Database Basics
and
An Introduction to the
Relational Database Model

Objectives
- Brief history of:
  - Data storage
  - Problems with storage before RDB
  - Databases and relational database development

- Brief coverage of:
  - Relational Database Technology
    - Schemas
    - Entities
    - Attributes
    - Key Structures

- Concepts to be demonstrated in lab and Project through use of MS Access

Why Know About Databases?
- Much of the data people keep is not stored as text, but is instead organized in the form of tables.
- Knowing how the data is structured and becoming proficient at retrieving and manipulating it is a very powerful skill to have.

“If spreadsheets are the ‘number crunchers’ of the digital world, databases are the real ‘information crunchers’. Databases excel at managing and manipulating structured information.”

So what is structured information?

-Rose Vines
GeekGirls.com

Storage of Information
- Your address book
  - Names, phone numbers, addresses

- Books available in a library or bookstore
  - Title, author, subject, availability

- Business
  - Employee information, product information
  - Accounting and financial data

- Hospital
  - Medical records and insurance information

- University of Washington
  - Student grades, tuition payments, classes offered
Data Storage – Flat Files

- Early days of computing 1950’s to 1980
  - File based data storage
  - Flat files
  - Programs had to be written to:
    - Read data from a file on the disk
    - Process data (enter student grades, update employee salary)
    - Write data back to the file
- But there are problems with file based data storage
  - They deal directly with how we think

Problems with Flat Files

- Need to write a program to get at data
  - Too hard to get to data from a business perspective
    - Time consuming and expensive
  - Too easy for anyone who writes the program to get to data
  - Data dependency
    - If you change format of any of data in flat file system, then you are forced to change all programs that access the data
      - Zip codes changing from 5 to 9 digits
  - Don’t know which programs are using the data
    - Files full of data on a disk don’t tell you who accesses them

Main Problem Is Redundant Data

- Redundant data leads to data anomalies
  - Integrity and update anomalies
- Bad data leads to bad decisions

Relational Database Technology

- Data is stored in tables
  - Tables are a common way to present information
  - Informal tables must be normalized to be database tables
- To normalize a table for use in a database
  - There should be one table for each type of object or “entity”
    - Thinking about the informal table already presented, each of the following would be given its own table structure:
      - Student table to list all students – one row for each student
      - Course table to list all courses – one row for each course
      - Instructor table to list all instructors – one row for each instructor
      - Class, Event table to show student, instructor and course instance associated together at a certain place and time
    - Each row has a unique identifier
      - Primary key (PK)
    - Primary key is never null and cannot be a duplicate
The Keys Are the Key to Relational DB’s

- Keys show how tables are related to one another

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>SID</td>
<td>Name</td>
</tr>
<tr>
<td>1</td>
<td>Paul Stevens</td>
</tr>
<tr>
<td>2</td>
<td>Holly Eggelston</td>
</tr>
<tr>
<td>3</td>
<td>Stephanie Wright</td>
</tr>
<tr>
<td>4</td>
<td>Lisa Spagnolo</td>
</tr>
<tr>
<td>5</td>
<td>Pam Green</td>
</tr>
<tr>
<td>6</td>
<td>Thomas Nguyen</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Course</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIT 100</td>
<td>MGH 420</td>
</tr>
<tr>
<td>LIS 540</td>
<td>EE1 045</td>
</tr>
<tr>
<td>INFO 480</td>
<td>EE1 025</td>
</tr>
</tbody>
</table>

Note: No Data Redundancies

Terms of a Database Table

- Name the structural parts of database tables

<table>
<thead>
<tr>
<th>Tuple (Row)</th>
<th>Table Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SID</td>
<td>Name</td>
</tr>
<tr>
<td>1</td>
<td>Paul Stevens</td>
</tr>
<tr>
<td>2</td>
<td>Holly Eggelston</td>
</tr>
<tr>
<td>3</td>
<td>Stephanie Wright</td>
</tr>
<tr>
<td>4</td>
<td>Lisa Spagnolo</td>
</tr>
<tr>
<td>5</td>
<td>Pam Green</td>
</tr>
<tr>
<td>6</td>
<td>Thomas Nguyen</td>
</tr>
</tbody>
</table>

Note: The database key is a field(s) that is unique for each row. Here it is the SID field.

Attributes have types – SID is an Integer, Name is a string

General Database Terminology

- The structure of a database is its database schema
- The schema specifies:
  - The list of tables forming the database
  - For each table, the fields of its records
  - For each field, its properties, i.e. data type, key or not key, default value, etc.

<table>
<thead>
<tr>
<th>Student</th>
<th>Integer</th>
<th>student UW ID #</th>
</tr>
</thead>
<tbody>
<tr>
<td>FName</td>
<td>String</td>
<td>student first name</td>
</tr>
<tr>
<td>LName</td>
<td>String</td>
<td>student last name</td>
</tr>
</tbody>
</table>

Primary Key: SID

Instances

- A database as the word is normally used, tables with specific contents, is know as a database instance (of a database schema)
- There can be many instances of a single database schema

<table>
<thead>
<tr>
<th>Student Number</th>
<th>Student Name</th>
<th>Grade</th>
<th>Sex</th>
<th>Class</th>
<th>Major</th>
<th>E-Mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LILJAPAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>ENGR</td>
<td><a href="mailto:jjohns@washington.edu">jjohns@washington.edu</a></td>
</tr>
<tr>
<td>2</td>
<td>LEWIS</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>MATH</td>
<td><a href="mailto:jlewis@washington.edu">jlewis@washington.edu</a></td>
</tr>
<tr>
<td>3</td>
<td>LING</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>MATH</td>
<td><a href="mailto:jling@washington.edu">jling@washington.edu</a></td>
</tr>
<tr>
<td>4</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
<tr>
<td>5</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
<tr>
<td>6</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
<tr>
<td>7</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
<tr>
<td>8</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
<tr>
<td>9</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
<tr>
<td>10</td>
<td>MILLIKAN</td>
<td>3</td>
<td>0</td>
<td>0-2</td>
<td>PHYS</td>
<td><a href="mailto:jmillikan@washington.edu">jmillikan@washington.edu</a></td>
</tr>
</tbody>
</table>
Database design is the process of setting up a database schema. Not every design is good...there are a lot of db's out there that don't avoid redundancy. Information is redundant if it is stored in multiple places in a database.

Example: UW groups needing your home address:
- Administration - to send tuition bills
- Dean - to send notice of being on “Dean’s List”
- IMA - to send you back your locker deposit

Multiple copies of the same information can have different values in different locations. Inconsistency of information is worse than no information.

Avoiding redundancy...keep specialized tables. The Administration keeps permanent records. Others keep their own tables.

Combining Tables:
When the Dean accesses the data, combine lists.

University Database Design:
- Avoiding redundancy...keep specialized tables
- The Administration keeps permanent records

Benefits of Relational Databases:
- Controls Data Redundancy
  - Information about one entity per table
  - Relationships are stored using keys
- Database Management System (DBMS)
  - Application that restricts access to data tables
    - You can’t just open a table without being inside the management system
  - Maintains information about data types (metadata)
    - Solves the data dependency problem
- Provides query facilities
  - More next week on this
Relational Database Management Systems (RDBMS)

- Database Management Systems are designed to manage access to tables (data stores)
- DBMSs normally include:
  - Data stores
  - Storage Engine
  - Query Processor
- Commercial package, like Access, includes:
  - Data Services
  - Logic Services
  - Presentation (Interfaces) Services

  This combination allows a developer to create a complete system for an organization.

You've Got To Keep 'em Separated!

- Separation of Data, Logic and Interface (Presentation) layers

Logic vs. the Query Engine

- In the lectures ahead we'll look at operations on tables and the use of SQL (Structured Query Language) to access and manipulate data
- One thing to keep in mind:
  - Logic deals with access and manipulation of data
  - Queries (SQL) deal with access and manipulation of data
  - But they aren't exactly the same
- Queries and query construction are part of Data Services
  - But they can perform simple Logic
- When queries aren’t enough to deal with all conditions, a move is made to use the Logic layer

Why Separate?

- By looking at each layer as a component, you can add and remove components without breaking entire system
- You've already worked with presentation and some simple logic in your VB programs. Mainly you can think of the VB form as the interface for the user. The code that most of you wrote was tied to the form – not just to a separate logic component
- We've learned about tables as structures for storing data about single entities and associations (by using Key attributes)
- But, having the data and having the interface doesn’t do much good if you don’t have a middle man to handle the often complicated interaction between the two.
For Wednesday

- Continuation of Introduction to Databases
- Project 3 due
- Project 4 will be handed out and discussed