Data Management and Database Design

INFO 6210

Fall 2016

Assignment _2

Submission Date – Sep/27/2016 Tuesday

Student Name: Malick Fairoz Sayeed Abuthahir

NUID: 001235450

Program: MS in Information Systems

Professor Name: Yusuf Ozbek

College: College Of Engineering

University: Northeastern University

Book -1

Exercises

1. As it happens, the film club holds meetings regularly in several different locations, which means a lot of redundancy in the Attendance table. What changes could you make to the database table structure?

ANS)

If we have all the information in one table then it will be very difficult for accessing the data as it grows more and there will be lot repeated values in many columns. This is will not allow us to create index in the table. Since Index requires unique values in its column. As a result it reduces the performance.

This can be overcome by making the tables separated.

In the base table “film club” we can have the columns

a) Member ID (Primary Index)
b) Name
c) Date of Birth
d) Street name
e) City
f) Zip code
g) Country
h) Email
i) Date of Joining

In the “Attendance” table we can have the columns

a) Meeting ID (Primary Index)
b) Meeting datetime
c) Location
d) Member attendance
e) Member ID

By adding “Meeting ID” column in the “Attendance” table, each meeting will have a unique identification number and that can be the primary index of the table. In addition to that having the data, time and location of the meeting along with the member ID column will help us to access the data better. This will reduce the redundancy in the Attendance table.

2. Write the necessary SQL to complete the changes required by Exercise 1 and at the same time split the location's address details into street, city, and state.

ANS)

In the beginning the "Film Club" table was created in the below structure which has much redundancy

```
CREATE TABLE `malick`.`film_club` (
  `Name` VARCHAR(75) NULL,
  `DOB` DATE NULL,
  `Address` VARCHAR(300) NULL,
  `email` VARCHAR(200) NULL,
  `Date_of_Joining` DATE NULL,
  `Meeting_date` DATE NULL,
  `Location` VARCHAR(200) NULL,
  `Did_member_attend` VARCHAR(2) NULL);
```

The table "Attendance" has been created to separate the base table into two parts of information.

```
CREATE TABLE `malick`.`attendance` (
  `Meeting_ID` INT NOT NULL,
  `Meeting_date_time` DATETIME(20) NOT NULL,
  `Location` VARCHAR(45) NOT NULL,
  `Member_Attendance` VARCHAR(2) NOT NULL,
  `Member_ID` INT NOT NULL,
  PRIMARY KEY (`Meeting_ID`),
  UNIQUE INDEX `Meeting_ID_UNIQUE` (`Meeting_ID` ASC));
```

The table "Film Club" is altered to have information only about the member's details and to alter the column that it should not contain the NULL values.

```
ALTER TABLE `malick`.`film_club`
DROP COLUMN `Did_member_attend`,
DROP COLUMN `Location`,
DROP COLUMN `Meeting_date`,
CHANGE COLUMN `Name` `Name` VARCHAR(75) NOT NULL ,
CHANGE COLUMN `DOB` `DOB` DATE NOT NULL ,
CHANGE COLUMN `Address` `Street` VARCHAR(200) NOT NULL ,
CHANGE COLUMN `email` `email` VARCHAR(200) NOT NULL ,
CHANGE COLUMN `Date_of_Joining` `Date_of_Joining` DATE NOT NULL ,
ADD COLUMN `Member_ID` INT NOT NULL FIRST,
```

```
ADD COLUMN `City` VARCHAR(45) NOT NULL AFTER `Street`,
ADD COLUMN `State` VARCHAR(45) NOT NULL AFTER `City`,
ADD COLUMN `Zip_Code` VARCHAR(45) NOT NULL AFTER `State`,
ADD COLUMN `Country` VARCHAR(45) NOT NULL AFTER `Zip_Code`,
ADD PRIMARY KEY (`Member_ID`),
ADD UNIQUE INDEX `Member_ID_UNIQUE` (`Member_ID` ASC);
```

The table "Attendance" is altered to separate the location column into Street, City and State information. Furthermore a foreign key index is added to the "Member_ID" column referencing the table "film_club"

```
ALTER TABLE `malick`.`attendance`
CHANGE COLUMN `Location` `Street` VARCHAR(45) NOT NULL ,
ADD COLUMN `City` VARCHAR(45) NOT NULL AFTER `Street`,
ADD COLUMN `State` VARCHAR(45) NOT NULL AFTER `City`,
ADD COLUMN `Zip_Code` VARCHAR(45) NOT NULL AFTER `State`,
ADD COLUMN `Country` VARCHAR(45) NOT NULL AFTER `Zip_Code`,
ADD INDEX `film_club_mem_id_idx` (`Member_ID` ASC);
ALTER TABLE `malick`.`attendance`
ADD CONSTRAINT `film_club_member_id`
  FOREIGN KEY (`Member_ID`)
  REFERENCES `malick`.`film_club` (`Member_ID`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;
```

Book -2

Exercises

1. Three new members have joined the film club. Add the following details to the database:

Member ID: 7 First Name: John Last Name: Jackson Date of Birth: May 27, 1974 Street: Long Lane City: Orange Town State: New State Zip Code: 88992 Email: jjackson@mailme.net Date of Joining: November 21, 2005

Member ID: 8 First Name: Jack Last Name: Johnson Date of Birth: June 9, 1945 Street: Main Street City: Big City State: Mega State Zip Code: 34566 Email: jjohnson@me.com Date of Joining: June 2, 2005

Member ID: 9 First Name: Seymour Last Name: Botts Date of Birth: October 21, 1956 Street: Long Lane City: Windy Village State: Golden State Zip Code: 65422 Email: Seymour@botts.org Date of Joining: July 17, 2005

You need to ensure that the date format matches the format expected by your database system. Remember that Oracle accepts day-month-year format (23 January 2004), whereas the other four databases expect the format year-month-day, such as 2004-01-23.

ANS)

To insert the data of three new members into the table “film _club” who have joined the club, we need to use the below mentioned insert queries.

INSERT INTO film_club VALUES (7,'John','Jackson','1974-05-27','Long Lane','Orange Town','New State','88992','jjackson@mailme.net','2005-11-21');

INSERT INTO film_club VALUES (8,'Jack','Johnson','1995-06-09','Main Street','Big City','Mega State','34566','jjohnson@me.com','2005-06-02');

INSERT INTO film_club VALUES (9,'Seymour','Botts','1956-10-21','Long Lane','Windy Village','Golden State','65422','Seymour@botts.org','2005-07-17');

2. 2. Bob Robson, MemberId 2, took a job in Hollywood and left the club. Delete his details from the database. Remember to delete not only his membership details but also his film category preferences.

ANS)

To delete the details of a member “Bob Robson” from the “film_club” table who has the member_id = 2, we need to use the below mentioned SQL query statement and the film category details of this particular member will also be deleted if the category table has foreign key constrains referring the base “film_club” table’s column.

DELETE FROM film_club WHERE Member_Id='2';

3. The government has decided to reorganize the boundaries of Orange Town. All residents living on Long Lane in Orange Town now live in Big City. Update the database to reflect this change.

ANS)

The below update SQL query statement will update the name of the city for the residents living on Long Lane in Orange Town.

UPDATE film_club1 SET City = 'Big City' WHERE Street = 'Long Lane' and city = 'Orange Town';

Book -3

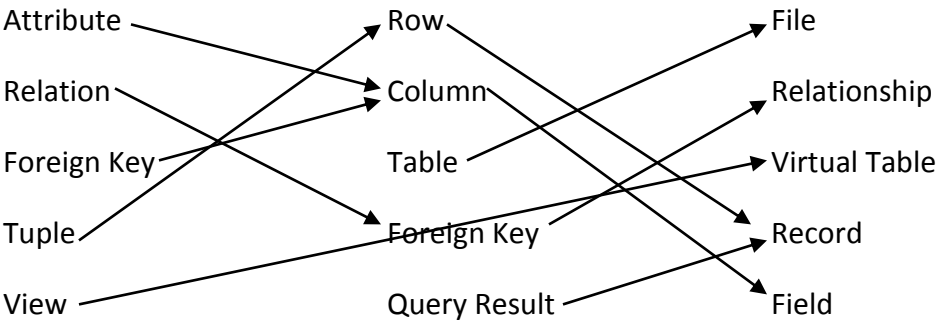
Exercises

1. What does the following check constraint on the SalesPeople table mean?
((Salary > 0) AND (Commission = 0)) OR ((Salary = 0) AND (Commission > 0))

ANS)

It is a table level check constrain since it validate more than one record field. Here this constrain describe that the record must have positive salary along with 0 commission values or it can have positive commission along with 0 salary values or both the conditions.

2. Draw lines connecting the corresponding terms.



3. Is State/Abbr/Title a superkey? Why or why not?

ANS)

The columns State/Abbr/Title is a **superkey** in this table, since there are no two rows has same values in these columns. They are also called unique keys.

4. Is Engraver/Year/Got a superkey? Why or why not?

ANS)

The columns Engraver/Year/Got are also a **superkey** in this table, since there are no two rows has same values in these columns.

5. What are all of the candidate keys for this table?

ANS)

The candidate key is a minimal of the superkey. Suppose if a column is removed from a superkey then it should not hold the power of superkey, this is known as candidate key.

In this table for the superkey “State/Abbr/Title” if you remove the column state then also it is a superkey since there are no two rows has a same value for columns “Abbr/Title”. This is not a candidate key because it is not a minimal superkey.

But in the same table for the superkey “Engraver/Year/Got” if you remove the column Engraver then it is not a superkey since there are two rows has a same value for columns “Year/Got”. This is a candidate key because it is a minimal superkey.

6. What are the domains of each of the table’s columns?

Column1	State	can have only the name of the states
Column2	Abbr	can have only the two letter of the state’s abbreviation
Column3	Title	can have only the short description of the state as a Title
Column4	Engraver	can have only the names which is engraved
Column5	Year	can have only the four digit numbers
Column6	Got	can have only the values ‘yes’ or ‘no’

7. If you don’t allow two people with the same name to share a room (due to administrative whimsy), what are all of the possible candidate keys for this table?

ANS)

The possible candidate keys for this table are

1) Room\ FirstName
2) Room\ LastName

8. If you do allow two people with the same name to share a room, what are all of the possible candidate keys for this table?

ANS)

There is no candidate keys to allow people with the same name to share the a room

9. What field-level check constraints could you put on this table's fields? Don't worry about the syntax for performing the checks, just define them.

ANS)

(Room>0)

(Room<999)

10. What table-level check constraints could you put on this table's fields? Don't worry about the syntax for performing the checks, just define them.

ANS)

((FirstName= Not Null) AND (LastName = Not Null))

((Phone = Not Null) OR (CellPhone = Not Null))

Book -4

Exercises

1. Take the following column titles, assign them to a data type, decide on the proper length, and give an example of the data you would enter into that column. A. ssn B. state C. city D. phone_number E. zip F. last_name G. first_name H. middle_name I. salary J. hourly_pay_rate K. date_hired

ANS)

There are different data types are assigned to each columns with proper length in the create statement of the table "student".

```
CREATE TABLE `malick`.`student` (
  `ssn`                    INT(9) NOT NULL,
  `state`                  VARCHAR(20) NOT NULL,
  `city`                   VARCHAR(20) NOT NULL,
  `phone_nnumber`          VARCHAR (10) NOT NULL,
  `zip`                    VARCHAR(10) NOT NULL,
  `last_name`              VARCHAR(45) NOT NULL,
  `first_name`             VARCHAR(45) NOT NULL,
  `middle_name`            VARCHAR(45) NULL,
  `salary`                 VARCHAR(10) NOT NULL,
  `hourly_pay_rate`        VARCHAR(5) NOT NULL,
  `date_hired`             DATE NOT NULL,
  PRIMARY KEY (`ssn`),
  UNIQUE INDEX `ssn_UNIQUE` (`ssn` ASC),
  UNIQUE INDEX `phone_nnumber_UNIQUE` (`phone_nnumber` ASC));
```

The samples values that would enter into the table columns are mentioned in the below insert query.

```
INSERT INTO student VALUES
(123545089,'Massachusetts','Boston','6178589071','02120','Sayeed Abuthahir','Malick
Fairoz',Null,'144000','100','2016-09-07');
```

2. Take the same column titles and decide whether they should be NULL or NOT NULL, realizing that in some cases where a column would normally be NOT NULL, the column could be NULL or vice versa, depending on the application. A. ssn B. state C. city D. phone_number E. zip F. last_name G. first_name H. middle_name I. salary J. hourly_pay_rate K. date_hired

ANS)

For the column "ssn, phone_number, "the values should be notnull and unique since they will be present uniquely for every person under this table.

For the column "middle_name" the values need not to be mandatory since some of the people will not have their middle name, so it can have null values.

For the remaining other columns the values need not to be unique since these can be common for many people. Similarly they will have a value mandatory so these have to be NOT NULL.

```
CREATE TABLE `malick`.`student` (
  `ssn`              INT(9)              NOT NULL,
  `state`            VARCHAR(20)         NOT NULL,
  `city`             VARCHAR(20)         NOT NULL,
  `phone_nnumber`    VARCHAR (10)        NOT NULL,
  `zip`              VARCHAR(10)         NOT NULL,
  `last_name`        VARCHAR(45)         NOT NULL,
  `first_name`       VARCHAR(45)         NOT NULL,
  `middle_name`      VARCHAR(45)         NULL,
  `salary`           VARCHAR(10)         NOT NULL,
  `hourly_pay_rate`  VARCHAR(5)          NOT NULL,
  `date_hired`       DATE                NOT NULL,
  PRIMARY KEY (`ssn`),
  UNIQUE INDEX `ssn_UNIQUE` (`ssn` ASC),
  UNIQUE INDEX `phone_nnumber_UNIQUE` (`phone_nnumber` ASC));
```