**Case assignment**

**Community library**

You recently get a job offer from a local community library. The manager asks you to develop a database system to track customers and books they check out.

**Task 1**

Your initial assignment is to create a table for customers. The manager tells you that this table needs to track the following information: all customers' social security numbers, first and last name, address, city, state and zip code, email, phone number, birth date, and date of applying for a library card. Your task is to first "sketch" a customer table (basically what data items or columns should be included in the table) using the Customer\_information table at background section as an example. Your table should contain the following:

* Column heading include all the attributes that need to be in this table.
* Populate the table with five records of data

**Task 2**

Your next task is to design one more table: Customer checkout table to track the books and CDs that the customer has checked out from the library. Customers and Customer Checkout table are related to one another.

The Customer table contains the basic data on each customer and the primary key is customer number. There is one row for each customer.

The Customer Checkout table contains the data for each book. The primary key consists of two pieces of data: Checkout number and customer number. One customer can check out 1 or more books/CDs at a time, but only one customer on a check out order.

Use the templates below to design the two tables.

**"Templates" for Table Design**

Customer Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Draw an ER diagram of your complete design for the community library. You can use Visio, or the Microsoft word feature to draw the chart, or write down the table schemata and describe the relationships among them.

Include half page report about your experience with this assignment.

**Case assignment expectations**

* Demonstrate understanding of using ER diagram to represent database design.
* Correctly design tables based on requirement

**Data modeling**

Data modeling is very important for database design. How the data is modeled will determine how the data will be accessed and manipulated. In module 1, we learned that there are various kinds of databases based on how data is modeled, e.g. hierarchical model, network model, relational data model, object-relational model, and object oriented model. The most popular data modeling is relational data model which is also the focus of this course. In a relational database design, generally speaking, the process is to:

* Decide on the purpose of this database and scope of the project
* agree upon the kinds of information you would like to retrieve from the database
* identify all relevant entities and relationships
	+ Note the difference between entities and attributes. For example, student is an entity (can also be called a table, or a relation), name and phone number are two of the attributes associated with the student entity.
* Use [ER diagram](http://www.sis.pitt.edu/~valeriab/1022-spring08/Chapter6.pdf) to describe the identified entities and relations
* Convert the ER model to a number of relation schemas
* Eliminate (or reduce) redundancy by splitting relations. This process is called [normalization](http://databases.about.com/od/specificproducts/a/normalization.htm)

Each relation (also called table in relational database modeling jargon) contains a collection of values associated with these attributes. For example, in module 1, we have the following Customer Information table.

Customer\_information

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Customer ID** | **Name** | **Phone number** | **Email**  | **City** | **…** |
| 0000001 | John Wayne | (650)-718-9920 | jwayne@hotmail.com | Mountain View |  |
| 0000002 | John Smith | (714)-110-8901 | jsmith@yahoo.com | Long Beach |  |
| 0000003 | Mary Jane | (562)-456-3490 | mjane@gmail.com | Los Alamitos |  |
| … | … | … | … |  | … |

In this module, we are going to use this table to introduce concepts that are important to database.

* Tuple (record): Each table has a set of tuples which are an ordered list of values. In this case, one tuple is

0000001 John Wayne (650)-718-9920 jwayne@hotmail.com mountain view

* Attributes: Each table has a set of attributes that describe this table. For example in this case, Customer\_ID, Name, Phone number, Email, and City are the attributes for Customer\_information table. Each attribute has different data types such as varchar, float, etc. All attributes must have unique names in the table.
* Primary key: Each table has an attribute or combination of attributes that could unique identify one record from the others. For example: Customer\_ID can distinguish one student from others even if they have the same name.

**Database design issues**

Pay attention to the following when you design a database

* Each field should have one discrete data. For example, it is better to have separate fields for street number, street name, city, state, and zip code than combining them into one field. Otherwise, it will be hard to retrieve one piece of data such as city when it is mixed with other data.
* No missing value is allowed in primary key.
* There should be no orphan tables in the database. Each table should be related to one or more than one table. When two tables are supposed to be related, there must be a field that relates the two databases.
* There is no need for each table to connect to every other table in the database.
* When two tables are in a one- to-many relationship, the primary key from the "one" table should be inserted as a foreign key in the "many" table, but not the other way around.
* All field names must be unique in each table. However, DBMS allows same names in different tables. However, it is good practice to create unique names for all tables to make it easier to write query.
* Make field name as descriptive as possible. Field1 and Field2 do not make much sense while writing queries.
* Set up build in Constraints to ensure data are entered correctly. For example: phone number and zip codes should fit in specified format.
* Referential integrity needs to be enforced so that there are no orphans in the table. When two table are in relationship, their data need to be checked when they are created or deleted.