Database Your Way, LLC

"Databases by Design"
databaseyourway.com

Case Study #1
A relatively large, complex database

Situation:

The organization concerned is an Army National Guard Regional Training Institute (RTI). This is a self-contained facility. All personnel, operational and logistical support, as well as the instructor staff are integrated into one organization, all in support of training soldiers.

Scope:

This RTI trains over 4200 Soldiers a year, through conduct of 21 different courses of instruction. Most of the courses conduct multiple classes, for a total of, on average, 97 classes per year.

From the time a Soldier signs in to a course until the time he/she graduates, the organization is responsible for keeping track of the Soldier and the Soldier's progress. A need exists for a tool to help the organization do this more efficiently and with less chance of something important falling through the cracks.

In determining how best to serve the needs of the organization, everyone who has a direct need for data is consulted. This is the basis for good database design – determine who needs data, what data they need, and what form that data will take. Once the basic concept is established, we need to determine who will input that data and how.

To start with, we looked at the information needs the personnel section. The basic data needed was:

- Soldier's reservation status
- Quota Source
- Rank
- Name
- Social Security Number (SSN)
- Military Occupational Specialty
- Home address
- Unit address
- Higher Headquarters address
- Method of travel (air, private vehicle, government vehicle)
- Classroom assignment
- Billeting assignment
- Height, weight, body fat measurement (if required)

Microsoft Office Specialist

mark@databaseyourway.com 803-438-0372

Central Contractor Registration

From this data, the personnel section identified various reports they would need:

- Sign-in roster
- Height/weight roster
- Alpha roster (complete class)
- Squad rosters (a single class of a specific course may have 150 students, divided in groups of 18-20)
- Physical fitness test score cards
- Graduation certificates
- Any number of memorandums, ie denied enrollment, release from the course (for any number of reasons), and referred reports for marginal graduates.

A side-note here: The organization already knows, basically, what data they are in the business of collecting. Looking at the end result, that is, what the organization needs to do with that data in the form of reports, etc. may identify data needs overlooked in the first estimate. Identifying up front what reports the organization needed allowed us to revisit the data needs before actually constructing the database. This part of the process is very fluid.

From here, we looked at the information needs of the operations section.

- Soldier's name
- SSN
- Whether or not a particular Soldier reported
- If the Soldier reported, whether the Soldier was a graduate or nongraduate
- Number of tests in a given course
- Number of questions on each test
- Which, if any test questions are weighted
- Individual test scores

From this data, the operations section identified various reports they would need:

- Reserved vs. reported statistical reports
- Land navigation lane assignments
- Overall score reports
- By-squad score reports
- Retest required reports (roster of students failing a test, and required to take a retest)
- Academic release reports.

mark@databaseyourway.com 803-438-0372

Next, we looked at the information needs of the logistics section.

- Soldier's name
- SSN
- Squad assignment
- Weapon numbers
- Weapon type
- Weapon status (operational or not).

From this data, the logistics section identified various reports they would need.

- Weapons register
- Daily weapons inventory
- Hand receipts (a form for recording equipment was issued to a student, for accountability)
- Weapons status

Last but not least, we looked at the information needs of the instructor cadre. Pure and simple, the cadre train. They are accountable for the Soldiers assigned to them for training, and are responsible for evaluating those Soldiers and assessing their performance. Their data needs are those that allow them to identify, evaluate, and maintain records on the Soldiers assigned to them for training. As such, the basic data needs already identified by the other sections meet their needs, as well as some of the reports, for instance, the alpha and squad rosters.

We identified a number of specialized reports, and determined they could be built using the data already identified, such as name, SSN, squad assignment, etc. For example:

- Initial counseling statements
- Test failure counseling statements
- End of course counseling statements
- Performance evaluations (several different)
- Student folder labels
- Graduation rosters

Another side-note: I typically refer to all paper documents produced from a database as a "report", because in MS Access, that's what they are. A report can be as simple as a list, or as complex as formatted document. A good example of the latter is the AER described later in this document. Reports can be output as a single document of multiple pages (for example the Alpha Roster, which lists all students in a class) or simply multiple pages (for example, the Squad Assignment Roster, which lists all the students, but grouped by squad so that each group prints on a separate page.)

Microsoft Office Specialist

Back to designing the database!

This was enough to get started. Some of the needed data overlapped the various sections. Knowing this helps identify the core data elements, and gives an idea of how to design the database efficiently. From this point, we proceeded by using the mind-mapping process, with a piece of chart paper.

The core of it all is the student information. Based on the information each section identified as needed, a table was designed to accommodate the required core data, with the SSN used as the primary key. A separate but identical table would be built for each course. There were three main reasons for this: to make the database more efficient by keeping the tables smaller (thus speeding up indexing); to isolate the various groups of students, so that if one table was lost for whatever reason, it would not affect any other courses; and to allow for the fact that classes of different course may overlap, which obviously would cause a problem if everyone was in the same table.

Based on the other requirements, we designed a number of auxiliary tables. The idea was to specialize each table to serve its purpose, again with efficiency and isolation in mind.
Operations needed a table to hold scores for each separate course. Each scores table had to hold the initial score, a retest score, and a second retest score for each test. The tables had to accommodate the raw score (actual number correct) as well as the percentage score and the final average. (The score conversion and averaging was automated by query, but that's not part of the basic table design – that comes later.)

Some of the tables were built based on desired end results. For example, we know that individual unit addresses will be needed, so a table was designed to hold that information for each student, linked on the primary key of SSN. Again, following the principals of efficiency and data isolation, a separate table was created for each course.

Graduation documents as well as dismissal letters require higher headquarters information. Each student application is submitted through their higher headquarters. Each higher headquarters is identified by a specific and unique Quota Source. A table was designed to contain the higher headquarters unit designation, component (Army National Guard, Army Reserve, Active Army), addressee attention lines, and addresses for every entity. The Quota Source was used as the primary key.

Microsoft Office Specialist

Everything in the database was designed to serve a purpose. The best designed database starts with the concept of what the organization does, what data it keeps track of, and who needs something from the database. Really, the next step should be to look at the desired results. You've determined who needs something, so what form does that take? Design follows function.

As an example of design following function, we'll look at one desired end result of the database under discussion.

One end-result document is the Academic Evaluation Report (AER). (Remember mention of this earlier?) Without getting into too much detail at this point, there are various evaluation areas, and levels within those areas, in which each Soldier is evaluated (an 'X" in the box). There is also an area for bulleted comments in support or clarification of the "X"ed evaluations.

A multi-part approach was designed to achieve this. We had already determined we needed a basic Student table for each course, a table to hold unit information (a record for each student, within each course), and a Higher Headquarters table. Looking at the end-result document (the AER) what else did we need to produce this?

An Evaluation table was designed to hold the basic student information (name, SSN, squad assignment), with a column for each possible evaluated area and columns for up to seven comments. The primary key would be the SSN. The name is included obviously for quick identification, and the squad assignment is included for grouping purposes.

For ease and speed of operator input, the decision was made to avoid memo fields for the comment columns. Instead, a separate table was designed to hold pre-written comments. The Comments table also included columns to reference the performance levels addressed, a sorting category, and a comment number. When completing the evaluation, the instructor only has to put in the comment number. (Later in the design process, a report was designed listing all available comments, as a reference.) While this may seem to be a cookie-cutter process on the face of it, we started with approximately 800 comments to try to cover all conceivable situations, and the comments 'report' was actually a booklet. As new situations arose, or anyone had an idea for a good comment, they were added to the table. There are now over 1000 comments.

Microsoft Office Specialist

The AER also requires both a rater and reviewer name block, with name, rank, and title. For the same reasons as the comments area, a table was designed to hold the name, rank, title, course assigned, and a unique ID number for each staff member. The input operator (instructor completing the evaluation) need only enter the rater ID and reviewer ID number.

The next part of the process was to design a table that contained every element required for the final AER. This table would contain the following:

Data	Source
Last name, first name, middle initial	course student table
SSN	" "
Rank	" "
Military Occupational Specialty	" "
Course title	Courses table
Component	Higher headquarters table (update query assigns component based on Quota Source)
Duration of Course (from, to dates)	Courses table
Performance Summary (Exceeded, achieved, marginal, failed)	Evaluation table
Demonstrated abilities (Written comm, oral comm, leadership, group work, research – not evaluated, unsatisfactory, satisfactory, or superior)	Evaluation table
Bulleted comments	Evaluation table/Comments table (update query converts numbers input to comments)
Original mailed to:	Higher headquarters table
Unit of Assignment:	Unit table
Height/Weight, pass or fail	course student table
Physical fitness test, pass/fail	course student table
Rater block	Evaluation table/Instructor table (update query converts numbers to name block)
Reviewer block	Evaluation table/Instructor table

An append and/or update query was written to populate the Complete Eval table with all of the above, plus four additional date blocks.

The methods used to this point were used throughout the database design process. Though the reports identified early on are used to help design the underlying tables, the reports themselves are not typically built until the database structure is more or less complete.

Though creating the actual report, the AER, was not necessarily the next step, I will address that next to maintain the flow of thought.

The AER was not an ad hoc report, with the data displayed however the organization desired. The AER is an actual Department of the Army form, with a pre-defined layout. We started with an actual form, and measured the dimensions of the form and each block within that form to the 32nd of an inch, converted that to decimal, and started recreating it in Report design view.

Then, a query was designed to pull all the information from the table to make it available for the report (the AER). A few questions about the process may arise at this time. For instance, why not base the report directly on the table? One reason is that every student that enrolls receives an AER, whether or not they actually graduate. A non-graduate AER requires an additional report (known as a referred report) spelling this out in detail and explaining the rebuttal and appeal process. The query can be assigned criteria that only pulls the records of graduates, or conversely, non-graduates. A query also allows for on-the-spot input of the date (remember those 4 separate date fields?).

Another question may be about the need for this table at all. If we're going to use a query to fill the report, why not just do that from the start rather than build a whole new table that contains data that is already spread throughout the database? Archiving. A select query only holds the current data. This organization conducts multiple classes every year – a single course may conduct 15 classes. To prevent bloat, the tables, forms, queries, reports, macros, and modules are reused for every class. This means dumping the data from the old class to make room for the next. A table specifically designed to hold all the data needed to create the report can be archived in case the data ever needs to be reviewed, or the report needs to be recreated.

This is another good example of the organization's needs, or even possible future needs, guiding the design of your database. Database design starts with the immediate, but should incorporate the possibilities.

Now we have tables and reports. To get data from one table to another, to get data to populate a report, to change (update) data, we need queries. Queries are the engines of the database, and can vary tremendously in complexity, depending on what you're trying to do. Some of the things we used queries for were to select specific data, update specific data, and perform calculations. Some queries you want to run singly or independently, some you may want to run in sequence. That's where macros come in.

Macros are not just for running a series of queries, however. They can be used to run any sequence of events the database is capable of. Sometimes they are used for a single event, such as writing an import specification.

Import specifications are used to populate the tables in this database. In this case, data is taken from the Army Training Requirements and Resources system (the big database in the sky). It is downloaded in the form of an Excel spreadsheet. As this spreadsheet contains way more than we need, it is modified to remove unwanted data and non-data (garbage lines like header information), then saved as a text file. The import specification is written to look for this specific file, retrieve the data, and store it in a specific table.

Here's another example of looking outside the box in designing the database. By considering the sequence of data elements in the data source (that is, knowing what comes first, then second, etc.), the table was designed to efficiently accept this data.

In this case, by importing data from a known source, we met the basic data requirements of this organization – information about individual Soldiers. Much of the data could only be identified on site after the arrival of the Soldier; for instance, actual unit of assignment and address, height, weight, squad assignment, etc. Some data is collected throughout the course; for instance, test scores and performance evaluations.

Forms were designed to facilitate local data entry. The forms were based on the underlying table the data needed to go into, and organized in a manner that made sense to the people that would have to actually enter the data. Throughout the form design process, we always kept in mind the concept of "user friendly".

Microsoft Office Specialist

mark@databaseyourway.com 803-438-0372

Central Contractor Registration

Some of the forms were designed to accommodate more than just data entry, by utilizing command buttons to automate the process as much as possible. (Though it may not be politically correct, we also tried to incorporate the concept of "idiot proof".) By utilizing command buttons for such things as printing reports, and running queries and macros in the background, the database became much like any other application you would use on a computer, not requiring an in-depth knowledge of what was happening behind the scenes.

As this is a large organization with diverse components, the last thing developed was the switchboard interface. The main switchboard uses buttons to list the basic function of the database. Pick a function, click it, and you either go to that function or you're taken to another switchboard page to refine your options. This makes the database easy to navigate. A user who is only concerned with one aspect can navigate directly to that function without wading through all possible functions. The main switchboard page also incorporates the organization's logo.

Summary

Database design starts with communication. The people who know the nature of the company and what data needs to be tracked, the people who will (or might) need some output from the data, and the people who will be putting the data into the finished database need to be involved with the database designer in the process.

We identified the nature of the organization and the needs of the organization (consulting all the players involved). We vetted the data we needed to capture by looking at the end products desired, and made any necessary adjustments to the data needs. We designed tables to hold that data, remembering the principles of efficiency, isolation, and relationship (that's where the primary keys were identified, allowing this to be a relational database).

We designed reports that gave the end users a product that was useful to them, and developed the queries that would not only populate those reports with data, but make data transfer and modification relatively simple.

We designed macros and modules to facilitate import of data and to automate as many processes as possible.

We then designed forms as a user interface, not only making the database user-friendly, but attractive as well. The final set of forms designed were those of the switchboard for simple navigation to the desired function, resulting in ease of use, less user frustration, and more user efficiency by not having to waste time wading through the depths of the database.

The organization's situational statistics are given at the beginning of this case study, but as this is the summary here they are again:

- Courses of instruction 21
- Classes per year (ave.) 97
- Students per year 4200

The statistics for the database itself are as follows:

- Tables 94
- Reports 440
- Forms 223
- Queries 929
- Macros 140
- Modules 13
- Switchboard pages 85
- Switchboard items 428

The end result is a very complex database, serving the organization's requirements and still in use after seven years. Though it has evolved, as the organization has evolved, the core product is the same.

Most databases will not be this extensive or complex - database complexity is based on the organization's need. But the process is the same. This case study shows what can be accomplished.