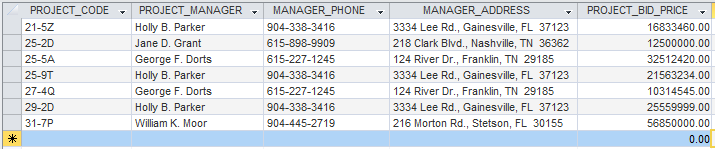
CS60 Project 1 Fall 2018

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 all within a single field make that field a *composite attribute* **for name**.

Before sorting we would want to redesign the table to have columns/fields be as follows:

Column Fields – PROJECT\_CODE

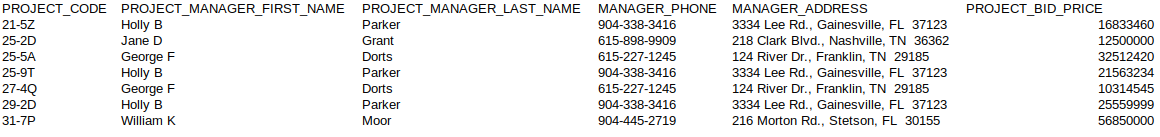
PROJECT\_MANAGER\_LAST\_NAME

PROJECT\_MANAGER\_FIRST\_NAME

MANAGER\_PHONE

MANAGER\_ADDRESS

PROJECT\_BID\_PRICE

The newly revised table with data 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.

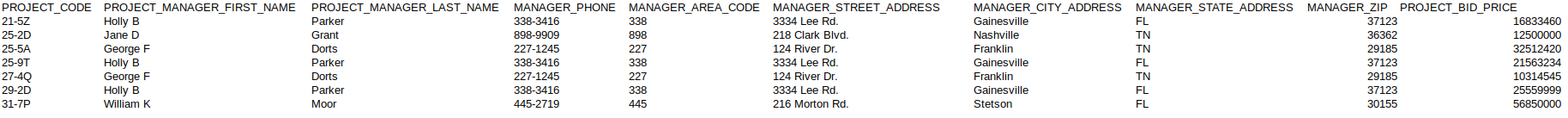
(3 Pts)

***a redesign would improve the situation for these also composite attributes (address, phone)***

The new column fields should contain at the very least the following:

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

E.g. The Revised Table:



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 data redundancies in this table fall under the three categories described (update/delete/insert)***

Update Anomaly - exists in the table when data is entered multiple times for the same manager. This redundancy may create a need to go back and change the data in multiple places for for the same person.

Deletion anomaly - is present (not really associated with the redundancy though) in which manager data could be removed from the database when removing a project from the table.

Insert anomaly - is present as well in the sense that creating a place for a new manager here would require creating a project for that manager which may or may not actually be the case in reality, but would possibly be required.

***Question - could insert anomaly also be solved by allowing a null entry in the field for Project Code?***

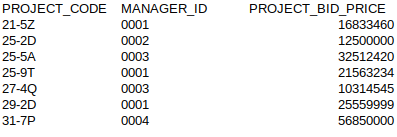
5. Using two relational tables, PROJECT and MANAGER, eliminate the redundancies you identified in Problem 4. Create a ManagerID column in both tables so you can link the two tables with the ManagerID being the primary key in MANAGER and a foreign key in PROJECT. Identify the primary key in each table. 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

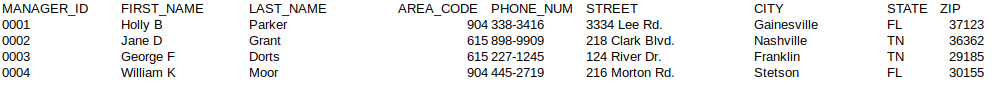
*Foreign Key* *Primary Key*

‘Projects’ Table: PK FK



‘Managers’ Table:

PK



***Redundancy has been reduced and with all relevant information remaining.***

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

∞

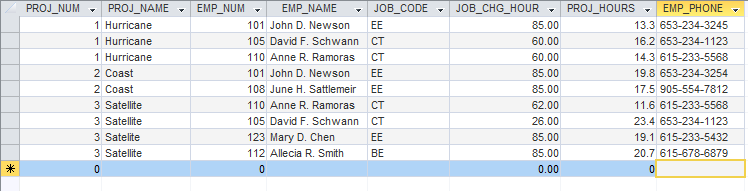
1

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

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

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

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)

***Although there is nothing to suggest a MANDATORY RULE of any kind given this set of data, but from observing the attributes and values of entries in rows and column fields it would seem like the following functional dependences exist,***

1. ***Between Project Number and Project Name***
2. ***Between Employee Number and Employee Name***
3. ***Possibly also 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 exist in the entry of information in the tables for some stuff like the name of project (listed in multiples places) and the employee names (also listed multiple times).***

***Besides potentially causing problems like update or insertion anomalies, the functionally dependent data related to this redundancy is from storing all data in a single table (and also leads to the deletion anomaly).***

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)

1

Employees

Employee

Employee

Employee

Employee

Employee

Departments

Department 1

Department 2

Department 3

Department 4

Dept Managers

(optional)

Man\_Department1

Man\_Department2

…

1

ABC Divisions

Division 1

Division 2

Division 3

1

M

M

1

***Department Managers could optionally not be an entity if a field attribute instead of employee, 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)

Customers

Customer1

Customer2

etc.

Books

Catalogue of many books….

M

M

***ERD accounts for the many-to-many relationship that exists.***

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

1

M

Flights

Flight #\_A1001

Flight #\_B1001

Flight #\_C1001

Airliner

Company\_1

Company\_2

etc.

***The ERD accounts for the one-to-many relationship between the airline and its outbound/inbound travel via flights***

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)

M

1

Regions

Region 1

Region 2

3

4

Factories

Factory

Factory

Factory

Factory

Factory

Employees

Employees

Employees

Employees

Employee

Employee

1

M

***This ERD accounts for the one-to-many relationship for between Regions and Factories, and one-to-many between Factories and employees. (an implied statement about 1-to-many relationship between regions and employees if needed could be added). However I did not add it because that is implied already with Factory entity relationships.)***

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

M

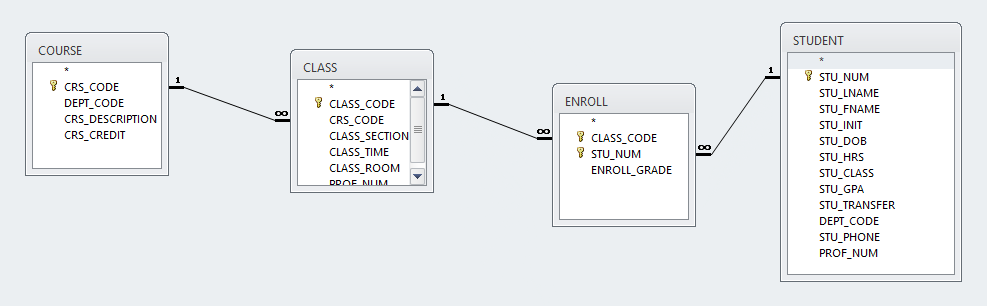
M

Degree

Employee

***This ERD accounts for the many-to-many relationship present in the description between employees and degrees.***

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 to be among rows and columns of the tables themselves. Therefore, without assuming what data may exist in relation to tuples or rows (entries into the table) what is clearly the case is these tables create a good structure for the data model to exist that could allow for many relationships among the entities, attributes, and even some constraints.***

***Business Rules for the Tiny College Data***

* Courses at the college can have several classes (of students) but each class must only cover a single course (1-to-many relationship exists).
* Each class can enroll a class full of students (many) , and since students may enroll is several classes the relationship existing is many-to-many among them.
* Another rule is for enrollment to occur, which requires a pairing or matching of Class codes from the Class entity with Student Numbers from the Student entity to populate using paired foreign keys the enroll table.
* Enrollment grade is also entered whenever creating a enrollment entry in the field attribute for the enroll junction table where student and class have already been mentioned.
* More relationships among entities exist, such as one-to-many for courses and enrollment relating the enrollment data with course offerings.
* The use of junction table for enrollmnet data creates new relationships that are 1 to many for enrollment separately with each of the other entities Class and Student, meaning a relating row from the student or class table may exist more than just one time within enrollment table (i.e. enrollment produces a list for students class schedules). NOTE: The existence of a junction table has implications for entity relationships that cause a high degree of funny stuff to occur.
* The courses and enrollment table which do not relate directly have a many-to-many relationship since basically for each course there are many classes, meaning many enrollments (of students) for that course (via the class offerings).
* Each enrollment entry should also match only one class to a student and therefore one student to one course (for just that enrollment entry).
* Students may enroll in many courses and many classes, but a constraint of this situation that may need to be addressed in DBMS could be if students signed up for multiple sections of the same course, which is not for students usually, but reserved for teacher entities who can correctly get assigned to several classes of the same course offering.
* Other requirements for business rules would include specifying the attributes of each field in the tables listed, and also discovering new entities (tables) to add into the ERD. A further exploration for constraints could help as well.

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

Class

Course

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

+has a

+is a

+is a

+has a

+is a

1…\*

1…1

0…\*

1…1

+has a

Student

Enroll

Class

Course