What is a Database?
- A Database, like a Spreadsheet, is a way of structuring information in order to solve problems,
- Unlike Spreadsheets, Databases are much more restrictive in where information can go,
- Those restrictions make it easier to ask certain types of questions (Queries)

Spreadsheets vs. Databases
- Spreadsheets: good at "Figuring Things Out"
  - Networks of interrelated calculations, where a change to one cell causes a ripple effect across the rest of the sheet.
- Databases: good at "Looking Stuff Up"
  - Searching existing tables for information that matches the query,
  - Joining (synthesizing) two or more tables into one.

One Skill You Will Develop
- When do you use a Spreadsheet?
  - Lots of calculations,
  - "What-if" scenarios,
  - Few cases of looking things up in existing information.
- When do you use a Database?
  - Looking for existing items that match queries,
  - Synthesizing new information from old,
  - Few calculations.
- Sometimes it's hard to tell which one to use.
- What are the penalties for guessing wrong?

Types of Databases
- Hierarchical
  - File system on your computer
- Network
  - Distributed Data across widely separated nodes
  - The Internet
- Flat-File
  - All information stored in one table
- Relational
  - Multiple tables plus relationships between tables
  - Anything a Flat-File can do, a Relational can do.

Tables in Flat-File & Relational
- Tables are made up of Records (numbered rows),
- Records: all information about a single object,
- Records are made up of Fields,
- Fields have:
  - Name (set by designer)
  - Data Type (set by designer)
  - Size (set by designer or determined by data type)
  - Default Value (optional)
  - Validation Rule or Validity Check (optional)
Here's a Sample Table

Data Types in Microsoft Access

- **Text** (1...255 characters, size set by designer)
- **Memo** (up to 64K characters of free-form text)
- **Date/Time** (8 bytes, like date/times in Excel)
- **Currency** (8 bytes, BCD numbers)
- **Yes/No** (1 bit, two-valued data)
- **Numbers** (lots of subtypes, next slide)

Number Sizes (SubTypes)

- **Byte** (one byte), integer, 0..255,
- **Integer** (two bytes), integer, -32768...+32767,
- **Long Integer** (four bytes), integer, ±2 billion,
- **Single** (four bytes), floating point, ±10\(^{±38}\), 5-6 decimal significant figures,
- **Double** (eight bytes), floating point, ±10\(^{±308}\), 15-16 decimal significant figures.

Here's the Sample Table Again

A Note About "Fields"

- Fields have two definitions, a precise definition and a sloppy definition:
  - The precise definition is that a field is a component of a record, as in "The Children field of record 12" (one datum),
  - The sloppy definition is that a field is a column of a table, as in "The Children field of the table" (the whole column).
Let's Build a Table about People

- We have to decide how many fields we need.
- For each field we need:
  - Name (should describe the field contents)
  - Type (Text, Number, Currency, Yes/No, etc.)
  - Size (where we are allowed to do so)
  - Default Value (common value for new records)
  - Validation Rule (constraints on field values)

A Rule to Follow

- When deciding Data Type or Size, pick the smallest one that entirely covers the expected range of values.
- Too small and not all values can be covered,
- Too large and you are wasting space.

Field: Last Name

- Type: Text
- Size: ?? (Make a guess: 15? 20?)
  - Too few characters cuts off names
  - Too many characters wastes space
  - You can always widen/narrow the field later
  - Caution: characters lost when narrowing field are gone forever
- No Default Value or Validation Rule

Field: First Name

- Type: Text
- Size: ?? (longest first name that I see on a regular basis is "Christopher" – 11 characters, I usually even it out at 12 characters)
- No Default Value
- No Validation Rule

Field: Middle Initial

- Type: Text
- Size: 1
- No Default Value
- No Validation Rule

Field: Salary

- Type: Currency
- Size: We don’t get a choice (8 bytes)
- No Default Value
- Validation Rule:
  
  \[ \text{>=20000 AND <=100000} \]
  
  (prevents stupid data entry errors such as 5000 or 500000)
<table>
<thead>
<tr>
<th>Field: <strong>Birthdate</strong></th>
<th>Field: <strong>Children</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Type: <a href="https://example.com">Date/Time</a> (Not Text!)</td>
<td>- Type: Number</td>
</tr>
<tr>
<td>- Size: We don't get a choice (8 bytes)</td>
<td>- Size: Byte</td>
</tr>
<tr>
<td>- No Default Value</td>
<td>- Default Value: 0 (how many kids assumed?)</td>
</tr>
<tr>
<td>- Validation Rule:</td>
<td>- Validation Rule: &lt;=20</td>
</tr>
<tr>
<td>&gt;=1/1/1949 AND &lt;=12/31/1996</td>
<td>(no lower limit needed because Byte enforces a lower limit of 0, but we do need to pick an upper limit)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field: <strong>Married (option 1)</strong></th>
<th>Field: <strong>Married (option 2)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Type: Yes/No</td>
<td>- Type: Text</td>
</tr>
<tr>
<td>- Size: We don't get a choice (1 bit)</td>
<td>- Size: 1</td>
</tr>
<tr>
<td>- Default Value: No</td>
<td>- Default Value: S</td>
</tr>
<tr>
<td></td>
<td>- Validation Rule:</td>
</tr>
<tr>
<td></td>
<td>=&quot;S&quot; OR =&quot;M&quot; OR =&quot;D&quot; OR =&quot;W&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field: <strong>Sex</strong></th>
<th>Field: <strong>Score</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>- Type: Text</td>
<td>- Type: Number</td>
</tr>
<tr>
<td>- Size: 1</td>
<td>- Size: Single</td>
</tr>
<tr>
<td>- No Default Value</td>
<td>- No Default Value</td>
</tr>
<tr>
<td>- Validation Rule:</td>
<td>- Validation Rule:</td>
</tr>
<tr>
<td>=&quot;M&quot; OR =&quot;F&quot;</td>
<td>&gt;=0.0 AND &lt;=100.0</td>
</tr>
</tbody>
</table>
### Structure vs. Data

- A Table can:
  - Exist
  - Have structure
  - Take up space on the disk
  - Yet contain no data (be empty)
- Only when the structure has been defined can data be added
- The structure can be modified after data have been added.

### Running Queries Depends on the DB

- dBase used text-based pseudo-English sentences:
  - LIST LastName, FirstName FOR Salary < 40000 AND Children >= 2
  - Requires only very modest computing power
- More modern databases (MS Access) use QBE (Query By Example) graphical screen-based forms to give examples of what is being searched for.
  - Easier to use than typing sentences
  - Requires more powerful computers to do gracefully.