Database Basics
and
An Introduction to the
Relational Database Model

FIT 100

FIT 100 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

FIT 100 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."

~Rose Vines
GeekGirls.com

So what is structured information?

FIT 100 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

FIT 100 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

FIT 100 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

FIT 100 Main Problem Is Redundant Data

Name	Course	Room#	Instructor
Paul Stevens	FIT 100	MGH 420	Whiteaker
Holly Eggelston	FIT 100	MGH 420	Whiteaker
Stephanie Wright	LIS 540	EE1 045	Boiko
Lisa Spagnolo	INFO 480	EE1 025	Whiteaker
Pam Green	FIT 100	MGH 420	Whiteaker
Thomas Nguyen	LIS 540	EE1 045	Boiko

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

FIT 100 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

FIT 100 The Keys Are the Key to Relational DB's

- Keys show how tables are related to one another

Student

SID	Name
1	Paul Stevens
2	Holly Eggelston
3	Stephanie Wright
4	Lisa Spagnolo
5	Pam Green
6	Thomas Nguyen

Course

CourseID	Name	Room
1	FIT 100	MGH 420
2	LIS 540	EE1 045
3	INFO 480	EE1 025

Class_Event

SID	CourseID	InstID
3	2	2
4	3	1
1	1	1
2	1	1
3	3	2

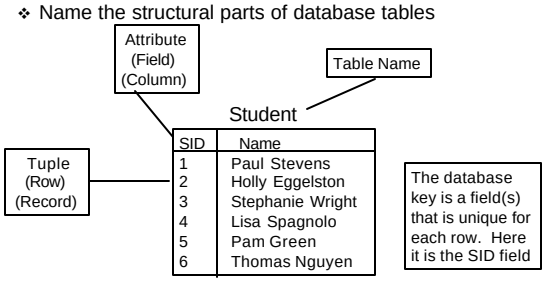
Instructor

InstID	Name
1	Whiteaker
2	Boiko

Note:
No Data Redundancies

FIT 100 Terms of a Database Table

- Name the structural parts of database tables



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

FIT 100 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.

Student

SID	Integer	*student UW ID #*
FName	String	*student first name*
LName	String	*student last name*

Primary Key: SID

FIT 100 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

Student Number	Student Name	Credits	Sex	Class	Major	E-Mail
[redacted]	LATA,POONAM	05.0	F	4	0-C-ANTH	pchant@u.washington.edu
[redacted]	LEE,BRADLEY J	05.0	M	6	0-A-N MATR	leebr@u.washington.edu
[redacted]	LIM,HENDRIK	05.0	M	3	0-C-ECON	helim@u.washington.edu
[redacted]	MALANA,WELLA JOYCE CUNANAN	05.0	F	2	0-C-PREMAJ	wjmalana@u.washington.edu
[redacted]	NG,SAI-LAI	05.0	M	3	0-C-ECON	sailai@u.washington.edu
[redacted]	PINNAMANENI,PRATHIBA	05.0	F	3	0-C-ANTH	prathi@u.washington.edu
[redacted]	SIMPSON,JILL KATHERINE	05.0	F	2	0-C-PREMAJ	jksimp@u.washington.edu

FIT 100 Again, Redundancy is Very, Very Bad

- 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

FIT 100 University Database Design

- Avoiding redundancy … keep specialized tables
- The Administration keeps permanent records

Student
- Student_ID
- First_Name
- Middle_Name
- Last_Name
- Birthdate
- Citizenship
- Primary Key: Student_ID

Personal Data

Home_Base
- Student_ID
- Street_Address
- City
- State
- Country
- PostalCode
- PK: Student_ID

Permanent Address

- Others keep their own tables

Top_Scholar
- Student_ID
- Nick_Name
- Major
- GPA
- PK: Student_ID

Dean's Data

Good_Sport
- Student_ID
- Locker_Number
- Dep_Amount
- Awards
- PK: Student_ID

IMA Data

These tables have no redundancy

FIT 100 Combining Tables

- When the Dean accesses the data, combine lists

Permanent Address

Student_ID	Nick_Name	Major	GPA
1021253	Chela	INFO	3.89
1021343	J.T.	SPAN	3.85

Dean Data

Student_ID	Street_Address	City	State	Country	PostalCode
1021253	14 Mountain Ave	Victoria	BC	Canada	V6N4T4
1014684	1774 Estates Way	Sacramento	CA	USA	92958
1015674	190 Via Mission	Chico	CA	USA	95928
1021343	1715 65th Ave	Seattle	WA	USA	98125
…	…			…	…

Dean's View of Database

Student_ID	Nick_Name	Major	GPA	Street_Address	City	State	Country	PostalCode
1021253	Chela	INFO	3.89	14 Mountain Ave	Victoria	BC	Canada	V6N4T4
1021343	J.T.	SPAN	3.85	1715 65th Ave	Seattle	WA	USA	98125
…	…	…	…	…	…	…	…	…

Concept: Arrange tables to avoid redundancy, join tables to provide the data users want to see

FIT 100 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

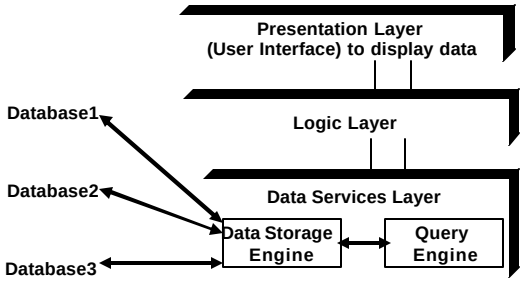
FIT 100 Relational Database Management Systems (RDBMS)

- 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.

Trivia note: Chorus line from OffSpring's "Come out and Play"

FIT 100 You've Got To Keep 'em Separated!

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



FIT 100 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.

FIT 100 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

FIT 100 For Wednesday

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