MIS 341 Relational Database Design 1

Term Project

# Overview

The term project for this course will be to design and construct a database that uses appropriate Transact SQL procedures and functions. The database can be of your choice. It can be a new type of business or an existing business that you feel you could design a more functional database. With that said, I do require that the data types, columns and tables have a unique naming convention.

The main purpose of this project is to gain hands on experience on identifying business goals and rules in order to creatively and critically think about feasible solutions. Once a high overview objective has been documented it will be required to identify the constraints, columns, and tables that can be illustrated through an ERD diagram. It is expected that your database will consist of at least 8 tables and recommended that it has fewer than 20 tables.

I will NOT accept a database idea that is not your own. That is, you cannot just use a database design from the Internet that is already complete.

Be creative and have fun!

# Term Project: Objective 1 – 50 Points

At this stage you should have already identified your database ideas and submitted them to the instructor for approval. The objective here is to illustrate, explain, and expand on your overall pre-approved database concept. I have listed tasks that should help you filter out your ideas and help in identifying a visual (ERD) database design. The outcome of this part of the term project is the documentation needed to create the database which you’ll be asked to do in the next term project assignment.

Complete the following:

1. Select one of your database ideas that your instructor has approved.
2. In a Word compatible document, provide a written description of the problem your database is intended to solve. Describe the situation in which your database will be used. This description can be similar to what you submitted previously but you may need to add more detail or information.
3. In the same Word document, create a list of the business rules that affect your database design. Keep in mind that business rules that specify many to many relationships between entities are fine. It is those business rules that imply an intersecting table will be needed to resolve the many to many relationship. You will likely need to add to your business rules as your design progresses so be sure to update your list of business rules to reflect any changes in your ERD.
4. Use Visio or a similar product to create an ERD of your database design. Your database design should be normalized to at least third normal form and include the following. It is expected that your database will consist of at least 8 tables and recommended that it has fewer than 20 tables. The ERD should show:

* Table names
* Primary and foreign keys
* Identify cardinality for each relationship
* Specify with action verbs the forward relationship in all one to many relationships
* Resolve all many to many relationships

1. In the same Word document create your data dictionary as you did in Exercise 2 the first week of class. The data dictionary has the following information which may be shown using a Microsoft Word tables for each table in the ERD as was done in Exercise 2.

* The columns names for each table. (If the business rules from exercise 1 didn’t define appropriate columns for a specific table, add some columns that seem logical to you.)
* Give a description for each column.
* Specify the primary and foreign keys
* Specify the data type and size for each column. Select sizes that seem logical to you. Of course in a real world situation you would talk with users or look at documentation to determine the maximum size for a given column.
* Specify what columns that are not primary or foreign keys must be UNIQUE (it will be assumed all other columns don’t have to be UNIQUE).
* Specify what columns that are not primary or foreign keys cannot be NULL (it will be assumed all other columns can be NULL). Your project must have some columns that fit this.
* Specify any fields that should be indexed (frequently queried attributes). Your project must have some indexes.
* Specify default values for any columns where it seems to make sense. Your project must have some.
* Add check constraints. For each constraint write a business rule. Your project must have some.

You should specify most of the above using Word tables as in Exercise 2. You could use something like the following.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Column | Key | Unique | Not Null | Data | Size | Description |
| Name |  |  |  | Type |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Needs | Default | Check Constraints |
| Name | Index | Value |  |
|  |  |  |  |
|  |  |  |  |

* Specify Business Rules

You should specify the above by creating a list that is either numbered or bulleted

# Term Project: Objective 2 – 50 Points

Now that you have your database designed (Objective 1), you should be ready to develop your database through SQL coding. (Note: Your database should be based on your ERD and data dictionary that you developed in Objective 1.)

Through this process you may find that modifications to your tables or columns may be necessary. If you make changes to tables or columns be sure to modify your ERD, data dictionary and/or business rules to reflect the changes. At the end of the term you will be expected to hand in an up-to-date ERD, data dictionary and set of business rules.

Complete the following:

* Write the SQL that creates the tables, indexes and sequences(for userID) for your database. (Place all your SQL in the same script file.)
* Be sure to include all your constraints: primary keys, foreign keys, non-primary key columns that should be not null, non-primary key columns that should be unique, and check constraints. You must have some columns that should not be null and some that have check constraints.
* There must be at least one sequence within your database.
* Include any default values (For example: date, zip code or area code value) for columns specified in your design. You must have some in your project.
* Write the SQL statements that create the indexes you specified in objective 1. You must have some in your project.
* Write SQL statements to INSERT at least 4 or 5 records into each of your tables. You may want more in some tables. You want to have enough data that allows you to test queries and Transact SQL code that will be developed later. For example you may want more than 4 or 5 records in an intersecting table that exists to resolve a many to many relationship between two tables.
* Write SQL statements that verify the data has been inserted into your tables
* Test your SQL thoroughly before handing it in to the instructor.

# Term Project: Objective 3 – 100 POINTS

Now that you have written your SQL for your tables, constraints, indexes, and sequences and inserted some data (Objective 2), you need to create procedures, functions and triggers. All of these need to be well commented so that it is apparent by reading the comments what the purpose of the procedure, function, or trigger is. Read all of the tasks below before starting.

* Create a total of at least six procedures and functions. At least two of these six program units must be functions and at least three must be procedures.
  + If you find that you did not INSERT enough data into your tables in Objective 2 to test these procedures and functions properly, INSERT more data as it is needed.
  + The program units cannot just do more of the same other procedures. That is they can do an INSERT, UPDATE or DELETE for the three procedures but if you wanted to add a fourth procedure then it would have to be more complex. For example if one procedure did an update, the second ‘update’ procedure must contain other programming code (such as an IF ELSIF or loop structure) that fits that context.
* There are many options for the two or more functions. For example a function could return the zip code given the customer id. Or a function could return the health insurance carrier given the patient id.
* At least three out of the six program units must contain EXCEPTION handling. For instance a ‘try…catch’ blocks.
* In addition to creating the six program units, you need to create at least three triggers. One of the triggers should fire on an INSERT, one should fire on an UPDATE and the third should fire on a DELETE. For example you could create triggers that correspond to the INSERT, UPDATE and DELETE procedures mentioned above. The triggers could add information to a log table that is used to track changes to one or more tables.

The set of procedures, functions, and triggers you create is not expected to be an entire software application. You only need to meet the specifications above in this document. For example even though there would be many procedures to manage the data in all the tables, you do not have to write the code for them. You only need to write the code for the six program units and the three triggers. Be sure to thoroughly test all of your procedures, functions and triggers.

Upload a script file that contains the procedures, functions and three triggers. In addition upload the latest version of your script that creates your database. It is needed to test your program units. Also if you have altered your documentation in Objective 1 since you turned it in, upload the latest copy of it as well.