CS411
Database Systems

01: Introduction

Kevin C. Chang

Welcome to CS411

- Web site:
  https://wiki.engr.illinois.edu/display/cs411fa13/
- Announcements, syllabus, policies, schedule, …

- Please read the class syllabus, policies, and lecture schedule; ask now if you have questions.

What is this?

Teaching Staff: The Front-End



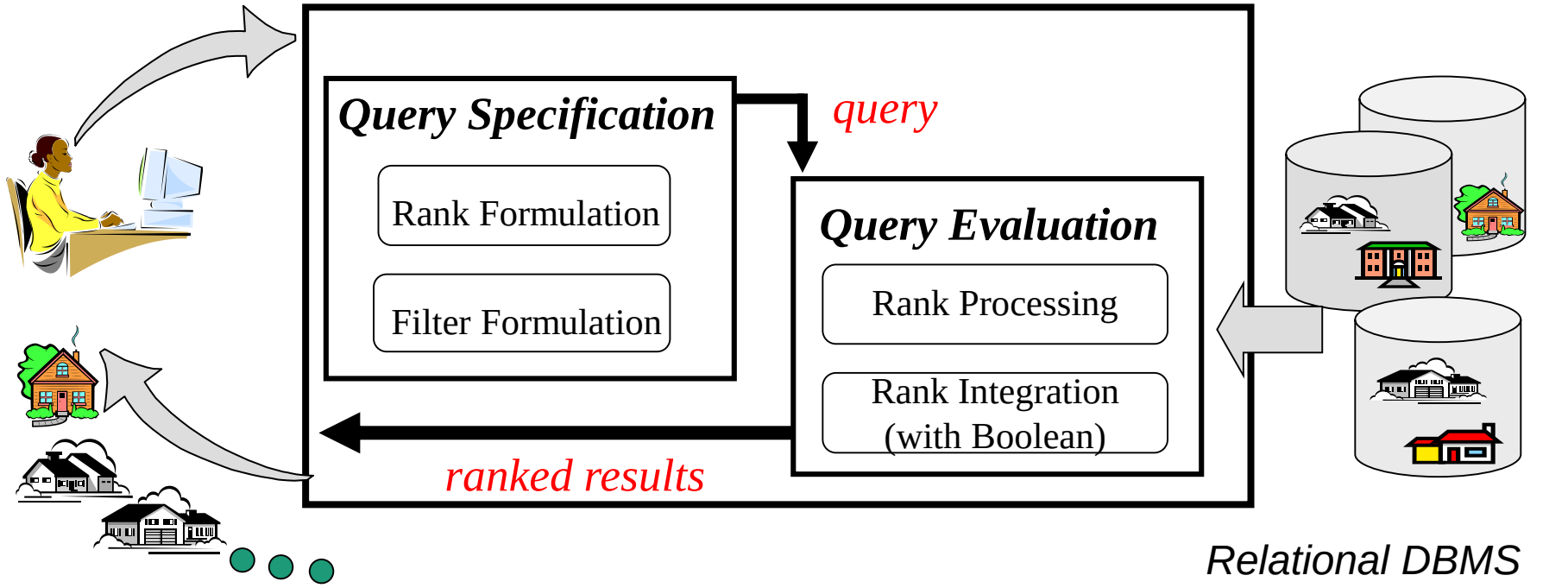

Miawei, Tanvi, Mangesh, Phuong

Teaching Staff: The Back-End

- Kevin C. Chang
- Research interest:
  - Database systems, Web integration and mining
  - Resaerch projects:
    - AIM, MetaQuerier, WISDM, ARISE, and we are recruiting!
- Hobbies:
  - Ocean diving, mountain climbing.
- Brief history
  - Taiwan (BS in EE from National Taiwan University)
  - California (MS in CS, PhD in EE from Stanford)
  - Illinois (associate professor in CS, UIUC)
  - “Data mining”: what can you predict? East or west? CS or EE?

The AIM Project: AIMing to the top

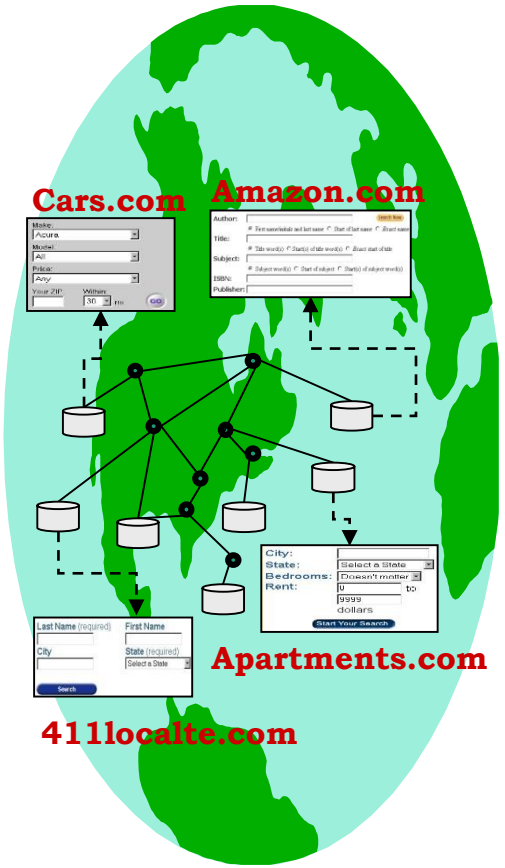
Supporting ranking in data retrieval



Relational DBMS or middleware sources (e.g,, Web)

The MetaQuerier Project

Exploring and integrating deep Web

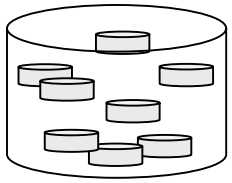


Explorer

- *source discovery*
- *source modeling*
- *source indexing*

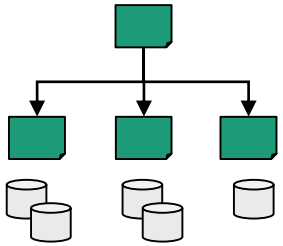
Integrator

- *source selection*
- *schema integration*
- *query mediation*



db of dbs

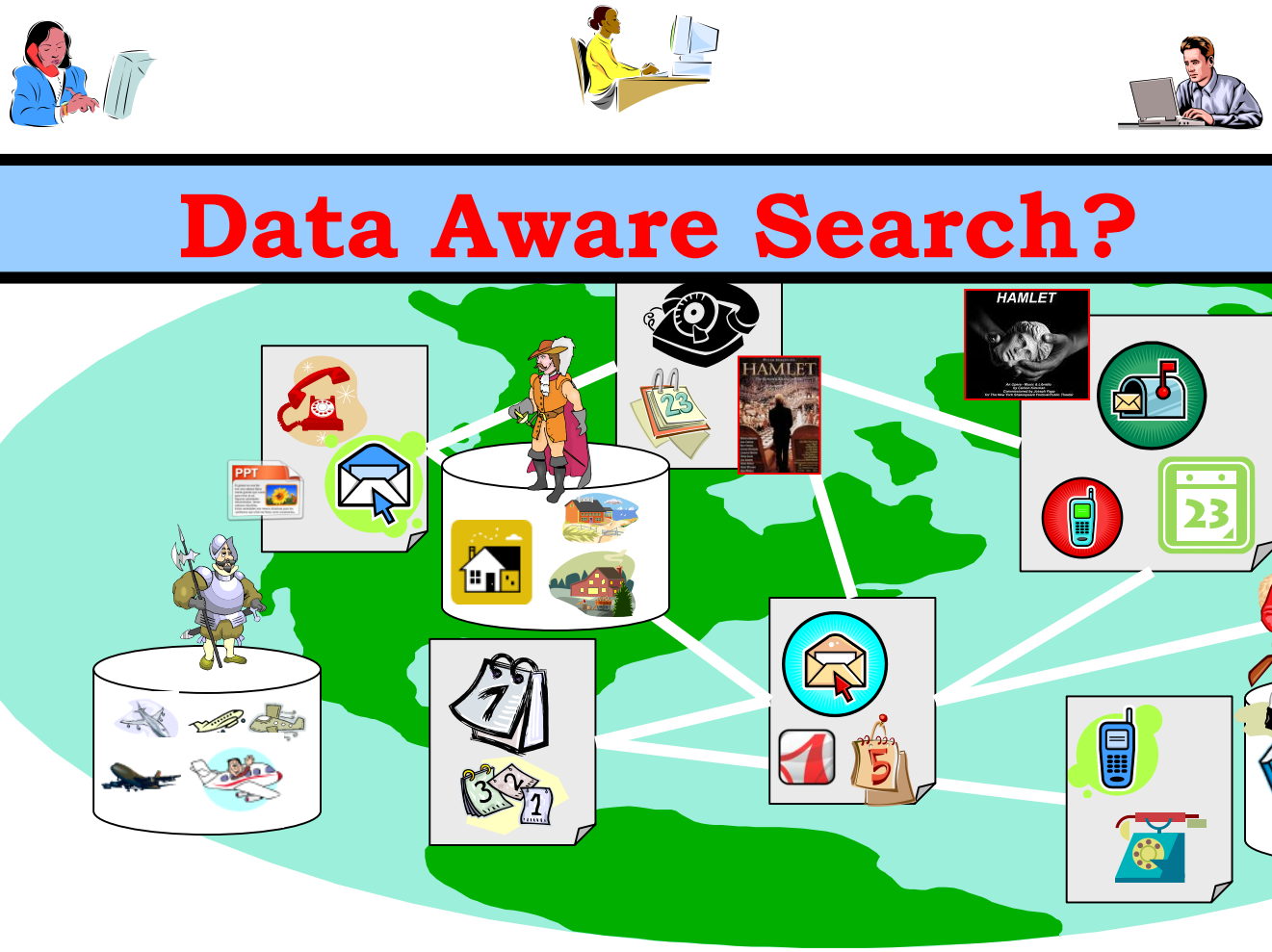
FIND *sources*



QUERY *sources*

unified query interface

The WISDM Project:
Data Aware Search over the Web



The ARISE Project:
Augmented Reality Information Search Engine

Real World

context ↓ ↑ **augmentation**

Augmented Reality Information Search Engine

Virtual World

Alex—Bob network

Bombay Grill restaurant

“global warming”

You Tell Me --

- Why are you taking this course?

- It enables many possibilities:
  - An IT Guru at Goldman-Sachs or Boeing.
  - A System Developer at Oracle or Google.
  - A CEO + receptionist + janitor (AKA startup cofounder).
  - None of the above– Please name it.

CS411 Goal: Two Perspectives of DBMS

- User perspective
  - how to use a database system?
  - conceptual data modeling, the relational and other data models, database schema design, relational algebra, and the SQL query language.
- System perspective
  - how to design and implement a database system?
  - data representation, indexing, query optimization and processing, transaction processing, concurrency control, and crash recovery

Perquisite

- Must have data structure and algorithm background
  - CS 225 or 400 equivalent
- Good programming skill
  - project will require lot of programming
  - need C++, Java, or PHP … to do a good job at talking with DB
  - you or your project group picks the language

Textbook

- Textbook:

  Database Systems: The Complete Book, 2/e, by Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer D. Widom
- Good references:
  - *Database Management Systems*, by Raghu Ramakrishnan an Johannes Gehrke, McGraw-Hill
  - *Database System Concepts*, by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan, McGraw Hill
  - *Fundamentals of Database Systems*, by Ramez Elmasri and Shamkant Navathe, Addison Wesley
  - *An Introduction to Database Systems*, by C. J. Date, Addison Wesley

Course Format

- For all students
  - two 75-min lectures / week
  - 5-6 assignments planned ← “MP Inside!”
  - project (significant)
  - a midterm and a final exam
- Graduate students: 4 credits option
  - do an extra project

Lectures

- Lecture slides will be posted shortly before or after the lecture
  - are to complement the lectures
- Lectures are important for guiding your reading of textbook
- Without your participation, a lecture is impossible
  - so please attend lectures regularly

Homework Assignments

- Mostly paper-based, some may involve machine exercise/ programming
- Due by noon of the due date
- No late homework will be accepted

Project:

The 411 Incubator, Where Ideas are Hatched!

- General Track: Database-Driven Web/Mobile-based Applications
  - select an “useful” and “innovative” application that needs a database
  - design and build it from start to finish
  - your choice of topic:
    useful, innovative, database-driven

- System Track: Take an open source DBMS (SQLite) and hack it.
  - strongly encouraged if you already have app experience

- Team work
- Significant amount of programming (we will provide tutorials)
- Will be done in stages
  - you will submit some work at the end of each stage
- Start from your PITCH and end with your DEMO.

Project Groups

- Project will be done in group of 3-4 students
  - learn how to work in a group: valuable skills
  - also use project group as study partners
- Try to form groups as soon as possible
  - can start by posting requests on the class newsgroup
- There will be a deadline soon for forming groups
  - if you have not formed groups by then
  - we will help assign you to groups
- Grading:
  - all members receive same grading
  - if someone drops out, the rest pick up the work

Exams

- Midterm and final
- There will be some brief review before each exam
- Check dates and make sure no conflict!
  - generally no makeup exams unless exceptional cases (see policy page)

In-Class Quiz

To encourage attending and participating in class:

- Three brief in class quiz.
- Each tests your participation in class.
- Each accounts for 1.5% of class grading.

Tutorials: 5-10 Sessions by Needs

- Homework Tutorial:
  - One for each homework assignment, to teach problem solving.
- Exam Tutorial:
  - One for each exam, to review and practice old exams.
- Project Tutorials:
  - Web/mobile programming, DB programming.
- Scheduled a few days before due. Likely at Tu or Th 5pm.
- Will be recorded.

Tentative Grading Breakdown

- Homework: 15%
- Project: 35%
- Midterm: 20%
- Final: 30%

How Do We Work Together?

Contacting the Staff --

Office Hours

- The best way for asking questions and clarifications
- Will have office hours every day
- See course Web for schedule

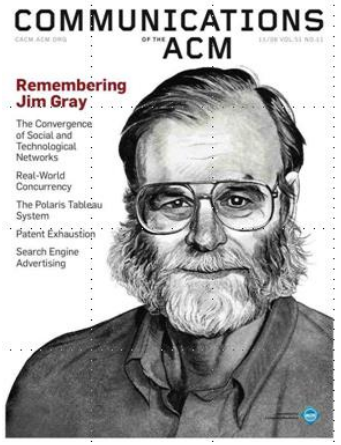
Communications

- Website: “Announcements” page
- Q/A forum: https://piazza.com/class/hj3y02os5no6zy
  - vitally important!
  - make sure to check it regularly for questions/clarifications
- If you have a question/problem
  1. talk to people in your group first
  2. post your question on newsgroup
  3. email TA
  4. go to office hours to talk to TA or instructor

Q/A Forum: Piazza

- Designed for you and your peer
  - to communicate and help one another
  - please do not post solutions/admin-requests
- TAs will monitor and try their best to help with your questions
- There can be many questions
  - may not be able to answer all of them timely manner
  - not good for more complex questions
  - hence should come to office hours or email TA

Data Management Evolution



Jim Gray: *Evolution of Data Management*. IEEE Computer 29(10): 38-46 (1996):

- Manual processing: -- 1900
- Mechanical punched-cards: 1900-1955
- Stored-program computer-- sequential record processing: 1955-1970
- Online navigational network DBs: 1965-1980
  - many applications still run today!
- Relational DB: 1980-1995
- Post-relational and the Internet: 1995-

Database Management System (DBMS)?

- System for providing EFFICIENT, CONVENIENT, and SAFE MULTI-USER storage of and access to MASSIVE amounts of PERSISTENT data

- Really?? Let's contrast with a **File System:**
  - Persistent?
  - Efficient? Convenient? Safe? Multi-user?
  - Massive?

- Homework: Contrast with main memory system.

DBMS Examples

- Most familiar use: many Web sites rely heavily on DBMS's. Examples?

- And many non-Web examples

Example: Banking system

- Data = information on accounts, customers, balances, current interest rates, transaction histories, etc.
- MASSIVE: many gigabytes at a minimum for big banks, more if keep history of all transactions, even more if keep images of checks -> Far too big for memory
- PERSISTENT: data outlives programs that operate on it

MULTI-USER Access

- MULTI-USER: many people/programs accessing same database, or even same data, simultaneously -> Need careful controls
- Alex @ ATM1: withdraw $100 from account #002
- Bob @ ATM2: withdraw $50 from account #002

- What should happen, then?

Why Direct Implementation Won't Work

- Storing data: file system is limited
  - size limit by disk or address space
  - when system crashes we may loose data
  - Password/file-based authorization insufficient
- Query/update:
  - need to write a new C++/Java program for every new query
  - need to worry about performance

- Concurrency: limited protection
  - need to worry about interfering with other users
  - need to offer different views to different users (e.g. registrar, students, professors)
- Schema change:
  - entails changing file formats
  - need to rewrite virtually all applications
- That’s why the notion of DBMS was motivated!

DBMS: More Requirements

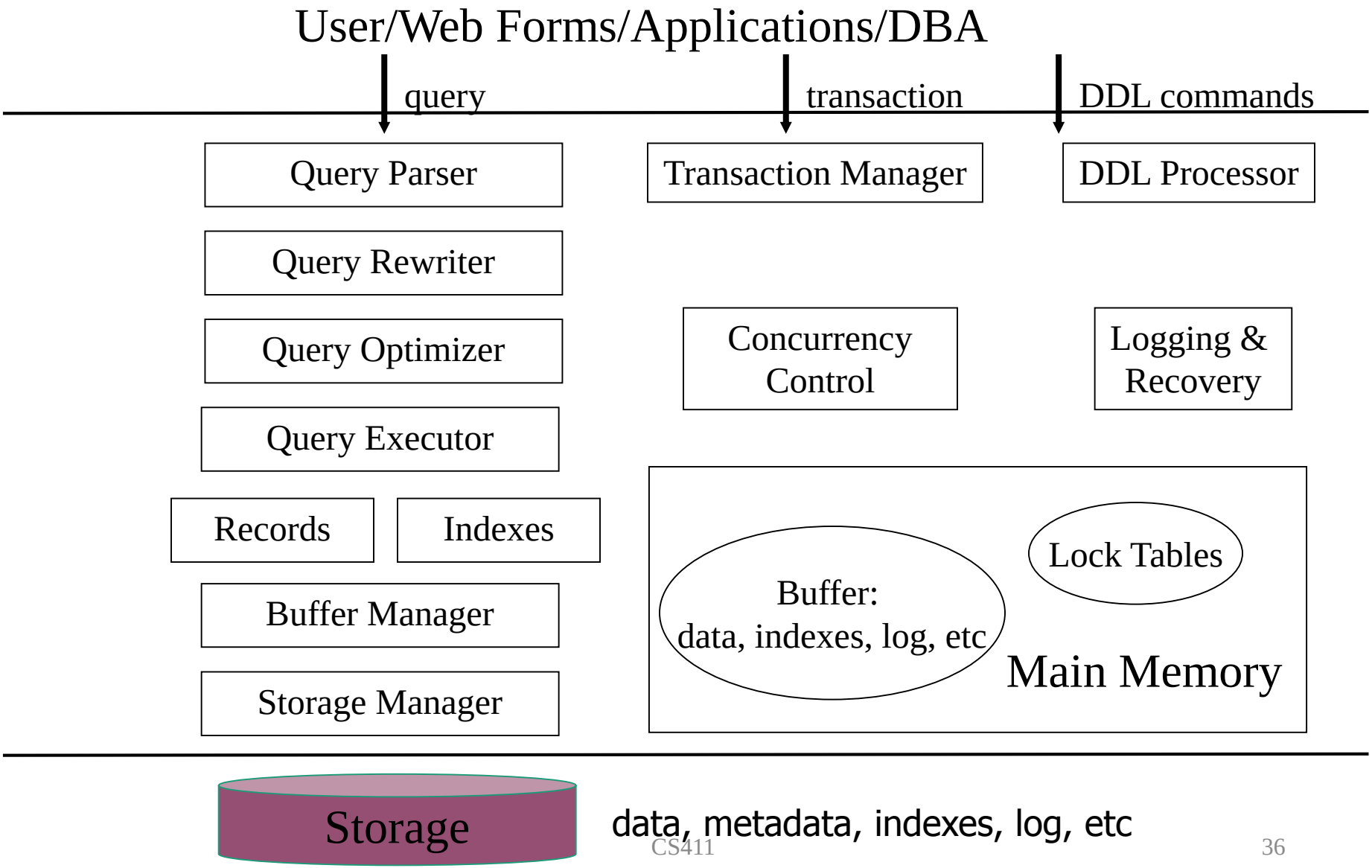
- SAFE:
  - from system failures
  - from malicious users
- CONVENIENT:
  - simple commands to - debit account, get balance, write statement, transfer funds, etc. ->
  - also unpredicted queries should be easy
- EFFICIENT:
  - don't search all files in order to - get balance of one account, get all accounts with low balances, get large transactions, etc.
  - massive data! -> DBMS's carefully tuned for performance

DBMS: A Software System

- Buy, install, set up for particular application
- Available for PC's, workstations, mainframes, supercomputers
- Major vendors:
  - Oracle
  - IBM (DB2)
  - Microsoft (SQL Server, Access)
  - Sybase

  ➢all are "relational" (or "object-relational") DBMS

DBMS Architecture



People

- DBMS user: queries/modifies data
- DBMS application designer
  - set up schema, loads data, …
- DBMS administrator
  - user management, performance tuning, …
- DBMS implementer: builds systems

Questions we must ask…

- What is data? How to view our data?
- How to “organize” data?
- How to “compute”/”query” on data?
- How to categorize (index) data?
- How to process data?
- How to acquire data?
- How to further scale up?

1/3 Topics: User Perspective

- Entity-Relationship Model
- Relational Model
- Relational Database Design
- Relational Algebra
- SQL and DBMS Functionalities:
  - SQL Programming
  - Queries and Updates
  - Indexes and Views
  - Constraints and Triggers

1/3 Topics: System Perspective

- Storage and Representation
- Indexing
- Query Execution and Optimization
- Transaction Management

1/3 Topics:
From **DB** (*relational*) to **BD** (*everything?*)

- Flexible (non-relational) data representation
  - XML/SOAP
  - JSON
  - NoSQL Databases
- Big data acquisition – Information extraction
- Big data processing – Map-Reduce

Special Topics: If You Ask!

- Databases in the Real World
- Advanced Database Research

Please nominate topics you like to hear about!

How to Get the Most out of CS411? – The classic wisdom:

- **Read and think before class**
  - welcome to ask questions before class!
- Study and discuss with your peers
  - discuss readings to enhance understanding
  - discuss assignments but write your own solution!
- Use lectures to guide your study
  - use it as a roadmap for what's important
  - lectures are **starting** points– they do not cover everything you should read

How to Get the Most out of CS411? – Knowing why you are here.

You can work at Goldman Sachs if you want.

You can work at Google if you want.

You are qualified to be a receptionist or a janitor, or a CEO, if you want.

Questions?

- Any questions? Please come talk to me.