Welcome to

CMPSCI 445
Information Systems

Instructor:
Prof. Gerome Miklau
Overview of Information Management

Gerome Miklau
CMPSCI 445 – Information Systems

UMass Amherst

Sep 3, 2008

Some slide content courtesy of Zack Ives, Ramakrishnan & Gehrke, Dan Suciu, Ullman & Widom
Today

• Overview of data management
• Course topics
• Course requirements
• Student information form
Goals of this course

• Relational databases
  – an introduction to their design and use.

• Web data management
  – an introduction to key technologies for managing data on the WWW.
Databases & DBMS’s

• A **database** is a large, integrated collection of data.

• A **database management system (DBMS)** is a software package designed to store and manage databases, allowing:
  – Define the kind of data stored
  – Querying/updating interface
  – Reliable storage & recovery of 100s of GB
  – Control access to data from many concurrent users
Can filesystems do it?

Not really.

• Schema for files is limited
• No query language for data in files
• Files can store large amounts of data, but
  – no recovery from failure
  – no efficient access to items within file
  – buffering in memory
• Concurrent access not safe
Evolution

• Early DBMS’s (1960’s), evolved from file systems.
• Data with many small items & many queries or modifications:
  – Airline reservations
  – Banking
Early DB systems

Data model
The data model includes basic assumptions about what an “item” of data is, how to represent it and interpret it.

- Tree-based *hierarchical* data model
- Graph-based *network* data model

- Encouraged users to think about data the way it was stored.
- No high level query language
The Relational Model

• The relational data model (Codd, 1970):
  
  – Data independence: details of physical storage are hidden from users
  – High-level declarative query language
    • say what you want, not how to compute it.
    • mathematical foundation
DBMS Benefit #1: Generality and Declarativity

• The programmer/user does not need to know details:
  – indices, sort orders, machine speeds, disk speeds, concurrent users, etc.

• Instead, the programmer/user programs with a *logical model* in mind

• The DBMS “makes it happen” based on an understanding of relative costs of different methods
Benefit #2: Efficiency and Scale

- Efficient storage of hundreds of GBs of data
- Efficient access to data
- Rapid processing of transactions
Benefit #3: Management of Concurrency and Reliability

- Simultaneous transactions handled safely.
- Recovery of system data after system failure.

- More formally: the ACID properties
  - Atomicity - all or nothing
  - Consistency - sensible state not violated
  - Isolation - separated from effects
  - Durability - once completed, never lost
How Does One Build a Database?

• Start with a conceptual **model**
• Design & implement **schema**
• Write **applications** using DBMS and other tools
  – Many ways of doing this (DBMS, API writers, library authors, web server, etc.)
  – Common applications include PHP/JSP/servlet-driven web sites
• The DBMS takes care of query **optimization** and **execution**
Conceptual Design

STUDENT
- sid
- name

PROFESSOR
- fid
- name

COURSE
- cid
- name
- semester

Takes

Teaches
Designing a Schema (Set of Relations)

- Convert to tables + constraints
- Then need to do “physical” design: the layout on disk, indices, etc.

<table>
<thead>
<tr>
<th>STUDENT</th>
<th>Takes</th>
<th>COURSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>name</td>
<td>sid</td>
</tr>
<tr>
<td>1</td>
<td>Jill</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Bo</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Maya</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROFESSOR</th>
<th>Teaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>fid</td>
<td>name</td>
</tr>
<tr>
<td>1</td>
<td>Diao</td>
</tr>
<tr>
<td>2</td>
<td>Saul</td>
</tr>
<tr>
<td>8</td>
<td>Weems</td>
</tr>
</tbody>
</table>
Queries

• Find all courses that “Mary” takes

```
SELECT  C.name
FROM    Students S, Takes T, Courses C
WHERE   S.name=“Mary” and
        S.sid = T.sid and T.cid = C.cid
```

• What happens behind the scene?
  – Query processor figures out how to answer the query efficiently.
Queries, behind the scene

Declarative SQL query → Query execution plan:

SELECT C.name
FROM Students S, Takes T, Courses C
WHERE S.name="Mary" and S.sid = T.sid and T.cid = C.cid

The optimizer chooses the best execution plan for a query
Architecture of DBMS

SQL Commands

Plan executor
Parser
Operator eval
Optimizer

Query evaluation engine

Files and Access methods
Buffer manager
Disk space manager

Transaction Manager
Lock Manager
Concurrency Control

Recovery Manager

Index files
Catalog
Data files

DBMS
An Issue: 80% of the World’s Data is Not in a DB!

• Examples:
  – Scientific data
    (large images, complex programs that analyze the data)
  – Personal data
  – WWW and email
    (some of it is stored in something resembling a DBMS)

• Data management is expanding to tackle these problems
  – XML data enables exchange across systems
  – Integration of diverse data sets
  – Structured queries replaced by search & approximate answers.
Why study data management?

• One of the broadest, most exciting areas in CS!
• A microcosm of CS in general
  • languages, operating systems, concurrent programming, data structures, algorithms, theory, distributed systems, statistical techniques.
Course topics and Requirements
Course topics

• **Fundamentals**: relational design, query languages, SQL.

• **Database internals**: storage, indexing, query processing, query optimization, transaction management.

• **XML** and semi-structured data models.

• **Security**: access control, privacy.

• **Other topics**: Information retrieval, advanced data types, performance tuning

• **Skills**: Postgres and PHP for web development.
Prerequisites

- CMPSCI 287: Programming Language Paradigms.
- Or consent of the instructor
Grading

- Homework: 25%
- Course Project: 20%
- Midterm: 20%
- Final: 25%
- Attendance, Participation: 10%
Homework: 25%

• Several assignments throughout the course
  – Written problem sets
  – Programming exercises with SQL, XQuery
Project: 20%

• General theme: build a web application using Postgres and PHP.
• Groups of 2-3 preferred.
• Project work will include:
  – Schema design, DB implementation
  – Web site design.
  – Multiple milestones, status report.
  – In-class presentation.
Exams

• Midterm (20%)
  – in-class around the 8th week.
• Final (25%)
  – not yet determined by registrar
Attendance & Participation

• Attend every class.
• Ask questions, contribute to answers.
• Participate in in-class exercises.
Academic honesty

• All submitted work must be your own.
  – Although students are encouraged to study together, each student is expected to produce his or her own solution to the homework problems.
  – Copying or using sections of someone else’s program or assignment, even if it has been modified by you, is not acceptable.
  – The University has very clear guidelines for academic misconduct and the staff of CS 445 will be vigorous in enforcing them. Please see the UMass policy on academic honesty here: www.umass.edu/dean_students/code_conduct/acad_honest.htm
Textbook

Database Management Systems
Ramakrishnan and Gehrke

Readings posted on the website before class.
Communication

• Instructor
  – Office hours:
    • Mon 9-10am, or by appointment
    • Held in CS building, Rm 208.
  – Email: miklau at cs.umass.edu

• Check the course webpage often

• Mailing list
  – For help: cs445-help AT edlab-mail.cs.umass.edu
  – Class list: cs445_AT edlab-mail.cs.umass.edu
Information about you

• Please fill out a student information form.
Questions about the course?