Database Intro

INFO/CSE 100, Autumn 2004
Fluency in Information Technology

http://www.cs.washington.edu/100

Why Study Databases?

- Some of us want to compute, but all of us want information …
  - Much of the archived information is in tables
  - Databases enhance applications, e.g. Web
  - Once you know how to create databases, you can use them to personal advantage
  - Databases introduce interesting ideas

Readings and References

- Reading
  - *Fluency with Information Technology*
    - Chapter 13, Introduction to Database Concepts
- References
  - *Access Database: Design and Programming*
    - by Steve Roman, published by O'Reilly

How to organize the data?

- Before relational databases (the kind we study) there were only “flat files”
  - Structural information is difficult to express
  - All processing of information is “special cased”
    - custom programs are needed
  - Information repeated; difficult to combine
  - Changes in format of one file means all programs that ever process that file must be changed
    - eg, adding ZIP codes
tab-delimited file example

| 1100322-1L3RA-X | D001 | N | N |
| 1100322-1L3RA-X | D002 | G | G |
| 1100322-1L3RA-X | D003 | G | G |
| 1100322-1L3RA-X | D004 | G | G |
| 1100322-1L3RA-X | D005 | G | G |
| 1100322-1L3RA-X | D006 | G | G |
| 1100322-1L3RA-X | D007 | G | G |
| 1100322-1L3RA-X | D008 | G | G |
| 1100322-1L3RA-X | D009 | A | G |
| 1100322-1L3RA-X | D010 | N | N |
| 1100322-1L3RA-X | D011 | N | N |
| 1100322-1L3RA-X | D012 | N | N |
| 1100322-1L3RA-X | D013 | G | G |
| 1100322-1L3RA-X | D014 | A | G |
| 1100322-1L3RA-X | D015 | N | N |
| 1100322-1L3RA-X | D016 | N | N |
| 1100322-1L3RA-X | D033 | A | G |
| 1100322-1L3RA-X | D034 | G | G |
| 1100322-1L3RA-X | D035 | G | G |
| 1100322-1L3RA-X | D036 | A | G |
| 1100322-1L3RA-X | D037 | G | G |
| 1100322-1L3RA-X | D038 | G | G |
| 1100322-1L3RA-X | D039 | G | G |
| 1100322-1L3RA-X | D040 | G | G |

Unix termcap example

```bash
# FILE FORMAT:
#
# The version you are looking at may be in any one of three formats: master
# (terminfo with OT capabilities), stock terminfo, or termcap. You can tell
# which by the format given in the header above.
#
# The master format is accepted and generated by the terminfo tools in the
# ncurses suite; it differs from stock (System V-compatible) terminfo only
# in that it admits a group of capabilities (prefixed `OT') equivalent to
# various obsolete termcap capabilities.
#
# ANSI capabilities are broken up into pieces, so that a terminal
# implementing some ANSI subset can use many of them.
ansi+local1:
    :do=\E[B:le=\E[Dind=\E[C:up=\E[A:
ansi+local1:
    :do=\E[B:le=\E[Dind=\E[C:up=\E[A:
ansi+local1:
    :do=\E[B:le=\E[Dind=\E[C:up=\E[A:
ansi+tabs:
ansi+local1:
    :it#8:tc=ansi+tabs:
ansi+local1:
    :it#8:tc=ansi+tabs:
ansi+local1:
    :it#8:tc=ansi+tabs:
```

Library example

- notice the redundancy

Relational Databases

- Information is stored in tables
  - Tables store information about **entities**
  - Entities have characteristics called **attributes**
  - Each row in a table represents a single entity
    - Each row is a set of attribute values
    - Every row must be unique, identified by a key
  - Relationships -- associations among the data values are stored

Table structure = schema
Table contents = instance
A Table in a Database

Tables have names, attributes, rows

| ID number | unique number(Key) |
| Last | person's last name |
| First | person's first name |
| JobCode | current position |
| Hire date | first day on job |

Redundancy in a database is Very Bad

- Not every assembly of tables is a good database
- Repeating data is a bad idea
  - Replicated data can differ in its different locations, e.g. multiple addresses can differ
    - Inconsistent data is worse than no data
  - Keep a single copy of any data
    - if it is needed in multiple places, associate it with a key and store key rather than the data

Relationships between tables
“You can look it up”

- When looking for information, a single item might be the answer, but a table is more likely
  - Which employees live in Kirkland?
    - Table of employees
  - Who is taking INFO/CSE 100?
    - Table of students
  - Whose mile run time \( \leq 4:00 \)?
    - Table of runners

Query to a database (set of tables) produces a new table

Relational Algebra: Tables From Tables

- There are five basic “algebraic” operations on tables:
  - Select -- pick rows from a table
  - Project -- pick columns from a table
  - Union -- combine two tables w/like columns
  - Difference -- remove one table from another
  - Product -- create “all pairs” from two tables

From this basis, many more complicated operations can be built up

Select Operation

- Select creates a table from the rows of another table meeting a criterion

\[
\text{Select from Example On Hire} < 1993
\]

Project

- Project creates a table from the columns of another table

\[
\text{Project Last, First From Example}
\]

This is a projection from 9 dimensions to 2 dimensions
Union

- Union combines two tables with same attributes
  
  All employees = perms UNION temps

 Difference

- Difference (written like subtraction) removes 1 table’s rows from another
  
  - Eastern = States - WestCoast

Product

- Product (written like multiplication) combines columns and pairs all rows
  
  Colors = Blues x Reds

  **Column Rule:** If A has $x$ columns, B has $y$ columns, A x B has $x+y$ columns
  
  **Row Rule:** If A has $m$ rows, B has $n$ rows A x B has $mn$ rows

Join

- Join (written like a bow tie) combines rows if common field matches
  
  Employee List = Perms JOIN JobCodes
**DB Operations**

- The five DB Operations can create any table from a given set of tables
  - All modern database systems are built on these relational operations
  - Join is not primitive, but can be built from 5
  - Join, select and project are used most often
  - The operations are not usually used directly, but are used indirectly from other languages

SQL, the DB language we learn, is built on basic 5