

Database Design Basics

Lou Coraggio
Troy University

Database Terms

<i>Record 1</i>	FIELD 1	FIELD 2	FIELD 3
<i>Record 2</i>	FIELD 1	FIELD 2	FIELD 3
<i>Record 3</i>	FIELD 1	FIELD 2	FIELD 3

- Each data element will become **FIELD**
- A collection of **FIELDS** is known as a **RECORD**
- All **RECORDS** are grouped together as a **TABLE**

Each FIELD occupies the same space – even if there is nothing in it.

Sorting

<i>Record 1</i>	XXX	265	FIELD 3
<i>Record 2</i>	MMM	125	FIELD 3
<i>Record 3</i>	AAA	356	FIELD 3

New records go to the bottom of the table. We don't want to shuffle/search the whole table each time we need to find something.

We create a KEY Table and Sort/Search that instead. Just a pointer to the record in the main table.

Key On Field 1

#1 *Record 3*

#2 *Record 2*

#3 *Record 1*

Key on Field 2

#1 *Record 2*

#2 *Record 1*

#3 *Record 3*

Process Overview

1. Obtain data element definitions from Data Dictionary (or equivalent)
2. Define data types for each Field
3. Define data validation rules
4. Organize elements into Records
5. Define data relationships
6. Organize Records into Tables
7. Establish sort orders (keys)
8. Define Table relationships
9. Establish referential integrity constraints

Basic Data Types

- **String**
- **Memo**
- **Integer**
 - Byte
 - Short
 - Long
- **Real**
 - single
 - double precision
- **Logical (true/false)**
- **Binary (images)**

STRINGS

- **Fixed length (usually 1-255)**
- **may be any of the ASCII characters**
- **1 Byte per character storage**
- **sort by ASCII code**
- **May be case sensitive**
- **Can be easily searched & sorted and compared**
- **Best for shorter fields like names, addresses**

Memos

- **Variable length (256 to 65000 typical)**
- **Expand to fit**
- **1 byte per char plus overhead**
- **Must pay attention to formatting characters (tabs, spaces, line feeds)**
- **Require special treatment on screens, reports**
- **Tough to search, process, or equal**
- **Best for notes, free text**
- **Not sortable**

Integers

- **Whole numbers (no fractions)**
- **Long (4 bytes)**
 - Range: 2,147,483,648 to -2,147,483,647
- **Short (2 bytes)**
 - Range: -32,768 to 32,767
- **Byte – 0 to 255**
- **Math & Sorting extremely quick**
- **Very compact storage**
- **Use whenever practical**

Real Data Type

- **Also known as Decimal or Float**
- **Double Precision (default)**
 - Eight bytes to store
 - Range: $\pm 2.225073858507201e-308 \dots \pm 1.79769313496231e+308$
 - 15 significant digits
- **Single Precision (not in common use today)**
 - Range: $0, \pm 1.175494e-38 \dots \pm 3.402823e+38$ (6 significant digits), Four bytes
- **Subject to rounding errors**
- **Slower math and sort**
- **What does = really mean?**

Other Data Types

- **Logical (True/False)**
- **Binary**
 - Use for pictures, drawings, audio etc
 - Very environment specific for storage and handling
 - Typically better to store a pointer
- **Date**
 - May include a time component
 - Environment specific (MS, Linux ,Oracle)
 - Note how you want to store

Validation Rules

- **Based on reality where possible**
 - Student age should be more than 16
 - CC Numbers should have 16 numeric digits
- **Always better to use a list than free entry**
- **May be absolute or “confirm please”**
 - Student may be younger than 16 but not likely
- **May use redundancy to ensure accuracy**
 - Birthdate & Age may be asked for
- **May use a format mask to indicate**
 - (###)###-#### for a phone number
 - DD/MM/YYYY for a date

Field Attributes

- **Size (Strings/Memos)**
- **Required (must be filled in)**
- **Format Mask (###-##-#### for SSN)**
- **Unique**
 - can multiple records have the same value
 - SSN is a good example
- **Case Sensitive (Strings)**
 - Are “LOU” and “Lou” and “lou” the same?
- **Value Range (Integers)**
 - Also helps pick the right data type
- **Must be in a List (a menu)**

Key Attributes

- **Fields to be included**
 - May be more than one
 - Each may have sort order (Ascending/Descending)
- **Unique (Key may have only one instance)**
 - Individual fields may be duplicated
 - Multiple First Name + Multiple Last Name + Multiple Birth Date = Unique Key
- **Auto ID – System will automatically generate a unique identifier (usually a long integer)**
- **Ignore Blanks –**
 - Don't Sort/Show fields with a Null value
- **Primary – Must be non-null, unique (often AutoID)**

Data Relationships

- **1:1 Each of these has exactly 1 of those**
 - Each Student has one SSN
- **1:Many Each of these may have Many of those**
 - Each Student has Many Classes
 - Each Class has Many Students
 - Each Teacher has Many Classes
- **Many:Many All of these may have Many of those**
 - Many Teachers have Many Students
- **Often referred to as Parent-Child**
 - Each Parent has Many Children (1:Many)
 - Each Child has Many Parents (Many:1)
 - Many Children have Many Siblings (Many:Many)

Designing Tables

- Use 1:1 to put fields into a single table
- Use 1:Many to define child tables
- Each pair of related tables must have at least one pair of fields in common
- Many:Many Relationships require a third table to join the two together
- Sort fields are known as Keys
- Joining tables are usually done with Keys

Database Relationships

STUDENT TABLE		
<i>SSN</i>	Name	Deg
1234	Lou Dude	MBA
9876	Mia Sad	MSM

Course Assignments		
Prof	Class	<i>Sect</i>
Dr. Lou	MIS 666	3325
Dr. Lou	MIS 666	2345
Dr. Lou	MBA 777	7788
Dr. Smif	MSM 888	4475

DEG-CLASS TABLE	
<i>Deg</i>	<i>Class</i>
MBA	MBA 777
MBA	MIS 666
MBA	MSM 888
MSM	MIS 666
MSM	MSM 888

CLASS DETAIL TABLE		
<i>SSN</i>	<i>Sect</i>	Grade
1234	2345	A
1234	7788	B
1234	4475	A
9876	3325	C
9876	4475	B

Two Unique, 2 Field Keys
in these tables

What Can We See

- **For Each Student**
 - Profs
 - Classes
 - Sections
 - Degree Req
- **For Each Degree**
 - Students
 - Classes
 - Profs
 - Sections
- **For Each Prof**
 - Degrees
 - Classes
 - Sections
 - Students

Referential Integrity

What happens to children when key fields on the parent file are changed/ deleted?

Possible Actions

- **Nothing** – Don't change /delete the children
- **Cascade** – Change (or delete) all of the children when the parent is changed
- **Restrict** – Do not allow changing or deleting the parent when children exist
- **Clear** – Set all the values of the child fields to null (typically in conjunction with the "Ignore Blanks" attribute of a Key Field)

Multi User Environments

Suppose two users want to work on the same record? How do we deal with that?

- **ADD** is no problem – first come, first served
- **DELETE** no problem – if X deletes while Y is working, Y just adds back the new record.
- **CHANGE** is tougher to deal with. Simple strategies that don't work:
 - **Hold/Lock the record** – bad boogie, people go off to lunch, go home. Nobody else can use.
 - **First Come First Served** – First user is overwritten by the second. (ATM transactions?)
 - The "**Deadly Embrace**"

The Deadly Embrace

- This occurs when two workstations attempt to hold the same record.
- I have the Customer Record held and need the Order record.
- You have the Order Record held but need the Customer record.

What happens now?

Handling Multi User Change Issues

- Save a Copy of record when opened.
- Do Changes
- Re-Read record now in the database
- Compare Copy to Database.
- IF the same
 - Lock, Change DB, Release.
- If DB now different from Copy
 - Inform user
 - Display the DB Record for additional changes

MINIMIZES LOCKING TIME & "OVERWRITES"

Issues For Non Techies

- **GIGO – Validation is context sensitive. Should be done before entry.**
- **Every “Real Time” key slows down the system. For infrequently needed sorts do it “On Demand”.**
- **Compacting databases saves on both storage and network traffic. Avoid “just in case” mentality for sizing fields.**
- **Database design should be done around dominant dataflows & processes. Design to meet high volume needs.**
- **Always good to add a couple of flag fields for later use. (I-9 at U of A)**

Issues For Non Techies (2)

- Aliases from Data Dictionaries are often good variable names
- MS Access is not a multi user database, but is a good prototyping tool
- Archive old data periodically
- Never type it more than once
- Never type it when you can pick it from a list
- Use pattern masks whenever possible
(what day is 03/08/05?)
- Real (Floating Point) numbers are subject to rounding errors. Can't carry the federal budget to pennies. Don't try.