School : Computer Science
Institution : University of Windsor
Term : Fall 2021
Course : Comp-3150 (03-60-315-1) : Database Management Systems
Instructor : Dr. C. I. Ezeife
Assignment #2 : Total : 50 marks
Handed Out: Thurs. Sep. 28, 2023; Due Thurs. Oct. 26, 2023

Objective of Assignment: To test on knowledge and design of relational model constraints, relational database schemas, functional dependencies and normalization of relational databases.

Scope: Assignment covers materials from Chapters 5 and 14 of book discussed in class.

Electronic Assignment Submission: Done through http://brightspace.uwindsor.ca

Marking Scheme : The mark for each of the questions is indicated beside each question.

Academic Integrity Statement : Remember to submit only work that is yours and include the following confidentiality agreement and statement at the beginning of your assignment.

CONFIDENTIALITY AGREEMENT & STATEMENT OF HONESTY

I confirm that I will keep the content of this assignment/examination confidential.
I confirm that I have not received any unauthorized assistance in preparing for or doing this assignment/examination. I confirm knowing that a mark of 0 may be assigned for copied work.

Spondon Sayeed	Spondon Sayeed
Student Signature	Student Name (please print)
110101278	2023-10-25
Student I.D. Number	Date

Marking Scheme : The mark for each question and sub question is shown with the question below. Place your solutions in tables where possible.

For office Use only

Question	Mark
1	/20
2	/10
3	/10
4	/10
Total	/50

CHAPTER 5: THE RELATIONAL DATA MODEL AND RELATIONAL DATABASE CONSTRAINTS

1. (Total marks 20) Given the same simple Farmer-growscrops-inRegions database schema, which contains three files (e.g., relations) described as follows, answer the following questions with regards to this database. (Total for que 1 is 10 marks)

   Farmer (Ssn: integer, Name: string, Age: integer, Regionid: integer)
   Growscrops (Ssn: integer, Regionid: integer, Crop: string, Revenue: real)
   Region (Regionid: integer, Regname: string, Cropbudget: real, managerid: integer)

   Note: Ssn, Name, Age, Regionid are the social security number, name, age and Regionid of the farmer respectively. Also, Regname, Cropbudget and managerid represent the regionid (e.g., SWO for southwestern Ontario), all crop budget (amount) for growing in this region (e.g., 100 for $100M), and managerid (a farmer who oversees farmer affairs in this region). Revenue from grown crop can be measured in millions of dollars as well (e.g., 0.5 means half a million). A farmer may grow crop in multiple regions.

   Assume that an update operation (a general term in this chapter, for an insert, a modify or a delete operation) is to be made to this database to enter information about a new region not already in the database but which has just got some crops grown in it by some farmers. Answer the following questions on what specific relations, attributes and operations (eg. insert, modify, delete) that need to be done for this update to be implemented in the entire database. This is not SQL query yet.

   Provide your answers both in descriptive sentence and using the formal (not SQL) database operations of INSERT, MODIFY, DELETE as used in Chapter 5 of book with specific attributes and relations when possible. An example formal insert of a farmer record into the Farmer table is:
   INSERT < Ssn, Name, Age, Regionid> into Farmer // for new Person record.
   And an example descriptive sentence is:
   i). do an insert operation for a new farmer record into the Farmer table.

   (a) Give the set of needed insert, modify or delete operations for this update operation scenario described above. 5 marks
   (b) What types of integrity constraints (explain using attributes, e.g., SSN of relevant files) would you expect to check for this update to be done? 5 marks
   (c) Which of these integrity constraints are key, entity integrity, and referential (foreign key) integrity constraints and which are not? 5 marks
   (d) Specify all the referential integrity (foreign key) constraints on this database in the format Referring_Relation.Attribute --> Referred_Relation.Attribute. 5 marks
   (Total for que 1 is 20 marks)

Solution:

Question	**Answers**
a. Give the operations for this update. 5 marks	One possible set of operations for the given update is the following: Do a insert operation for new region into the region table: INSERT <Regionid, Regname, Cropbudget, managerid> into Region Do a update operation for regionID for the growscrop table Update < Regionid> into GrowsCrops Do a update operation for regionID for the farmers table Update < Regionid> into Farmer
b. What types of integrity constraints would you expect to check? (explain using attributes, eg, Ssn of relevant files) 5 marks	Entity Integrity: Ensures that each Region has a unique Regionid Domain Integrity: Ensures that the data types of attributes are consistent with their domains. Ie, SSN being a string, Name being a string aswell and not numbers, Regionid being a int. Referential Integrity (Foreign Key): Ensures that the managerid in the Region table refers to a valid Ssn in the Farmer table.
c. Which of these integrity constraints are key, entity integrity, and referential integrity constraints and which are not? 5 marks	Entity Integrity constraint is well, an entity integrity and is relevant to the Regionid in the Region table. Domain Integrity constraints are not one of the three integrity constraints listed, but they are relevant to all attributes in their respective tables. Referential Integrity constraint is also a referential integrity and is relevant to the managerid in the Region table.
d. Specify all the referential integrity constraints on this database. 5 marks	Farmer.Regionid --> Growscrops.Regionid -Regionid in Growscrops references a valid Regionid in the Farmer. Growscrops.Regionid --> Region.Regionid -Regionid in Region references a valid Regionid in the Growscrops. Region.managerid --> Farmer.Ssn managerid in Region references a valid Ssn in the Farmer.

2. (total marks 10) Using your own Farmer-growscrops-inRegions database instance from assignment 1, login to the SQL query processor on our cs server, called Oracle Sqlplus to create the three database tables and insert the tuples in your database state with the following sequence of instructions. The entity tabes need to be created first before the relationship tables that are referencing them or it will complain. Thus, follow your table creation in the order given in the instructions. Note that this exercise is to get you beginning to connect to SQLplus while preparing to learn full SQL language syntax in Chapters 6 and 7. You will be given the instructions to use now. Show the result of this exercise through a Unix script file you will attach as a .txt file.

   (Total for que 2 is 10 marks)

   i. First connect to our cs.uwindsor.ca through either Bitvise SSH client or NoMachine. You need to connect to global protect before connecting to any campus server. Check course web site and brightspace pages for more details on how.
   ii. Then hand in a Unix script file to capture your Unix session when you connect to Sqlplus after your instructions for creating your database are working. You can create your Unix script file using the following sequence of instructions on a Unix terminal on our cs server. You need to transfer this script file to your personal computer using a file transfer protocol (eg. Bitvise SFTP or Filezilla) in order to attach it in your assignment submission.

```
>script   username_assn2que2.txt
>sqlplus <username>
>password

SQL>   CREATE TABLE FARMER(
SSN NUMBER(3) NOT NULL,
NAME    VARCHAR2(15),
AGE NUMBER(3),
REGIONID  NUMBER(3),
PRIMARY KEY(SSN));

SQL> CREATE TABLE REGION
 (
 REGIONID  NUMBER(3) NOT NULL,
 REGNAME VARCHAR2(15) NOT NULL,
 CROPBUDGET  NUMBER(10,2),
 MANAGERID NUMBER(3),
 PRIMARY KEY (REGIONID),
 FOREIGN KEY(MANAGERID) REFERENCES FARMER(SSN));

SQL> CREATE TABLE GROWSCROPS
(
SSN   NUMBER(3)  NOT NULL,
REGIONID  NUMBER(3) NOT NULL,
CROP VARCHAR2 (15) NOT NULL,
REVENUE NUMBER(10, 2),
PRIMARY KEY(SSN, REGIONID, CROP),
FOREIGN KEY(SSN) REFERENCES FARMER(SSN),
FOREIGN KEY(REGIONID) REFERENCES REGION(REGIONID));

SQL> -- A sample insert of a record into each of the 3 tables is given below
SQL> INSERT INTO FARMER VALUES (10, 'Jobe Bata', 65, 1);
SQL> INSERT INTO REGION VALUES(1, 'SWO', 5.0, 10);
SQL> INSERT INTO GROWSCROPS VALUES (10, 1,  'corn', 2.50);

SQL> COMMIT;
// Repeat similar INSERT instructions for all the data in all your tables
// starting with the entity tables first, eg, Farmer, Region, before Growscrops.

SQL> select * from cat;   // to show all the objects in your catalogue

SQL> select * from Farmer;  // to show the contents of this table

SQL> exit                     //to exit sqlplus


>exit                              // to exit and create Unix script file to hand in
```

** More Hint: While in Sqlplus, if you want to delete data from your tables and drop them before issuing your instructions for creating your Unix script file for handing in, you can use the following instructions for each table to first delete the data from the table and then drop the table.

```
delete from GROWSCROPS;
delete from FARMER;
delete from REGION;

commit;

drop table GROWSCROPS cascade constraints;

drop table FARMER cascade constraints;

drop table REGION cascade constraints;

commit;
```

***

Also Note: you can start creating a script file only after you have created your tables correctly and inserted data in the tables. In that case, you cannot re-create existing tables. Then, you can just run the desc table (eg. Desc Person) command for each table to show the structure of each table before using (for example), the (select * from Person;) to show the tuples of each table or delete data and drop the tables as explained above so you can re-create the tables more correctly.

Solution: (10 marks)

An attached Unix script file showing execution of CREATE TABLE instructions and INSERT INTO tablename VALUES instructions with the few SELECT instructions to show contents of the catalogue and all tables (your database instance).

CHAPTER 14: Database Design Theory: Introduction to Normalization Using Functional and Multivalued Dependencies

3. (total marks 10) Consider the following relation:

Enrolled(Studid, Crsid, SName, Score, Lettergrade)

Assume that a student (Studid) may be enrolled in multiple courses (Crsid) and hence {Studid, Crsid} is the primary key.
Thus, the following functional dependency exists:

{Studid, Crsid} -> {SName, Score, Lettergrade}

Additional dependencies are:

Studid -> SName

Score -> Lettergrade

Based on the given primary key,

i. is this relation in 1NF, 2NF, or 3NF? Why or why not?
ii. If not in 2NF at least, normalize it completely into 2NF and 3NF? Provide your answers using functional dependencies (FDs).

(Total for que 3 is 10 marks)

Solution (i): (5 marks)

Answer:
Given the relation schema
Enrolled(Studid, Crsid, SName, Score, Lettergrade)

with the functional dependencies
{Studid, Crsid } -> {SName, Score, Lettergrade}
Studid -> SName
Score -> Lettergrade

(i) is this relation in 1NF, 2NF, or 3NF? Why or why not?

In 1NF, all attributes in a relation must contain atomic values.
The attributes are Studid, Crsid, SName, Score, Lettergrade and since these are all unique and cant be subdivided this relation passes the 1st normal form

In 2NF, the relation needs to already be 1NF and there should be no partial dependencies. The candidate keys are StudID and CrsID and with the listed dependencies from above there should be no partial dependencies because there is no proper subset of the keys to determine any non-prime attribute(s).

In 3NF, the relation needs to have both 1NF and 2NF, and there needs to be no transitive dependencies. However, there is a transitive dependency from Studid to SName through {Studid, Crsid}, so the relation is not 3NF and **2NF**

Solution (ii) (5 marks)

(ii) If not in 2NF at least, normalize it completely into 2NF and 3NF? Provide your answers using functional dependencies (FDs).

To normalize this relation to be 2NF we would need to remove the partial depenecies and to do that we create two separate relations, i.e.
Enrollments (Studid, Crsid, SName, Score)
PK = (Studid, Crsid)
FD = (Studid, Crsid) -> (Sname,Score)

CourseInfo (Crsid,Lettergrade)
PK = (Crsid)
FD = none needed

To normalize this into 3NF we need to remove transitive dependencies. So we would create another new relation so it would be
Enrollments (Studid, Crsid, Score)
PK = (Studid, Crsid)
FD = (Studid, Crsid) -> (Score)

CourseInfo(Crsid, Lettergrade)
PK = (Crsid)
FD = none needed

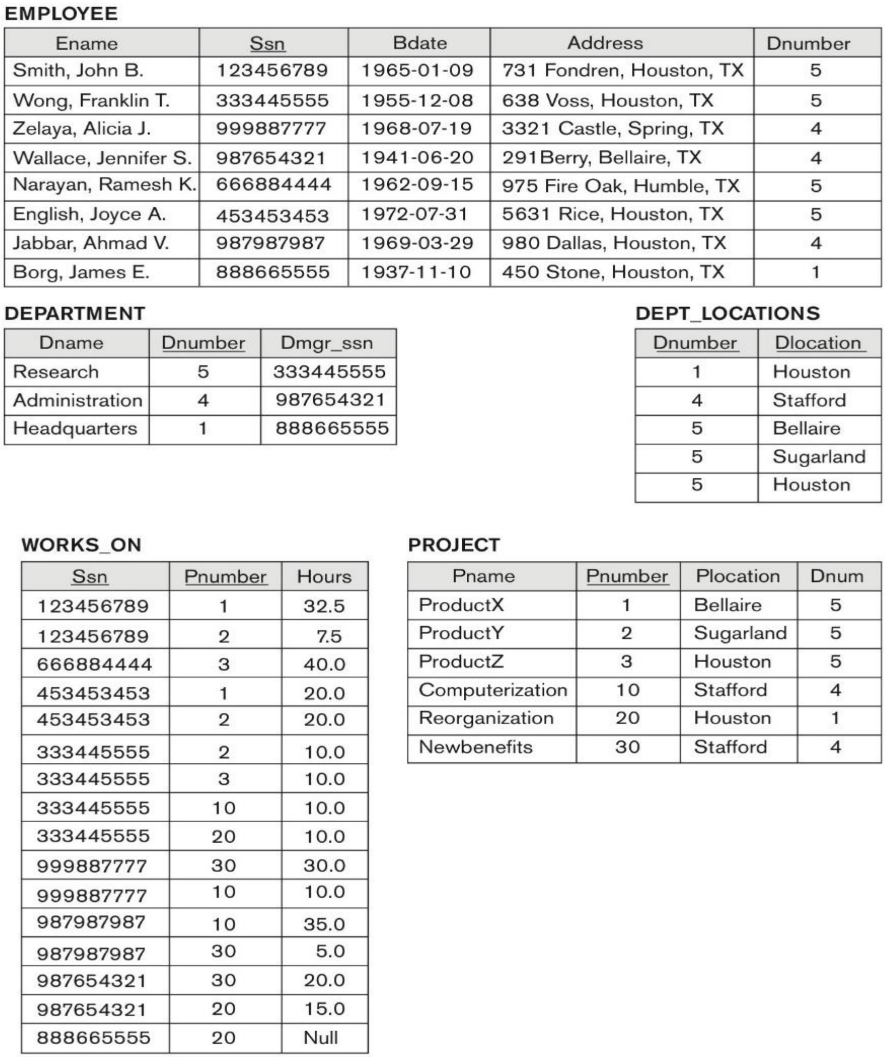
StudentInfo(Studid, SName)
PK = (Studid)
FD = none needed

4. (total marks 10) What (i) update, (ii) delete and (iii) insertion anomalies occur in the DEPARTMENT_PROJECT relation obtained by doing a natural join of the two relations DEPARTMENT and PROJECT of Fig 14.2 on page 463 of book? Explain with examples using this database and the DEPARTMENT_PROJECT relation schema with state given below as Figures 4.1 and 4.2 below.

(Total for que 4 is 10 marks)

Note: 3 marks for correct discussion of each anomaly and 1 mark for attempt.

Figure 14.2 (book): Sample database state for a simplified COMPANY relation DB



EMPLOYEE

Ename	Ssn	Bdate	Address	Dnumber
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	1941-06-20	291Berry, Bellaire, TX	4
Narayan, Ramesh K.	666884444	1962-09-15	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

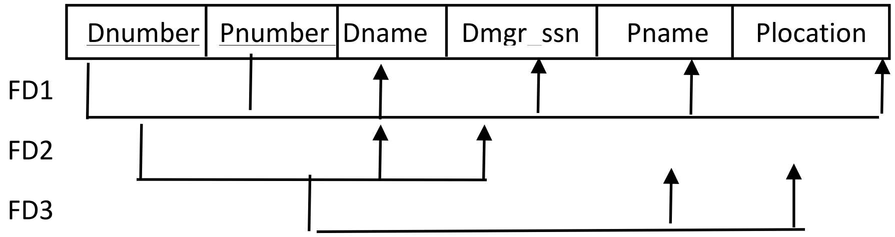
Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

****

Fig 4.1: DEPARTMENT_PROJECT DB schema suffering from update anomalies



***

Fig 4.2: A database state of the DEPARTMENT_PROJECT DATABASE derived from Fig 14.2

DNUMBER	PNUMBER	DNAME	DMGR_SSN	PNAME	PLOCATION
5	3	Research	333445555	ProductZ	Houston
5	10	Research	333445555	Computerize	Stafford
5	20	Research	333445555	Reorganize	Houston
5	30	Research	333445555	Nbenefits	Stafford
5	1	Research	333445555	ProductX	Bellair
5	2	Research	333445555	ProductY	Sugarland
4	3	Administration	987654321	ProductZ	Houston
4	10	Administration	987654321	Computerize	Stafford
4	20	Administration	987654321	Reorganize	Houston
4	30	Administration	987654321	Nbenefits	Stafford
4	1	Administration	987654321	ProductX	Bellair
4	2	Administration	987654321	ProductY	Sugarland
1	3	Headquarters	888665555	ProductZ	Houston
1	10	Headquarters	888665555	Computerize	Stafford
1	20	Headquarters	888665555	Reorganize	Houston
1	30	Headquarters	888665555	Nbenefits	Stafford

```
    1      1           Headquarters   888665555   ProductX    Bellair
    1      2            Headquarters  888665555   ProductY    Sugarland

18 rows selected.
```

Solution: (3 + 3 + 3 + 1 marks)

Update

If a attribute (e.g., DName, Pname, Plocation) is updated, it affects all rows where the Dnum matches. For example, if you update the "Plocation" for a project with Dnum 5, it will change the "Plocation" for all projects associated with department 5.

Delete

When you delete a row in the "DEPARTMENT_PROJECT" relation, you lose that combination of a department and a project. Like, if you delete a row that represents the "Research" department with Dnum 5 and the project "ProductZ," you lose information about that specific combination.

Insertion

Inserting new data into the "DEPARTMENT_PROJECT" relation can be difficult to make sure that you include complete information for both departments and projects. If you want to insert a new combination of department and project, you need to insert the department and project information again, potentially causing redundancy. There is also another chance of redundancy because if the same department and project combinations can be entered multiple times.