Tech Report 11

Introduction to Database Design

This report will focus on database design. Creating a database program involves a number of different steps. We will need to create tables that hold the data, develop forms that allow data to be easily entered and displayed, create queries that allow us to manipulate the data to get information out of database, and generate reports that provide the data in an easy to read format.

Differences Between Databases and Spreadsheets

As we start this section on database design and programming, many people wonder about the difference between a database and a spreadsheet. After all, In Tech Report 6 we used the LibreOffice Calc spreadsheet which used a table to enter information and calculate results. In Calc there is even a menu option titled 'Data' that offers some tools to work with the data in the spreadsheet. So why create a database?

When it comes to very simple data tables, spreadsheets such as Calc can be used. However, Calc (and spreadsheets in general) are designed primarily to perform calculations. Databases are designed to collect large volumes of data and manipulate that data. You might find that you start off collecting data in a spreadsheet, but as the data grows and becomes more complex, the need to convert to a database becomes apparent. The good news is that databases and spreadsheets are complimentary programs. Data extracted from a database may be used in a spreadsheet to perform complex calculations and create charts and graphs. Likewise, data from a spreadsheet can be imported into a database and the information used in the database queries to help generate reports.

A significant difference between the two programs is the way the user interacts with the software. In a spreadsheet the user is generally entering information directly into the spreadsheet. We did exactly this when we created the single station altitude calculator in Tech Report 6. With a database the user interacts with the data through a form or other program. The form can be used to check for data integrity before the data is written to the database tables (and this is what we will do as we create our database).

Another difference from a spreadsheet is that often the data itself is not stored on the same computer that is being utilized by the user. Instead the data is stored on a central server. This allows multiple people to use the database at the same time. The database has controls built in that make it possible for all of these individual users to access the data without corrupting the data. For smaller databases the data can be stored on the same computer but in a data file separate from the program, or the data and the program can be combined into a single file.

Introduction to Database Design

This report is designed to provide a basic introduction into relational databases and how to design a database. The time spent thinking about and working through the design of your database is quite important. A well thought out database will allow you to program the database quicker and more accurately, as you will have a road map to guide you. In the future you may want to expand on the database, adding new data fields and features. A well thought out database will allow you to do this much easier.

8-Step Design Process

We use an eight step process for designing a database. It is a process that we have used in the past and has worked well for us. These eight steps include:

1. What is the purpose of the database?
   What is the database supposed to do? If you don't have any idea of what your database is to be used for, you will not be able to design an effective, efficient database.

2. Information research and organization
   What information do you need to collect in this new database? As you begin to look at the available information, is it adequate or do you need to find additional information. Once you have the information, organize it.

3. Information is sorted into tables
   Start with the main subject of the each 'stack' of information. Each subject will become a table in your database (such as "Team Members" or "Rocket Fleet")

4. Information details become fields
   If the subject becomes the table, then the details that comprise the subject become the fields. So for a Team Members table you would have details like "Last Name" "First Name" and "Address" that become the names of the fields in the database

5. Identify the Primary Key
   As you sort the information into tables and fields, you will need to identify the key to each record. Often this becomes your Primary Key for the row. It can be as simple as adding an increasing, non-repeatable number to identify the record. What's even better is that the database can actually do this for you automatically.

6. Relationships
   Looking at your tables determine how various tables are related to one another. For example, you may have a table of club committee members, and they are composed of the various individual members of the club. This would mean the Committee Membership table and the Members table are related to each other. You can use the Primary key from the Members table to relate to a similar field in the Committee table. If you need to add a field or two to clarify things, then do it.

7. Improve the design of the database
   Look at your database design and see if it makes sense. If it does, you can even create a few test tables to see if it works as you thought it would. If it doesn't make the necessary adjustments.

8. Normalize the database
   Normalization is a process that helps avoid errors and duplication in the database. You should strive to accomplish at least the first three levels of database normalization.

Problems and Questions = Database Purpose

Most all databases start out as a way to answer a question or solve a problem. For example, if you are doing a research project with a team you may want to know how many people are on your team, what roles do they occupy, and are their parts of the project up to date or behind schedule. You might want to keep a record of your rocket collection, how many rockets you have, what engines do they use, and a record of each flight. Perhaps your club wants to hold a contest and you need to track all of the contestants, the events they have entered, and the results of each event as well as the overall combined event winners. These types of questions and problems are the ideally suited to the creation of a database.

Information

Once you know the problem you want to solve or the questions you want answered, it will point you in the direction of the information you need to collect. Similar types of information will naturally develop into tables. The information that makes up the tables becomes the fields.

Fields can sometimes be overly broad. Going back to our Team Members table, we didn’t just have a “Name” field, but instead broke it down into “Last Name” “First Name” and “Middle Name”. This kind of information parsing is critical in a database design. Breaking down information into its basic components will result in a database that is much more accurate and useful.

Keys

Relational databases, by their very nature, cannot contain duplicate rows of information. Each row must have a unique key that identifies that specific row, and this is called a Primary Key. Primary keys must always be unique and they must never be empty or null.

Sometimes a field of a table may qualify as a primary key. For example, if you provide every club member with a unique member ID that is never given to anyone else, that can be a primary key. If there is no field that naturally lends itself to being a Primary Key, most databases will create one using an ‘auto-increment’ field. This is a field that will start with a number (usually either 0 or 1) and increase that number by 1 for each row that exists in the database.

Relationships

The tables you include in your database will be related to other tables. These relationships allow you to keep your data consistent across all tables, and helps stop any ambiguity that may exist between tables. There are three different types of relationships that you can set up between tables:

• One-to-many
  One record in a table may be associated with many records in another table. If we go back to our club membership database planning, there will be a table identifying each member of the club (the ‘one’ in this relationship). But that member may be participating in several committees (the ‘many’ in this relationship)

• One-to-one
  One record in this table is associated with only one record in another table. Again, looking at our club membership planning, we know we may have a table that identifies all of the committees in the club. We could

also have another table that provides additional information on each committee (such as when they meet, what they are responsible for, etc.) Each committee is unique (the 'one' in this relationship) and each committee description applies to only one committee (the other 'one' in this relationship).

• Many-to-many
In this situation you have relationships that can involve many from one table having many with another table. Consider the example of authors and books. Each book may have a number of authors (the 'many' in this relationship). Each author may have also written a number of books (the other 'many' in this relationship). These relationships are very difficult to work with.

Improve and Normalize

As you begin the design of the database, you will likely find that you need to add additional fields (columns) to certain tables to make them more useful. You might also find that you can make the database more effective through the use of additional tables to store common information (such as state names and abbreviations) that you might want to 'lookup' at various places in the database. You may also find that you have a rather large table that can be made more efficient by splitting it into two separate tables. There will likely be other areas that you can improve on your database design.

During this time, you want to go through the process of 'normalizing' your database. You should at least go through the first three stages of normalization on any database design.

• First Normal Form (1NF)
A database has achieved 1NF if every cell within a database table contains a single value and not a listing of values. Additionally there should be no repeating of groups of fields. Lastly there should be a primary key for each set of related data.

• Second Normal Form (2NF)
To achieve 2NF the database design must first meet all the requirements of 1NF. Secondly, every non-key field is fully dependent on the primary key.

• Third Normal Form (3NF)
The database must be 2NF compliant and all fields are dependent on the primary key and no other fields in the database.

Any database that you design should keep these basic principles in mind. It doesn't matter what type of database you are developing. If it is a relational database, these principles apply.

If You Enjoy Rocketry, Consider Joining the NAR

If you enjoy model rocketry and projects such as the Arduino Launch Control System, then consider joining the National Association of Rocketry (NAR). The NAR is all about having fun and learning more with and about model rockets. It is the oldest and largest sport rocketry organization in the world. Since 1957, over 80,000 serious sport rocket modelers have joined the NAR to take advantage of the fun and excitement of organized rocketry.

The NAR is your gateway to rocket launches, clubs, contests, and more. Members receive the bi-monthly magazine "Sport Rocketry" and the digital NAR Member Guidebook—a 290 page how-to book on all aspects of rocketry. Members are granted access to the “Member Resources” website which includes NAR technical reports, high-power certification, and more. Finally each member of the NAR is cover by $5 million rocket flight liability insurance.

For more information, visit their web site at https://www.nar.org/