S1.level, S1.age

# HAVING COUNT (\*) *>*= ALL (SELECT COUNT (\*)

FROM Student S2

WHERE s1.age = S2.age

GROUP BY S2.level, S2.age))