CSE 544
Principles of Database Management Systems

Magdalena Balazinska (magda)
Winter 2009
Lecture 1 - Class Introduction
Outline

• Introductions

• Class overview

• What is the point of a db management system (DBMS)?

• Main DBMS features and DBMS architecture overview
Course Staff

- **Instructor: Magda** *(magda@cs.washington.edu)*
  - Office hours by appointment
  - Location: CSE 550

- **TA: Evan Welbourne** *(evan@cs.washington.edu)*
  - Graduate student in the database & ubicomp groups
  - Office hours:
    - Monday 12pm-1pm
    - Wednesday 9:30am - 10:30am
    - By appointment
  - Location: CSE 405
Who is Magda?

• Assistant Professor since January 2006
• PhD from MIT, February 2006

• Areas of interest: databases and systems
• Current research focus
  – Cloud computing
  – Scientific data management
  – RFID data management
  – Stream processing
Goals of the Class

• **Study principles of data management**
  – Data models, data independence, normalization
  – Data integrity, availability, consistency, etc.

• **Study key DBMS design issues**
  – Storage, query execution and optimization, transactions
  – Distribution, parallel processing, massive data processing
  – Data warehousing, streaming data, etc.

• **Ensure that**
  – You are comfortable using a DBMS
  – You can write applications that use a DBMS as a back-end
  – You have an idea about how to build a DBMS
  – You know a bit about current research topics in data management
Class Format

• Two lectures per week: MW @ 10:30am

• Mix of lecture and discussion
  – Mostly based on papers
    • Must read papers before lecture and submit paper review
  – Come prepared to discuss the papers assigned for the class
    • Class participation counts for a non-negligible part of your grade

• One guest lecture: David Lomet from Microsoft Research
Readings and Notes

• **Readings are based on papers**
  – Mix of old seminal papers and new papers
  – **Papers available online on class website**
  – Many come from the “red book” [optional]
  – Three types of readings
    • Mandatory, additional resources, and optional

• **Background readings from the following book**

• **Lecture notes (the ppt slides)**
  – Posted on class website after each lecture
Class Resources

• Website: lectures, assignments, projects
  http://www.cs.washington.edu/544
  List of all the deadlines

• Mailing list:
  cse544@cs.washington.edu
  Make sure you register!
Evaluation

• **Class participation 10%**
  – Paper readings and discussions

• **Paper reviews 5%: Individual**
  – Due before each lecture
  – Reading questions are posted on class website

• **Assignments 25%: Groups of two**
  – HW1: Using a DBMS (SQL, views, indexes, etc.) & writing apps
  – HW2 & HW3: Building a simple DBMS

• **Project 35%: Groups of two to four**
  – Small research or engineering. Start to think about it now!

• **Final exam 25%: During finals week**
Class Participation

• An important part of your grade

• Because
  – We would like you to read and think about papers throughout the quarter
  – Important to learn to discuss papers

• Expectations
  – Ask questions, raise issues, think critically
  – Learn to express your opinion
  – Respect other people’s opinions
Paper reviews

- Between 1/2 page and 1 page in length
  - Summary of the main points of the paper
  - Critical discussion of the paper

- Reading questions
  - For some papers, we will post reading questions to help you figure out what to focus on when reading the paper
  - Please address these questions in your reviews

- Grading: credit/no-credit
  - You can skip one review without penalty
  - MUST submit review BEFORE lecture
  - Individual assignments (but feel free to discuss paper with others)
Assignments

- **Goals:**
  - Hands-on experience using a DBMS and writing apps for DBMS
  - Hands-on experience building a simple DBMS

- **HW1:** Check website for instructions and due date
  - Setup a db from scratch
  - Practice writing SQL queries & browse the system catalog
  - Get experience with integrity constraints & triggers
  - Play with indexes and views
  - Writing an application that uses a db as a back-end

- **HW2 & HW3:** Build a simple DBMS

- We will accept late assignments with valid excuse
Project Overview

• Topic
  – Choose from a list of mini-research topics
  – Or come up with your own
  – Can be related to your ongoing research
  – Can be related to a project in another course
  – Must be related to databases
  – Must involve either research or significant engineering
  – Open ended

• Final deliverables
  – Short conference-style paper (8 pages)
  – Conference-style presentation
Project Goals

- **Apply database principles to a new problem**
  - Understand and model the problem
  - Research and understand related work
  - Propose some new approach
    - Creativity will be evaluated
  - Implement some parts
  - Evaluate your solution
  - Write-up and present your results

- **Amount of work may vary widely between groups**
Project Milestones

- Jan 19th: teams formed
- Feb 2nd: project proposal
- Feb 20th: milestone report
- March 11th: project presentations
- March 13th: final project reports

- More details on the website, including ideas & examples
- We will meet with you regularly throughout the quarter
Let’s get started

• What is a database?

• Give examples of databases
Let’s get started

- **What is a database?**
  - A collection of files storing related data

- **Give examples of databases**
  - Accounts database; payroll database; UW’s students database; Amazon’s products database; airline reservation database
Data Management

• Data is valuable but hard and costly to manage

• Example: Store database
  – **Entities**: employees, positions (ceo, manager, cashier), stores, products, sells, customers.
  – **Relationships**: employee positions, staff of each store, inventory of each store.

• What operations do we want to perform on this data?
• What functionality do we need to manage this data?
Required Functionality

1. Describe real-world entities in terms of stored data
2. Create & persistently store large datasets
3. Efficiently query & update
   1. Must handle complex questions about data
   2. Must handle sophisticated updates
   3. Performance matters
4. Change structure (e.g., add attributes)
5. Concurrency control: enable simultaneous updates
6. Crash recovery
7. Access control, security, integrity

Difficult and costly to implement all these features
Database Management System

• A DBMS is a software system designed to provide data management services

• Examples of DBMS
  – Oracle, DB2 (IBM), SQL Server (Microsoft),
  – PostgreSQL, MySQL,…
Market Shares

• In 2004 (from www.computerworld.com)
  – IBM, 35% market with $2.5 billion in sales
  – Oracle, 33% market with $2.3 billion in sales
  – Microsoft, 19% market with $1.3 billion in sales
Typical System Architecture

“Two tier system” or “client-server”

Data files

Database server (someone else’s C program)

connection (ODBC, JDBC)

Applications
Main DBMS Features

- Data independence
  - Data model
  - Data definition language
  - Data manipulation language
- Efficient data access
- Data integrity and security
- Data administration
- Concurrency control
- Crash recovery
- Reduced application development time

How to decide what features should go into the DBMS?
A Quick Look Inside a DBMS

- Process Manager
  - Admission Control
  - Connection Mgr

- Query Processor
  - Parser
  - Query Rewrite
  - Optimizer
  - Executor

- Storage Manager
  - Access Methods
  - Lock Manager

- Shared Utilities
  - Memory Mgr
  - Disk Space Mgr
  - Replication Services
  - Admin Utilities

When not to use a DBMS?

• DBMS is optimized for a certain workload

• Some applications may need
  – A completely different data model
  – Completely different operations
  – A few time-critical operations

• Examples
  – Text processing
  – Scientific analysis
Levels of abstraction in a DBMS

- **External Schema**
  - views
  - access control

- **Conceptual Schema**
  - a.k.a logical schema describes stored data in terms of data model

- **Physical Schema**
  - includes storage details
  - file organization
  - indexes

**Disk**