CS 317 & CS 387
Database Information Systems

Course Logistics

- Instructor
  - Umesh Bellur, 2nd Floor, Kanwal Rekhi Building.
  - umesh.bellur@iitb.ac.in

- Teaching assistants (cs317-tas@cse.iitb)
  - Mahak Patidar
  - Namrata Jain
  - Shivani Agrawal
  - Rohan Francis
  - Manjinder Singh

Logistics (2)

- Timing and Venue
  - 317 (Theory)
    - M F – 11.35am-12.30pm
    - Th – 8.35-9.30am
    - Venue: Kanwal Rekhi Bldg Seminar hall
  - 387 (Lab)
    - Tue 2-5pm
    - Venue – old software lab (Math Bldg.)

- Course web site:
  - http://www.cse.iitb.ac.in/~cs317

Books

- Database System Concepts, Silberschatz, Korth and Sudarshan, McGraw Hill

- A First Course in Database Systems, **Second edition**, Ullman and Widom, Prentice Hall, 2002.

- Database Management Systems, Ramakrishnan and Gehrke, McGraw Hill.

- Fundamentals of Database Systems, Elmasri and Navathe, Addison-Wesley.

Why Study DBs?

- Academic:
  - Databases involve many aspects of computer science
  - Fertile area of research
  - Three Turing awards in databases

- Programmer:
  - a plethora of applications involve using and accessing databases

- Student:
  - Get those 9 credits and I don't have to ever do this again!
  - Google will hire me!

What will I learn?

- Design
  - How do you model your data and structure your information ina database?

- Programming
  - How do you use the capabilities of a DBMS?

- The course achieves a balance between
  - a firm theoretical foundation to designing moderate-sized databases and
  - creating, querying, and implementing realistic databases

What wont I learn?

- Implementation
  - How do you build a system such as ORACLE or MySQL?

Course Goals

- Create E/R models from application descriptions.

- Convert E/R models into relational designs.
- Identify redundancies in designs and remove them using normalization techniques.
- Create databases in an RDBMS and enforce data integrity constraints using SQL.
- Write sophisticated database queries using SQL.

Course Goals (2)

- Understand tradeoffs between different ways of phrasing the same query.
- Implement a web interface to a database.

Course Structuring

- **Weeks 1–3: Data Modeling**
  - Entity-Relationship (E/R) approach
  - Specifying Constraints
  - Good E/R design
- **Weeks 4–8: Relational Design**
  - The relational model
  - Converting ER to"R"
  - Normalization to avoid redundancy

Outline (2)

- **Weeks 9-14: Query/Manipulation Languages**
  - Relational Algebra
  - Relational Calculus
  - SQL
  - SQL in a programming environment
  - Recursion in Queries (SQL3)

Lab

- Start with java, j2ee
- SQL with Oracle 8i
- Using SQL with java
- Project
  - Analysis and modeling
  - DB design, interfaces, workflows …
  - Implementation using java + Oracle

Grading

- Course
  - mid-term (35%)
  - 2-4 quizzes (30%)
  - End-sem (35%)

- Lab
  - Assignments (50%)
  - Project (50%)
    - specification of problem: 5%
    - database and program design: 15%
    - code, implementation, demo: 25%
    - final report and presentation: 5%

Important note

- Attendance
- Plagiarism

What is a DBMS?

- Database Management System (DBMS) = data + set of programs to access/manipulate data

- Features of a DBMS
  - Support massive amounts of data
  - Persistent storage
  - Efficient and convenient access

- Secure, concurrent, and atomic access

RDBMS Features

- Support <u>massive</u> amounts of data
  - Giga/tera/petabytes
  - Far too big for main memory

- <u>Persistent</u> storage
  - Programs update, query, manipulate data.
  - Data continues to live long after program finishes.

RDBMS Features (2)

- <u>Efficient</u> and <u>convenient</u> access
  - Efficient: do not search entire database to answer a query.
  - Convenient: allow users to query the data as easily as possible.

- <u>Secure</u>, <u>concurrent</u>, and <u>atomic</u> access
  - Allow multiple users to access database simultaneously.
  - Allow a user access to only to authorized data.

- Provide some guarantee of reliability against system failures - ACID

ACID Properties of Transactions

- ***A****tomicity* –all or nothing execution

- ***C****onsistency* – expectations about relationships between data elements is always retained at Xaction boundaries.

- ***I****solation* – Each transaction appears to execute as if no other transaction is executing at the same time.

- ***D****urability* – the effect of a transaction, once completed is never lost

RDBMS uses

- Examples?
  - Search engines, banking systems, airline reservations, corporate records, payrolls, sales inventories.

- New applications: Wikis, biological/multimedia/scientific/geographic data, heterogeneous data.

RDBMS - History

- File systems

- The earliest databases (1960s) evolved from file systems
  - Navigational and hierarchical
  - User programmed the queries by walking from node to node in the DBMS.

DBMS Industry

- A DBMS is a software system.

- Major DBMS vendors: IBM, Microsoft, Oracle, Sybase

- Free/Open-source DBMS: MySQL, Prostgres SQL, Firebird.
  - Used by companies such as Google, Yahoo, Lycos, BASF.

- All are “relational” (or “object-relational”) DBMS.

RDBMS - Evolution

- <u>Relational DBMS (1970s to now)</u>
  - View database in terms of relations or tables
  - High-level query and definition languages such as SQL
  - Allow user to specify what she wants, not how to get what she wants

- <u>Object-oriented DBMS (1980s)</u>

- inspired by object-oriented languages
- object-relational DBMs

Example

- RDBMS = “Relational”DBMS
  - The relational model uses relations or tables to structure data

- **`ClassList`** relation:

Example

- Relation separates the logical view (externals) from the physical view (internals)

- Simple query languages (SQL) for accessing/modifying data
  - Find all students whose grades are better than B.
  - **`SELECT Student FROM ClassList WHERE Grade >“B”`**

Simplified DBMS Architecture

DBMS architecture

Query processor

- Understands SQL queries
  - Using schema and user authorization information
- Works out an optimal plan for executing the query
  - Using indices and statistical data
- Executes the query and sends results to the application program

Transaction management

- As seen earlier, a transaction is a sequence of steps that constitute a single activity. All or none should be executed

- Transaction manager ensures consistent state of data in a data base
  - When transactions are executed concurrently
    - Concurrency-control manager
  - In case of disk, database, or OS crash

Storage management

- Storage manager handles
  - Efficient storage, retrieval and updates to physical data
    - Manages files, indices, data dictionary, buffers
  - Ensures authorized access to data

- Transaction manager is actually a part of general storage management

DBIS architecture

Data base Administration

- Coordinates all activities of the database system
- Has a good understanding of the enterprise’s information resources and needs.
- Database administrator's duties include:
  - Schema definition and modification
  - Storage structure and access method definition
  - Back ups and recovery
  - User authorization
  - Performance monitoring and tuning

Application components

- Application programs typically perform following tasks
  - User interaction
  - Executing application logic
  - Accessing and updating data base

User interaction

- Display menu choices to user

- Collect data input
  - Data entry forms
  - Data validation is required
- Collect query parameters
- Display or print results
- Any special interaction (mouse, …)

Application logic

- Carries out processing of data
  - Computation of monthly pay
  - Calculating students' performance (CPI)
- The application processing typically uses
  - User provided input data
  - Data from the data base
- Data base is often updated as a result

Accessing and updating database

- Access control and user authorization has to be carried out
- Proper transaction processing is to be ensured (ACID properties)
- User queries need to be processed efficiently and results forwarded to users

Application architecture

- Ordinarily, in the conventional 3GL systems, a single program could do all of these tasks
- With the advent of computer networks, it is possible to have large systems (servers) connected to a number of smaller systems (clients) from which users access the database on server(s)

Two tier architecture

- Data bases are on server machines managed by DBMS (back end)

- User interaction portion and the application logic runs on client machines (front end)
- This is two tier or “client-server” architecture

Three tier architecture

- It is possible to factor out application logic and to run it as a separate layer (application server)
  - Either on a separate central machine or on the same machine as data base server

- Thus the back end is “split”

- A server could in turn access another
  - Distributed multi- tier architecture