BUAN 6320 Database Foundations

Spring 2019

Instructor: Dr. James Scott

Assignment #1 – Intro SQL

General Instructions

- Students may study together for the assignment and review each other's completed work
- Students must each complete the assignment by their own hand
- Please use the provided word document template
- Please save the completed word document into PDF format before uploading
- Please submit the PDF file electronically through eLearning before the due date and time
- Do not include output – only the SQL
- **Use table aliases for all tables in all queries (unless otherwise specified)**
- Column aliases are required for all derived columns including aggregate columns (unless otherwise specified)
- Do not use column aliases unless required as stated previously
- If a problem does not ask for a specific sort order, use your best judgement to add a sort order

Chapter 1 Problems – Database Systems

Do Problems 1-8 on page 32 of our textbook using the data provided in the book figures.

1) How many records does the file contain? How many fields are there per record?
   There are 7 records. There are 5 fields per record

2) What problem would you encounter if you wanted to produce a listing by city? How would you solve this problem by altering the file structure?
   City is present in MANAGER_ADDRESS. The problem is, to get the city we need to break the MANAGER_ADDRESS into address, city, state, zip it. We can do the internal string search but takes longer type to execute. This can be overcomed by taking City as a separate column.

3) If you wanted to produce a listing of the file contents by last name, area code, city, state, or zip code, how would you alter the file structure?
   To get the last name we need to decompose the PROJECT_MANAGER string into Last Name and First Name, Intial

To get the City, State, Zip Code we need to decompose the MANAGER_ADDRESS string into different attributes
To get the Area Code we need to decompose the MANAGER_PHONE into Area Code and Phone Number.
The file structure is altered as follows,
After decomposing them we need to take each of the decomposed strings into different attributes (columns)

4) What data redundancies do you detect? How could those redundancies lead to anomalies?
In the PROJECT_MANAGER column, we have some Managers like Holy B. Parker and George F. Doris is repeated which gives rise to redundancy. It means a manager can manage many different projects.
But if there is any change in one of the attributes related to the manager like address or phone number it need to be changed in all the entries, if it is not done properly this redundancies may lead to the anomalies.

5) Identify and discuss the serious data redundancy problems exhibited by the file structure shown in Figure P1.5.
The given file structure has many redundancies causing many anomalies.
For example if any change is made for a particular JOB_CODE like changing its JOB_CHG_HOUR, it need to be changed in its every entry otherwise causing anamolies.
If an employee is deleted from the file say John D. Newson, the information related to him w.r.t the project is also deleted causing loss ($85*19.8hrs = $1683) to the company.
In this particular data we can also see that same JOB_CODE 'CT' has different JOB_CHG_HOUR. So this file structure makes us difficult to update the anamolies and it is not possible to determine whether the changes are accurately reflected in each record.

6) Looking at the EMP_NAME and EMP_PHONE contents in Figure P1.5, what change(s) would you recommend?
EMP_NAME is composed of FirstName, Initial, LastName. So decomposing them into three different categories into EMP_FNAME, EMP_INITIAL, EMP_LNAME would help us organize employee data well.
Similarly EMP_PHONE is composed of Area code and Phone Number. So decomposing EMP_PHONE into EMP_AREACODE and EMP_PHNE would help us organize employees by their area code.

7) Identify the various data sources in the file you examined in Problem 5.
Employee Data (Names and Phone Numbers), Project Data (Project Name), Job Data (Project Hours, Charge per hour)

8) Given your answer to Problem 7, what new files should you create to help eliminate the data redundancies found in the file shown in Figure P1.5

The PROJECT file should contain project related details like project name, project manager, project budget etc., The EMPLOYEE file might contain employee names, phone number, address etc., The JOB file might contain charge per hour for each job types. The CHARGE file may be use to keep in track the number of hours by job type being billed for employee in a given project.

Chapter 2 Problems – Database Models

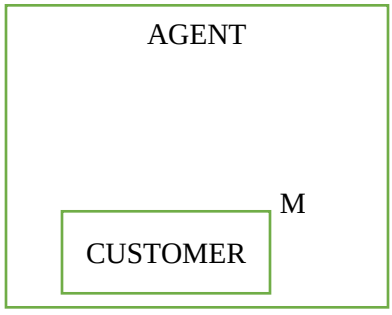
Do Problems 1-9 on pages 64-65 of our textbook using the data provided in the book figures.

Use the contents of Figure 2.1 to work problems 1-3.

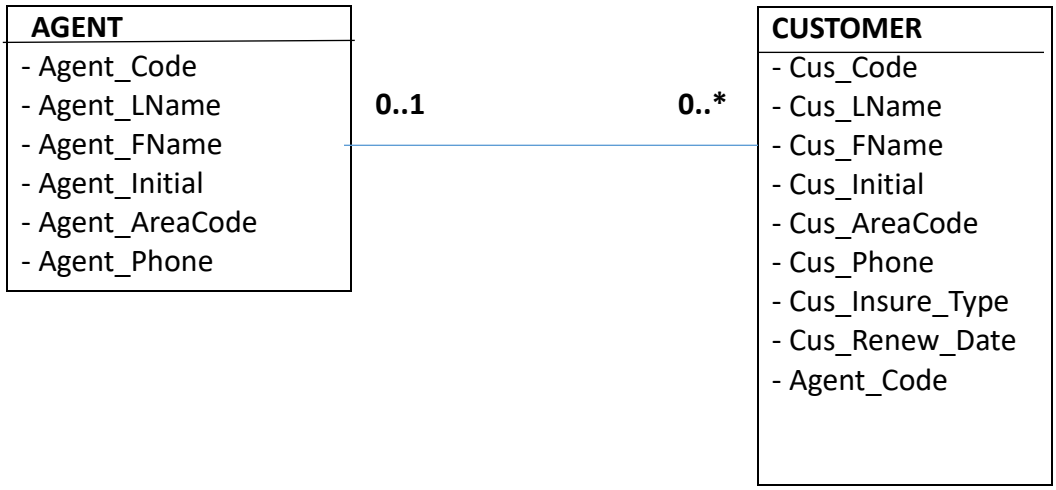
1. Write the business rule(s) that governs the relationship between AGENT and CUSTOMER
   By looking at the given data AGENT and CUSTOMER tables are connected through AGENT_CODE. In the two tables we observe that AGENT_CODE is occurring many times in the CUSTOMER Table. So there is a 1:M relation between AGENT and CUSTOMER.
   It means one Agent has many customers and many Customers can be under one single agent.
2. Given the business rule(s) you wrote in Problem 1, create the basic Crow's Foot ERD.

3. Using the ERD you drew in Problem 2, create the equivalent object representation and UML class diagram. (Use Figure 2.4 as your guide.)

   Object representation

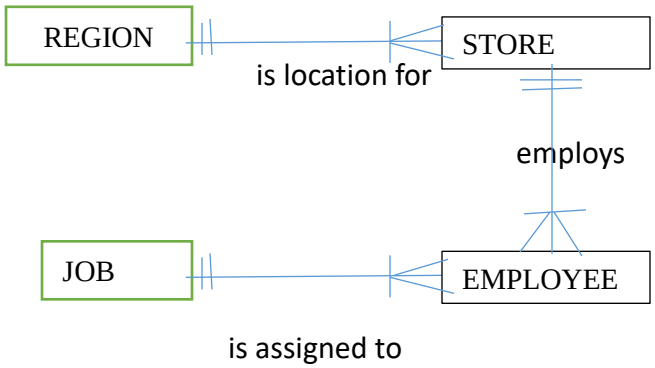


UML Class Representation



4. Identify each relationship type and write all of the business rules.
   One region may have many stores. Each store is located only in one region. 1:M relationship between REGION and STORE
   Each store has many employees. Each employee is assigned to one store. 1:M relationship between STORE and EMPLOYEE.
   One employee is assigned to only one job, but one job category can be assigned to many employees. So 1:M relationship between JOB and EMPLOYEE.

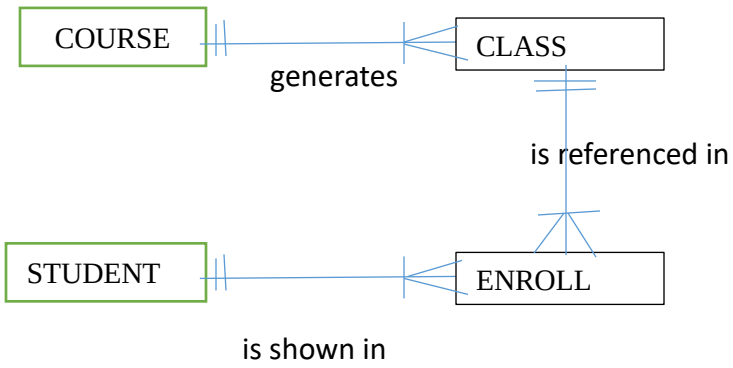
5. Create the basic Crow's Foot ERD for DealCo



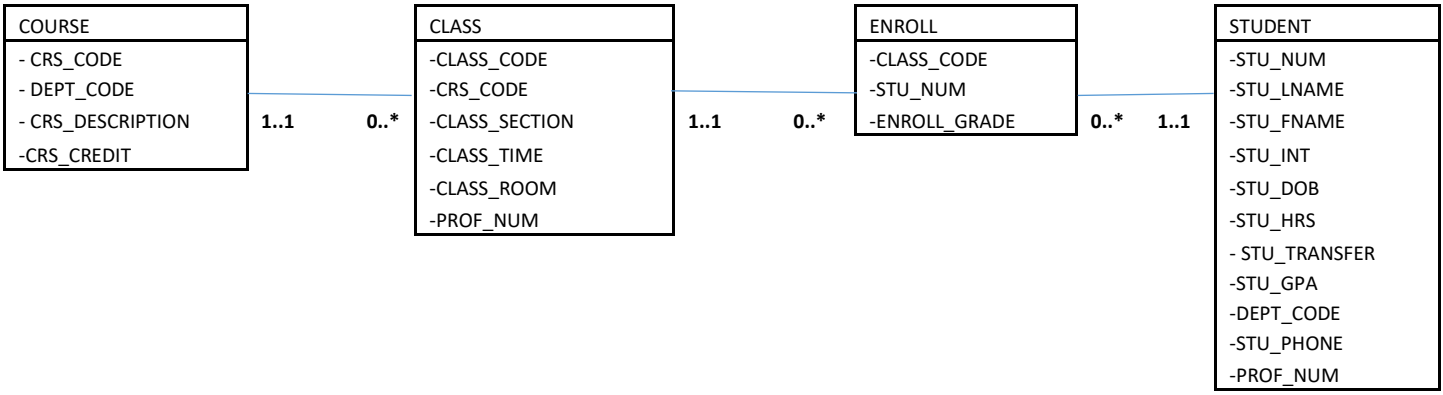
6. Identify each relationship type and write all of the business rules
   Each course can generate many classes. Each class is generated by one course. So 1:M relationship between COURSE and CLASS.
   One class can be enrolled many times. Each ENROLL refers to one CLASS. So 1:M relationship between CLASS and ENROLL.

Each student enrolls many classes, so has more than one entry In ENROLL. So 1:M relationship between STUDENT and ENROLL.

7. Create the basic Crow's Foot ERD for Tiny College.



8. Create the UML class diagram that reflects the entities and relationships you identified in the relational diagram



9. Typically, a hospital patient receives medications that have been ordered by a particular doctor. Because the patient often receives several medications per day, there is a 1:M relationship between PATIENT and ORDER. Similarly, each order can include several medications, creating a 1:M relationship between ORDER and MEDICATION.

   a. Identify the business rules for PATIENT, ORDER, and MEDICATION.

   For Patient
   Each patient may have many medical orders.
   Each medical order written for one patient
   For Order
   Each medical order may have many medications.
   Each medication can be given by many different orders.

For Medication
Each medication can be written in many orders.
Each medical order can have many medications.

b. Create a Crow's Foot ERD that depicts a relational database model to capture these business rules

For the following SQL coding problems use the UniversityDDL Database provided by instructor.

Problem #1 – Retrieving all rows and all columns from a table

For each of our tables, retrieve all rows and all columns.
Tables are Student, Faculty, Offering, Course, and Enrollment
(no need to sort at this point)

```
Select * from Student S ;
Select * from Faculty F;
Select * from Offering  O;
Select * from Course C ;
Select * from Enrollment E ;
```

Problem #2 – Retrieving a subset of columns from a table and sorting them both with and without the ASC keyword

Retrieve the student number, student first name, and student last name for all students
Sort the results by student last name then by student first name
Use the ASC keyword on the query

```
Select S.StdNo, S.StdFirstName , S.StdLastName
from Student S
order by S.StdLastName ASC, S.StdFirstName ASC ;
```

Repeat the query omitting ASC

```
Select S.StdNo, S.StdFirstName, S.StdLastName
from Student S
order by S.StdLastName , S.StdFirstName ;
```

Ascending Order is default, so results are the same for the both queries

Problem #3 – Retrieving a subset of columns from a table and sorting them on multiple columns mixing ascending and descending order, using both named and positional notation

Retrieve the student last name, student first name, and GPA for all students
Sort the results by GPA highest first, then by student last name, then by student first name
Use column names to sort (omit ASC)

```
Select S.StdLastName, S.StdFirstName, S.STDGPA
from Student S
order by S.STDGPA desc, S.StdLastName, S.StdFirstName ;
```

Repeat the query using positional notation

```
Select S.StdLastName, S.StdFirstName, S.STDGPA
from Student S
order by 3 desc, 1, 2 ;
```

Problem #4 – Retrieving columns from a table both with and without duplicates

Retrieve the student city and class for all students with duplicates

```
Select S.StdCity, S.StdClass from Student S;
```

Repeat query without duplicates

```
Select distinct S.StdCity, S.StdClass from Student S;
```

Problem #5 – Retrieving a subset of rows with a single Boolean expression

Retrieve the student last name, student first name, and GPA for all students with a GPA greater than 3.2

```
Select S.StdLastName, S.StdFirstName, S.STDGPA
from Student S
where S.STDGPA > 3.2 ;
```

Problem #6 – Retrieving a subset of rows with multiple complex Boolean expressions

Retrieve the student last name, student first name, and GPA for all students with a GPA (more than 2.2 and less than 2.7) OR (more than 3.2 and less than 3.8)

```
Select S.StdLastName, S.StdFirstName, STDGPA
from Student S
where S.STDGPA > 2.2 and S.STDGPA < 2.7
or S.STDGPA > 3.2 and S.STDGPA <  3.8 ;
```

Problem #7 – Retrieving a subset of rows with the BETWEEN operator

Retrieve the student last name, student first name, and GPA for all students with a GPA that is between 2.7 and 3.2 inclusive

```
Select S.StdLastName, S.StdFirstName, STDGPA
from Student S
where S.STDGPA between 2.7 and  3.2 ;
```

Problem #8 – Retrieving a subset of rows with testing for NULLs

Retrieve the offer number, course number, year, and faculty number from all course offerings that has not yet been assigned a Faculty

```
Select O.OfferNo, O.CourseNo, O.Offyear, O.FACNO from Offering O where O.FacNo is null ;
```

Repeat query for course offerings that have been assigned a Faculty

```
Select O.OfferNo, O.CourseNo, O.Offyear, O.FACNO from Offering O where O.FacNo is  not null ;
```