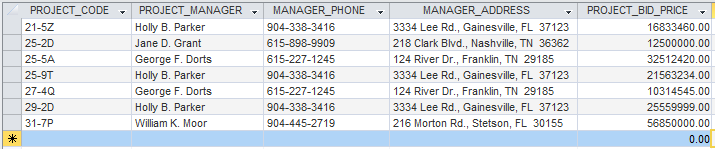
CS60 Project 1 Fall 2018

Due Midnight, Sunday Sept 9

Total 25 Pts

*Figures from below are also available as data files if you prefer to use MS Access.*



1. How many records (rows of raw data) does the above table store, and how many fields (columns or attributes) are in each record? (2 Pts)

7 rows

5 column attributes

2. What problem would you encounter if you wanted to list the records in order of the manager’s last name, or if you sometimes wanted to omit the first name or middle name in a display or printout? This design fault is referred to as a **composite attribute**. Show the table structure of an altered table that will correct this problem? (2 Pts)

The first, middle and last names contained within a single field are *composite attributes* **for name** that make sorting more difficult unless redesigning the table as follows:

Column Fields – PROJECT\_CODE

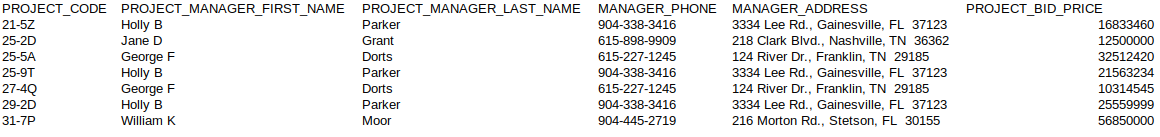
PROJECT\_MANAGER\_LAST\_NAME

PROJECT\_MANAGER\_FIRST\_NAME

MANAGER\_PHONE

MANAGER\_ADDRESS

PROJECT\_BID\_PRICE

The newly revised table should look something like this:

3. What problem would you encounter if you wanted to list the records in order of the street address, city, state, or zip, or area code? Building upon the improvements that you’ve already made, show the table structure of an altered table that also corrects this problem? Show all columns and rows in this revised table, including the new ones from Step 2.

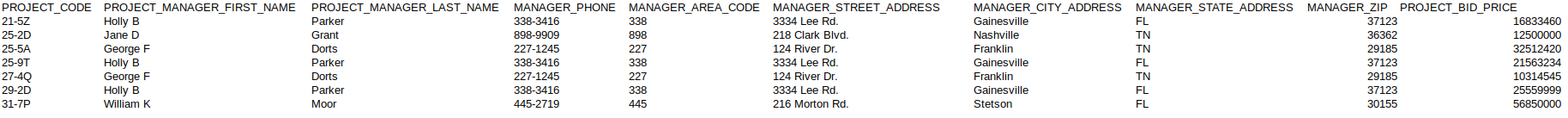
(3 Pts)

*Table redesign would also improve the composite attributes for address and phone by making use of the*

*extended* fields for:

MANAGER\_STREET\_ADDRESS, MANAGER\_ADDRESS\_CITY, MANAGER\_ADDRESS\_STATE, MANAGER\_ADDRESS\_ZIP, MANAGER\_AREA\_CODE.

E.g.



4. What data redundancies do you detect; i.e., what unnecessary repetitions are occurring? How could these redundancies lead to update anomalies, delete anomalies, or insert anomalies? (2 Pts)

***The resulting consequences of data redundancy in this table falls under the three categories of update/delete/insert anomalies.***

Update Anomaly - when data is entered multiple times for the same manager this redundancy creates the need to go back and change it in multiple places.

Deletion anomaly - is present (not really associated with the redundancy though) for manager data that could unintentionally (mistakenly) be removed/lost when making deleting a project.

Insert anomaly - is present in the sense of creating entries for new managers would require a new project for that manager which may or may not actually be the case in reality, and yet is required by table structure.

5. Using two relational tables, PROJECT and MANAGER, eliminate the redundancies you identified in Problem 4. Create a ManagerID column in both tables so you can link the two tables with the ManagerID being the primary key in MANAGER and a foreign key in PROJECT. Identify the primary key in each table. With words, show how the two tables join together by a foreign key that references a primary key. The columns must correct all faults (composite attributes and redundancies) that you saw above. (5 Pts)

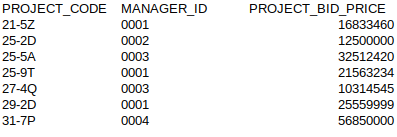
***Answer:***

PROJECT.MANAGER\_ID references MANAGER.MANAGER\_ID

*Foreign Key* *Primary Key*

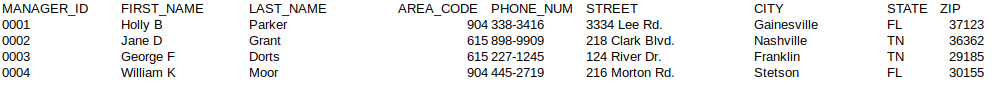
‘Projects’ Table:

PK FK



‘Managers’ Table:

PK



***Redundancy reduced and all relevant information remains.***

6. Create the **relational schema** to show the two tables and their columns, primary keys, foreign key, a line that shows how the two tables join, and the symbols 1 and ∞ (for *many*).The columns must correct the faults you saw above.(5 Pts)

Projects

Managers

1

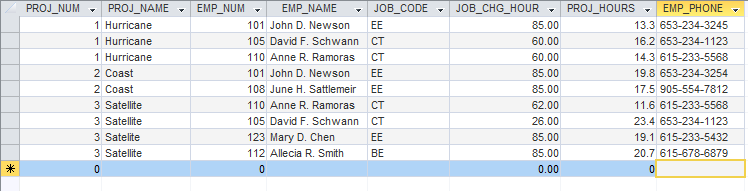
|  |
| --- |
| **Project\_Code** |
| Project\_Bid\_Price |
| New\_Field |
| ManagerID |

∞

Possibly a new field for other project related or cost related item.

|  |
| --- |
| **Manager\_ID** |
| First\_Name |
| Last\_Name |
| Area\_code |
| Phone\_num |
| Street\_Address |
| City |
| State |
| Zip |

A new table for questions 7 and 8:



7. Based on the table above, identify pairs of columns (***actually should be pairs of rows not columns)*** that for the same value in one column, the 2nd column also has the same value. Such columns are **dependent** upon each other, or one column **determines** the other. You could write this functional relationship as

Column2 = function(Column1)

Unlike mathematical functions such as y = x2 and functions that are plotted or graphed as y = f(x), this function is a tabular function with data stored in a table. (3 Pts)

***It appears attributes and values of entries in rows show functional dependences in,***

1. *Project Number and Project Name*
2. *Employee Number and Employee Name*
3. *And possibly between Employee Name/Number and Job Code*

8. These dependencies lead to what redundancies in the table (what data is being stored redundantly)? Do you see any relationship between the pairs of columns that you identified in Question 7 and the occurrence of redundancies?

(3 Pts)

***Redundant entries like the names of projects or employees can cause update or insertion anomalies; functional dependence in data table (with redundancy) can also lead to deletion anomalies.***

9. Create an ERD for each of the following descriptions. (Note the work *many* merely means *more than one* in the database modeling environment.)

a. Each of the ABC Corp’s divisions is composed of many departments. Each department has many employees assigned to it, but each employee works for only one department. Each department is managed by one employee and each of the managers can manage only one department at a time. (4 Points)

Employees

Employee

Employee

Employee

Employee

Employee

Dept Managers

(optional)

Man\_Department1

Man\_Department2

…

Departments

Department 1

Department 2

Department 3

Department 4

1

ABC Divisions

Division 1

Division 2

Division 3

1

1

M

1

M

***Department Managers could optionally not be an entity but rather attribute of employee table and I believe it is.***

b. During some period of time, a customer can download many ebooks from BooksOnline. Each of the books can be downloaded by many customers during this period of time. (2 Pts)

Books

Catalogue of many books….

Customers

Customer1

Customer2

etc.

M

M

***Books to customers have a many-to-many relationship.***

c. An airliner can be assigned to fly many flights, but each flight is flown by only one airliner. (2 Pts)

Airliner

Company\_1

Company\_2

etc.

Flights

Flight #\_A1001

Flight #\_B1001

Flight #\_C1001

M

1

***Airliner to Flight has a one-to-many relationship between the airline and its outbound/inbound travel.***

d. QuickTime Corp operates many factories. Each factory is located in a region and each region can be home to many

QuickTime factories. Each factory has many employees but each employee is employed by only one factory.

(4 Pts)

Factories

Factory

Factory

Factory

Factory

Factory

Regions

Region 1

Region 2

3

4

1

M

Employees

Employees

Employees

Employees

Employee

Employee

1

M

***Regions of the corporation have a one-to-many relationship with Factories, and those factories also have a one-to-many relationship with employees. The regions are therefore also one-to-many with employees (implied relationship).***

e. An employee may have earned many degrees and each degree may have been earned by many employees. (3 Pts)

Employee

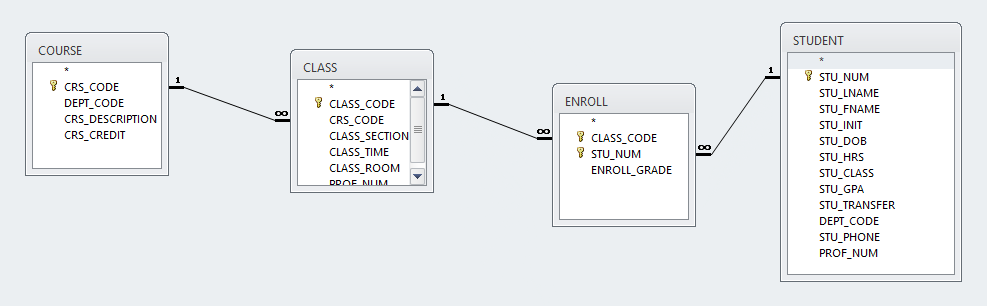
Degree

M

M

***A many-to-many relationship exists.***

10. Use this Figure as a guide to answer parts (a) to (c).



a. identify each relation type and write all business rules. (3 Pts)

***Relation types are usually considered among rows and columns. Therefore, without assuming a specific data model, which could be any of several types like Entity Relationships (Relational Model or other types of databases) the entities, attributes, and some constraints can be addressed with business rules.***

* Each course can generate many classes, but each class refers to only one course.
* Each class can generate many enrollments, but each enrollment is for a single class.
* Each student may enroll many times, but each enrollment entry is for a single student.
* Each enrollment entry must match a class to a student and thus students to courses.
* Other potential business rules:
  + Students enrollment should be made into classes, not courses.
  + Students should be allowed to enroll into only one class in a given course, i.e. no duplicate course.
* Rules for keys and tables:
  + Paired foreign keys student and class form a candidate key (not null and unique characteristics)
  + The class sections and course codes are a unique pairing as well that could make up a candidate key.

b. create the basic Crow’s Foot ERD for Tiny College. (3 Pts)

Course

Class

Enrollment

Student

c. create the UML class diagram that reflects the entities and relationships you identified in the relational diagram.

(4 Pts)

1…\*

1…1

1…1

Class

+has a

+has a

+is a

+is a

+has a

Course

1…\*

Enroll

Student

+is a

0…\*

1…1