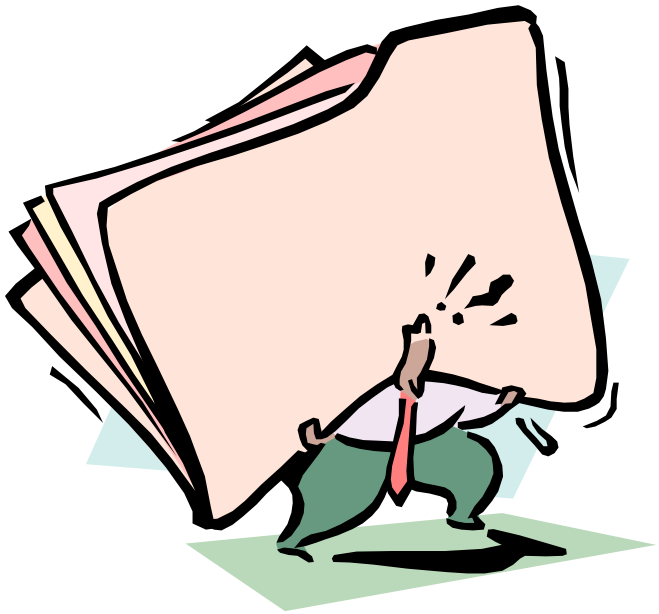
CPSC 304
Database Systems



- Instructor: Laks V.S. Lakshmanan
- Textbook: *Database Management Systems* by Raghu Ramakrishnan & Johannes Gehrke 3rd Edition.

Most (*not* all) of my slides are based on the above textbook.

Your TAs

- Jianing Yu.
- Kailang Jiang.
- Sampoorna Biswas
- Yan Zhao
- Nayantara Duttachoudhuri

Go to course home page for your TAs' contact info. and for a whole bunch of other practical info.

- Fastest way to get info./help about CPSC 304 → our online discussion forum **Piazza.**
  - **Signup:** https://*piazza.com/ubc.ca/winterterm22014/cpsc304*
  - **Piazza Class Home:** *piazza.com/ubc.ca/winterterm22014/cpsc304/home*
  - ***Course Home Page:*** *http://www.cs.ubc.ca/~laks/cpsc304/304home.html*.

A few Administrative Details

- **Online Discussion of Course Material:**
- We will use the Piazza system (www.piazza.com) for all online discussion of course material. Piazza is a *next generation Question & Answer system* specifically designed to help you get answers to your questions fast. The best way to get an answer to a question about 304 is to post it on Piazza. Piazza allows both instructor, TAs and students to answer questions, and makes it easy to edit both questions and answers to improve them. To join the Piazza group for 304, please go to piazza.com/ubc.ca/winterterm22014/cpsc304 and follow the instructions. Be sure to indicate your full name, student ID, and the email address that you want to be registered on Piazza (for this course) with.

More Admin. Details

- Please note that Piazza is a service that is hosted in the United States. (It is a startup that originated at Stanford University.) If you wish to preserve your anonymity on Piazza, so that none of your personal information is stored in the United States, you can create a new anonymous email account and request **Kailang Jiang** (jiangkl@cs.ubc.ca) to use that email account for registering you. (If you already use a gmail, hotmail or other US hosted email account, this is essentially moot.)
- We'll be using Piazza also for posting announcements and additional course related material.
- Register your clickers ASAP! (See course home page for tips.)

Unit 1

Introduction

Read → Text: Chapter 1

Our focus

- tell you what the course is about
- tell you what to expect out of the course

Learning Goals

- Explain what a Database is
- Explain what a *database management system* (DBMS) is
- Explain benefits that result from using a DBMS
- Describe the basic structure of a DBMS

Why database systems? 1/2

- One of the most successful industries.
- What powers your ATMs, or shopping portals, or travel planning/reservations sites, …?
- How reliant we have come to be: Royal Bank's infamous "software glitch" in June 2004!
- Social Networking & Recommender Systems: DBMS – Underlying core powering facebook, myspace, flickr, del.icio.us, Yahoo!Answers, rottentomatoes.com, ….
- Data management encompasses all major applications of CS.
- data management will remain important for ever:
  - Continued improvement/extensions of relational technology.
  - Developing technologies for managing data not managed (well): e.g., text, multimedia, web data, graphs, matrices, …

Why database systems? 2/2

Suppose we are building a system to store the information pertaining to the university using what we know, say Java.

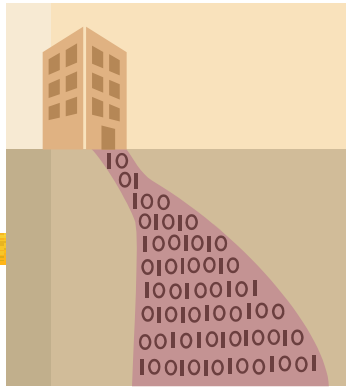
What do we need to do?

- Store data on files (how we'll organize them?)
- Write programs to access and update the data (a lot of them)
- Make sure that updates don't mess things up.
- Provide different views and access on the data to different users (registrar versus students)
- Need to write programs to deal with crashes.
- And do all the above from scratch for each application!

Alternatively:

- Use a database to store the data and a DBMS to maintain it.

What is a database?



- A **database** is an ***organized*** collection of ***related*** data, usually (but not necessarily) stored on disk. It is typically:
  - important, online
  - shared
  - secured
  - well-designed
  - variable size
- A DB typically models some real-world ***enterprise***
  - Entities (e.g., students, courses, movies, songs, cameras, …)
  - Relationships (e.g., *Hopkins* **stars in** *Fracture*;
  - *Mary* **takes** *CPSC 304; Jack* **rented** *a Civic* from *Hertz.).*

What is a DBMS?

- A ***Database Management System (DBMS)*** is a bunch of software designed to store and manage databases. It is used to:
  - define, modify, and query a database
  - provide access control
  - allow concurrent access
  - maintain integrity
  - provide backup and recovery from crashes
  - Provides a uniform data administration/maintenance
    - provides centralized control and easy data management
    - reduces application development time/effort.
  - etc.
- OK, We have to do much less now, but…

How do we design such large systems?

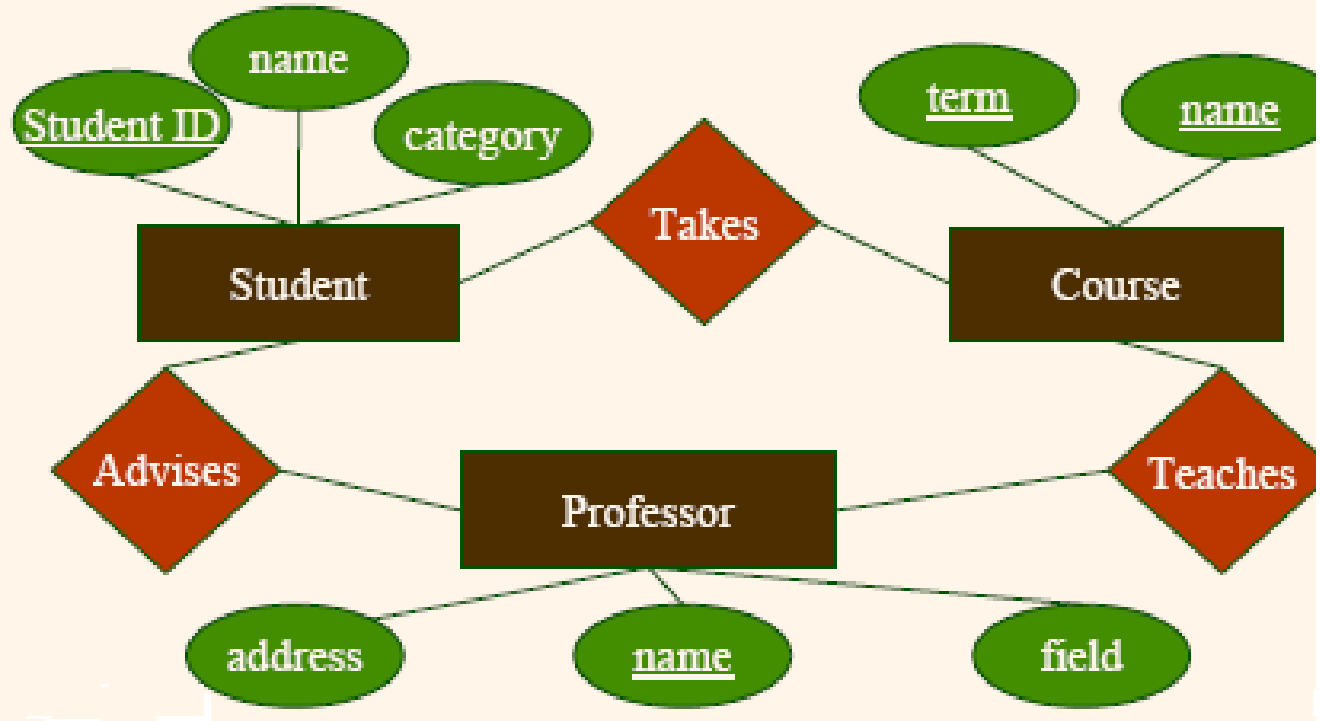
We do it in stages and we use different models to represent the info at each stage:

1. First we model the real world concepts using some high level models, called **conceptual models**
2. Then we translate the world model into a database model (called **logical data model**) that the database system understands
3. We make this data model as efficient as possible (optimization stage)
4. Design the code to *query* and *maintain* the data
5. Write the code for the application we want to develop.

In this course we'll look at all of these ….

Conceptual Data Models

- Also called **semantic data models**
- High level (more abstract) models used for capturing the real world at the initial stage.
- We'll use the most popular: Entity-Relationship model (ER)



Logical Data Models

- A major purpose of a DB system -- provide an abstract view of the data using a data model
- Data model : a collection of tools for describing
  - data , data relationships, semantics, constraints

- We'll use the **relational model** : most widely used model today.
  - Main concept: relation, basically a table with rows and columns.
  - Relations represent everything

- Other models:
  - object-oriented model
  - semi-structured data models
  - older models: network model and hierarchical model

An Example of a Relational Database

student

sid	name	address	major	gpa
92001200	Smith G.	1234 W. 12th Ave, Van.	CPSC	85
93001250	Chan J.	2556 Fraser St., Van.	MATH	80
94001150	Campeau J.	*null*	*null*	*null*
…	…	…	…	…

course

did	**num**	**title**	**credits**
CPSC	124	Principles of CS I	3
CPSC	126	Principles of CS II	3
MATH	100	Calculus I	3
…	…	…	…

etc.

Optimizing the Database

- We'll learn how to find a design that
  - captures all the information we need to store
  - takes up less space
  - is easy to maintain
- We'll look at different "Normal Forms"
  - (don't worry about it for now!)

Designing queries

- We'll mainly use Structured Query Language (SQL):

  Example: **Find all the CPSC students who have at least 75% gpa:**

  Student(sid, name, address, major, gpa)

```
select   name
from     Student
where    major="CPSC" and
         gpa >=75
```

- A declarative language – the query processor figures out how to answer the query efficiently
- We'll also look at *Relational Algebra* and a high level and intuitive logical language called *Datalog:*

Datalog Sampler

- Redo the previous query using Datalog:
- Find all the CPSC students who have at least 75% gpa:

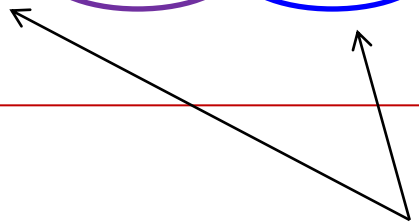
- answer(Name) ← student(Sid, Name, Addr, Major, Gpa), Major = "CPSC", Gpa >= 75.

Student(sid, name, address, major, gpa)

Datalog Sampler

- Redo the previous query using Datalog:
- Find all the CPSC students who have at least 75% gpa:

- answer(Name) ← student(Sid, Name, Addr, Major, Gpa), Major = “CPSC”, Gpa >= 75.



“Don’t Care” Variables

Student(sid, name, address, major, gpa)

Datalog Sampler

- Redo the previous query using Datalog:
- Find all the CPSC students who have at least 75% gpa:

- answer(Name) ← student(_, Name, _, “CPSC”, Gpa),
  Gpa >= 75.

Note: Datalog is essentially a pattern-driven query language.

Student(sid, name, address, major, gpa)

Create an Application

- In the term project you will use
  - a programming language to write code for an application
  - and a standard interface to the database to access the data for the application
- We'll focus on Java & JDBC
  - you can also use C++ and ODBC
  - or HTML and PHP

In Addition:

- We'll discuss Data Analytics -- Data Warehousing & OLAP as well as Data Mining.

Course Learning Outcomes

At the end of the course you will be able to :

- describe how relational databases store and retrieve information
- develop a database that satisfies the needs of a small enterprise using the principles of relational database design.
- express data queries using formal database languages like relational algebra and datalog.
- express data queries using SQL
- develop a complete data-centric application with transactions and user interface using Java, JDBC (or equivalent) and a popular DBMS
- explain key concepts and techniques used in:
  - Data Warehousing & OLAP.
  - Data Mining.

Course Activities

- You'll learn the most by *doing*:
  - assigned reading before each lecture
  - in-class clicker exercises (lectures)
  - in-class group exercises (lectures).
  - homework assignments started in tutorials
  - project – in groups of size 4 will go through the design and implementation of a realistic database application
  - additional exercises posted on the Piazza from time to time (for your own practice)

Examinations and Grading

- Midterms (70 minutes each) :
  - **Tuesday, February 3,** in class.
  - **Thursday, March 5**, in class.
- Final examination:
  - to be scheduled by the university in April
- Course grades will be calculated as follows:
  - 10% for the tutorial work (homework assignments)
  - 3% for the clicker questions
  - 2% for the (in-class) group exercises
  - 20% for the project
  - 20% for midterm exams, and
  - 45% for the final exam.
  - Plus 0.5 point for just completing midterm survey.

Passing the course

- To pass the course, you must obtain an overall passing mark and pass the project and the final.

Course Resources

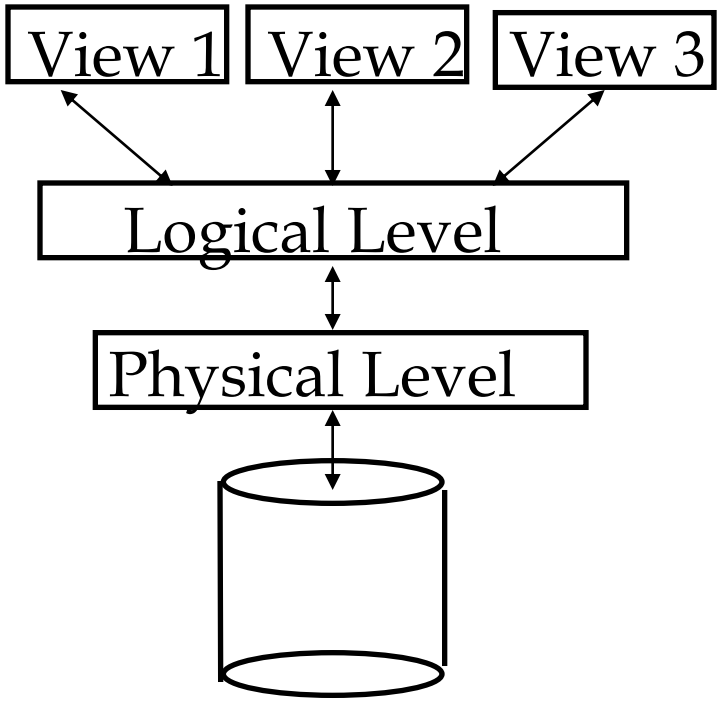
- CPSC 304 home page: http://www.cs.ubc.ca/~laks/cpsc304/304home.html
  - all course materials – notes.
  - Online discussion forum: Piazza.
    - Announcements, additional material, and grades
    - tutorials, project, exercises, etc.
- Office Hours:
  - Laks: Face2Face hour: Tues 4:30-5:30; “Piazza hour”: Thu 4:30-5:30 pm.
  - See course home page for details on TAs and tutorials/labs.
- Keep visiting course pages and piazza very often.

Back to Databases - DBMSs

- What else DBMSs are good for?
- Who are the main users of databases?
- What is the basic structure of a DBMS?

Levels of Abstraction

- Usually database systems allow users to access the data at three abstraction levels:
  - **Physical level**: shows how data are actually stored
  - **Logical (or conceptual) level**: shows data using the system's data model (i.e., tables, etc.)
  - **External (or View) level**: describes different part of the database to different users
    - for convenience, security, or other reasons
    - i.e. compare views of a bank database for teller, bank manager, & database administrator



Schema and Instances

- Similar to types and variables in programming languages
- **Schema** – the structure of the database
  - **Physical schema**: database structure at the physical level
  - **Logical schema** (or **Conceptual schema** or just **Schema**): database structure at the logical level
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
- **Logical Data Independence** – Provided by the views
  - Ability to change the logical schema without changing the applications

Relational Example: University Database

- Logical schema:
  - ***student(sid**: integer, **name**: string, **address**: string, **major**: string, **gpa**:float)*
  - ***course(dept**: string, **num**:string, **credits**:integer)*
  - ***enrolled(sid**:integer, **dept**: string, **num**:string, **grade**:integer)*
- Physical schema will define:
  - The type of file that stores each relation
  - The type of records for each file
  - Any Indexes on the files, etc.
- External Schema (View):
  - ***course_info(dept**: string, **num**:string, **enrollment**:integer)*

An Instance of the University Database

student

sid	name	address	major	gpa
92001200	Smith G.	1234 W. 12th Ave, Van.	CPSC	85
93001250	Chan J.	2556 Fraser St., Van.	MATH	80
94001150	Campeau J.	*null*	*null*	*null*
…	…	…	…	…

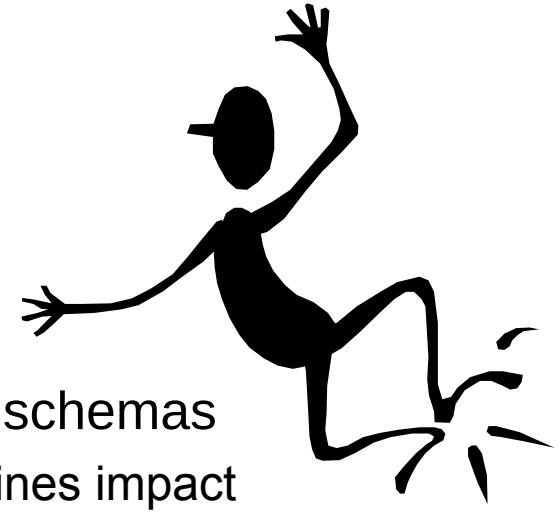
course

did	**num**	**title**	**credits**
CPSC	124	Principles of CS I	3
CPSC	126	Principles of CS II	3
MATH	100	Calculus I	3
…	…	…	…

etc.

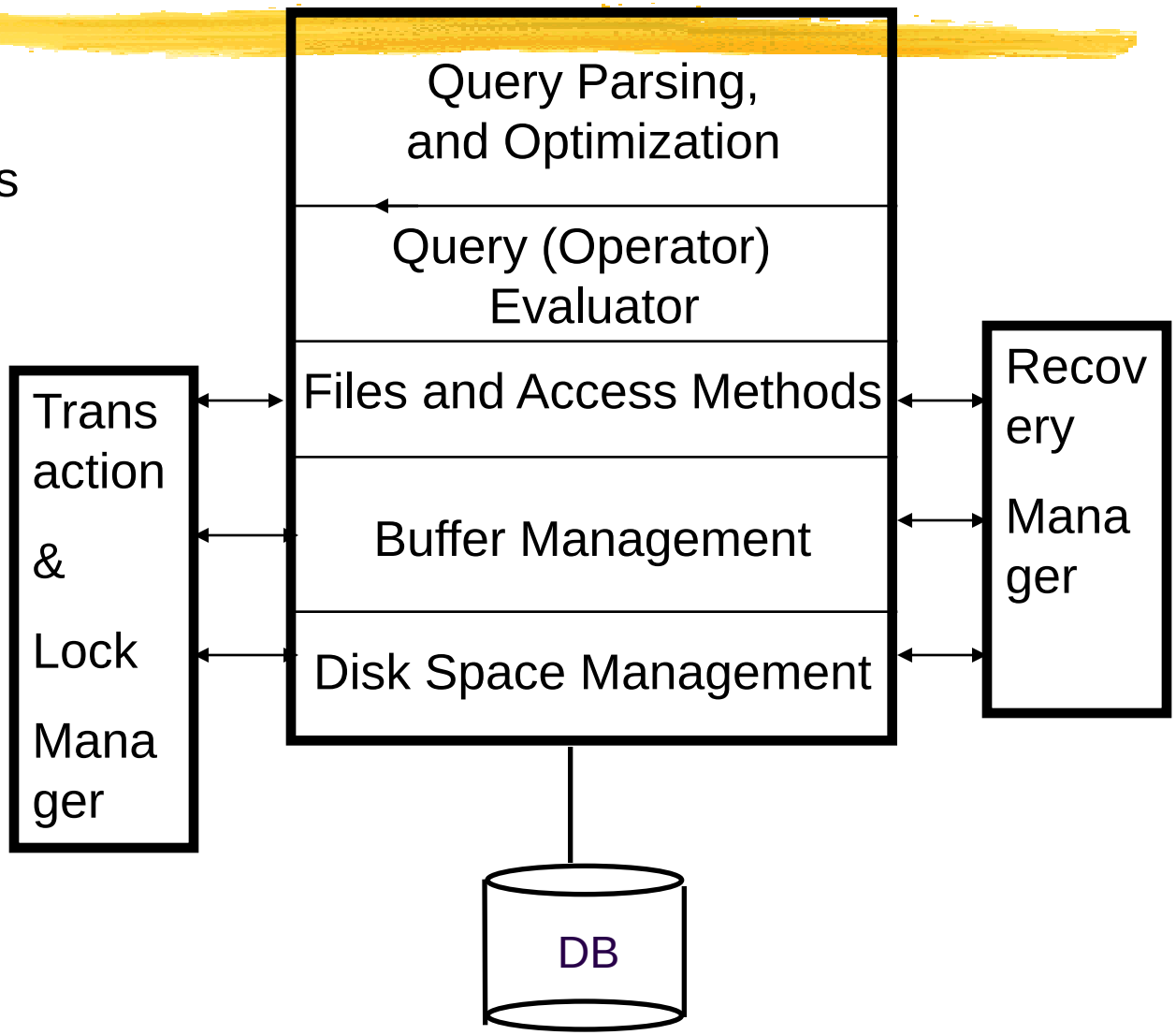
Database Primary Users

- End users (including senior mgmt)
- DB application programmers
- Webmasters
- DB Administrator
  - maintains conceptual, physical and external schemas
    - modifies the design, as req'ts change; determines impact
  - handles security and authorization
  - handles backups and recoveries
  - checks database performance and performs necessary tuning
    - indexes, reorganization, query analysis



Structure of a DBMS

- A typical DBMS has a layered architecture.
- This is one of several possible architectures; each system has its own variations.



Summary

- DBMS provide many benefits in maintaining & querying large datasets including
  - abstract representations of the data at different levels,
  - recovery from system crashes,
  - concurrent access, data integrity, and security
  - quick application development.
- Main goal of 304 is to:
  - give you the knowledge of how a DB application works
  - train you as a database programmer
  - provide you with the early steps to become a DBA
- 404, on the other hand:
  - looks deeper into a DBMS
  - trains you more as a DBA