INFS 614
Database Management
Fall 2013

Lecture One - Introduction

Instructor: Dr. Jessica Lin
Basics

**Instructor:** Dr. Jessica Lin

**Contact Info:**

- Email: [jessica@cs.gmu.edu](mailto:jessica@cs.gmu.edu)
- Homepage: [http://www.cs.gmu.edu/~jessica](http://www.cs.gmu.edu/~jessica)
- Office: Engineering Building Room 4419
- Phone: (703)993-4693
- Office Hours: TR 3-4pm

**Class Meeting:** **Tuesday 4:30-7:10pm Art and Design Building 2026**

**Pre-requisites:** SWE 510, INFS 501, 515, 519
Outline

• Course syllabus
• Satisfactory of prerequisites
  – Signed form must be submitted TODAY
• Introduction to DB & DBMS
• Introduction to ER (Entity-Relationship) model
Administration Trivia

• In most cases, I will put the slides online the night before the lecture.
• I recommend that you print out the slides before attending lecture.
• You are 100% responsible for any announcements and updates on the class webpage, so visit the page frequently.
Electronic Device Policy

• No electronic devices allowed.
Textbook

• Required Book:

• Recommended Book:
  – Oracle 10g Programming: a Primer, by Rajshkhar Sunderraman, Addison Wesley, ISBN 0321463048
Database Management Systems

Third Edition

Ramakrishnan · Gehrke

NEW material on Database Applications

3rd Edition

www.cs.wisc.edu/~dbbook
Grading

- Midterm Exam: 25%
- Final Exam: 35%
- Project: 25%
- Quiz: 12%
- Class Participation: 3%

- Homework (practice questions) will be assigned, but they will not be graded.

- There will be 4 quizzes in the class.

- All exams are closed-book, closed-notes. The final exam is comprehensive.
Honor Code System

• GMU honor Code
  http://academicintegrity.gmu.edu/honorcode/

• In addition, the CS Department has specific honor code policies for programming projects, etc.:

• For this class
  – You may work in a team of 2 for the project.
  – Exams: individual effort, closed books/notes
Tools we will use for the class

• Oracle
  – More on this later

• Piazza
  – A free online class Q&A platform. You should have received an invitation to sign up.
  – Think before posting the question (e.g. is the answer in the book or the lecture slides?). You are encouraged to answer each other’s questions.

• Blackboard
Project Submission

• [https://mymasonportal.gmu.edu/webapps/portal/frameset.jsp](https://mymasonportal.gmu.edu/webapps/portal/frameset.jsp)

• Login with your GMU student account

• Click on the Courses tab on the upper right hand corner

• Choose INFS 614

• Use Blackboard for:
  – Electronic submission of projects
  – Checking grades
  – Getting course materials such as homework solutions
Contact Policy

- Contact the Oracle support group ([oracle@ite.gmu.edu](mailto:oracle@ite.gmu.edu)) if you
  - Encounter technical difficulties with Oracle

- Contact the TA (or post on Piazza) if you
  - Have questions about the course materials
  - Have questions about homework or project

- Contact me if you
  - Have questions about the exams
  - Have general questions/concerns about the course
Email Policy

• I strongly prefer that you only email me from your official GMU email. If you must email me from another account, you must state your full name and your official GMU email address.
Class Schedule

• See class website for the tentative class schedule:
Useful links for your computing needs

- [http://labs.ite.gmu.edu/](http://labs.ite.gmu.edu/) for
  - Mason account information (FAQ -> “I need an account or password…”)  
  - ITE computing lab and Oracle DBMS information (Go to FAQ->Oracle).
Satisfaction of prerequisites

• Prerequisites
  – INFS501 (Discrete mathematics)
  – INFS515 (Computer architecture/organization)
  – INFS519 (Program Design and Data Structures)
  – SWE510 (Object-oriented programming in Java)

• For IS/SWE/ISA students
  – Satisfaction = foundation requirements are all satisfied. Check letter of admission for foundation requirements.
  – Student signs the form.

• For other students
  – Satisfaction = talk to instructor and instructor agrees.
  – Instructor signs the form.

• The SoP form must be signed in order to receive a grade for the course.
Any Questions?
Introduction to Databases
What are Database and DBMS?

• Database:
  – A very large, integrated collection of data.
  – Models real-world enterprise.
    • Entities (e.g., students, courses)
    • Relationships (e.g., Frodo taking INFS 614)

• A Database Management System (DBMS) is a software package designed to store, provide access and manage databases.
Examples of DBMS Usage

• Airlines: reservations and schedules (expedia.com)
• Universities: student info, grades
• Banking: customer info and accounts (bankofamerica.com)
• Credit Cards: customer info, transactions
• Sales: customer info, inventory (Amazon.com)
• Government: taxes, census
Why do we need database management systems?

• A Database Management System (DBMS) is a tool that allows us to store, modify, and query data.

However, I can store, modify and query data in a text file!

What can a DBMS do that I can’t do with my text file solution?

A simple solution to manage data:- stick them all in a text file!
Enforcing Constraints

• With the text file solution, there is no way to enforce integrity constraints on the data. In other words people can put bad data into the text file.

• In contrast, a DBMS allows us to enforce all kinds of constraints. This really helps (but does not guarantee) that our data is correct.

A typo gives Roberta Wickham a GPA of 44.00
Scalability

• The text file solution might work for small datasets. What happens when we have big datasets?

• Most real world datasets are so large that we can only have a small fraction of them in main memory at any time, the rest has to stay on disk.
Query Expressiveness

• The text file solution would allow me to search for keywords or certain numbers (slowly).

• With a DBMS I can search with much more expressive queries. For example I can ask.. “Find all students whose GPA is greater than 2.5, and who don’t own a phone” or “what is the average GPA of the students”
Query Expressiveness II

• I could write some program that might allow more expressive queries on my text file, but it would be tied into the structure of my data and the operating system etc..

• With a DBMS we are completely isolated from the physical structure of our data. If we change the structure of our data (by adding a field, for example) or moving from a PC to a Mac, nothing changes at the front end!
Different Views

• The text file solution only allows one view of the data.

• With a DBMS I can arrange for different people to have different views of the data. For example, I can see everything, a student can see only his/her data, the TA can see data for students in his/her section, etc.
Concurrency

• Suppose I leave my text file on UNIX account, and I log in and begin to modify it at the same time my TA is modifying it!

• A DBMS will automatically make sure that this kind of thing cannot happen.
Security

• Suppose I leave my text file on UNIX account, and a student hacks in and changes their grades…

• A DBMS will allow multiple levels of security.
Crash Recovery

• Suppose I am editing my text file and the system crashes!

• A DBMS is able to guarantee 100% recovery from system crashes (to a consistent state).
Data Independence

- Applications are insulated from how data is structured and stored.
- *Logical data independence*: Protection from changes in *logical* structure of data.
- *Physical data independence*: Protection from changes in *physical* structure of data.

*One of the most important benefits of using a DBMS!*
Purposes of DBMS

• Provide support for “easy-to-use” data
  – Data model (data)
  – Transaction model (operation)

• Provide efficient storage and access of the data in terms of the data model and transactional model.
To sum up: Why Use a DBMS?

*Easier and More Efficient*

- Data independence and efficient access.
- Query expressiveness
- Reduced application development time.
- Data integrity and security.
- Concurrent access, recovery from crashes.

- Any reasons to NOT use a DBMS?
Why Learn about DBMS?

- Many decisions about how to use a DBMS for an application depend on the capabilities of the DBMS.
- To use it well, it’s necessary to also understand how a DBMS works.
Database Users

- End users (or DB application users)
- DB application programmers (more precisely, they are DBMS users)
  - E.g. webmasters
- Database administrator (DBA)
  - Designs logical /physical schemas
  - Handles security and authorization
  - Data availability, crash recovery
  - Database tuning as needs evolve

*Must understand how a DBMS works!*
Overview of Database Design

- **Conceptual design**
  - Use *ER Model*: E- *Entities* and R-*Relationships*
  - Decide the *entities* and *relationships* in the enterprise.
  - Decide what information about these entities and relationships should we store in the database.
  - Decide the *integrity constraints* or *business rules*.

- **Implementation (logical design)**
  - Map an ER model into a relational schema.
Example: University Database

Students(sid:string, name:string, login:string, age:integer, gpa:real)
Faculty(fid:string, fname:string, sal:real)
Courses(cid:string, cname:string, credits:integer)
Enrolled(sid:string, cid:string, grade:string)
Teaches(fid:string, cid:string)
Levels of Abstraction

- Many *views*, single *conceptual (logical) schema* and *physical schema*.
  - Views describe how users see the data.
    - e.g. `CourseInfo(cid:string, fname:string, enrollment:integer)`
  - Conceptual schema defines logical structure
    - e.g. `Courses(cid:string, cname:string, credits:integer)`
  - Physical schema describes the files and indexes used.

![Diagram showing the relationship between views, conceptual schema, and physical schema](image-url)
Summary

- DBMS used to maintain, query large datasets.
- Benefits include recovery from system crashes, concurrent access, quick application development, data integrity and security.
- Levels of abstraction give data independence.
- We will learn how to
  - *Design* and set up a database
    - *Design* (ER and Relational Models), and *refine* (Relational Normalization Theory)
  - *Query* the database
    - Relational Algebra and SQL
  - *Implement* database applications