BBM371- Data Management

Lecture 1: Course policies,
Introduction to DBMS

26.09.2017
Today

- **Introduction**
  - About the class
  - Organization of this course

- **Introduction to Database Management Systems (DBMS)**
About the class
Database Management Systems, Raghu Ramakrishnan, McGraw-Hill Education
Database System Implementation, Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom
The course web page will be updated regularly throughout the semester with lecture notes, announcements and important dates.

http://web.cs.hacettepe.edu.tr/~bbm371
Course Work and Grading

- **1 midterm exam (25 points)**
  - Closed book and notes

- **Pop quizzes (25 points)**
  - Closed book and notes

- **Final exam (50 points)**
<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>26-Sept</td>
<td>Course policies and outline – Introduction to DBMS</td>
</tr>
<tr>
<td>2</td>
<td>3-Oct</td>
<td>Storage Devices</td>
</tr>
<tr>
<td>3</td>
<td>10-Oct</td>
<td>Basic File Concepts</td>
</tr>
<tr>
<td>5</td>
<td>24-Oct</td>
<td>Index Types</td>
</tr>
<tr>
<td>6</td>
<td>31-Oct</td>
<td>Hash based file organization</td>
</tr>
<tr>
<td>7</td>
<td>7-Nov</td>
<td>Hash based file organization</td>
</tr>
<tr>
<td>8</td>
<td>14-Nov</td>
<td><strong>Midterm Exam</strong></td>
</tr>
<tr>
<td>9</td>
<td>21-Nov</td>
<td>Tree based indexing</td>
</tr>
<tr>
<td>10</td>
<td>28-Dec</td>
<td>Tree based indexing</td>
</tr>
<tr>
<td>11</td>
<td>5-Dec</td>
<td>External sorting</td>
</tr>
<tr>
<td>12</td>
<td>12-Dec</td>
<td>External sorting (cont.)</td>
</tr>
<tr>
<td>13</td>
<td>19-Dec</td>
<td>Spatial Indexes</td>
</tr>
<tr>
<td>14</td>
<td>26-Dec</td>
<td>Overview</td>
</tr>
</tbody>
</table>
Introduction to Database Management Systems
What is a DBMS?

- A very large, integrated collection of data.
- Models real-world enterprise
- A Database Management System (DBMS) is a software package designed to store and manage databases

Information about:
- Entities: such as students, faculty, courses
- Relationships: between entities for example a student is enrolled to a course

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke
Data-Centric Applications

- Applications in which data plays an important role
  - Airline reservation systems
    - Data: aircrafts, flights, flight attendants, passengers, etc.
  - Banking applications
    - Data: clients, deposits, withdraws, etc.
  - Hospital systems
    - Data: patients, physicians, diagnosis, prescriptions, etc.
  - University systems
    - Data: students, teaching staff, courses, enrollments, etc.

Taken from the slides of Mohamed S. Hassan – Purdue University
History of DBMS

► Even from the early days of computers data must be stored for applications
► Late 1960 IBM’s Information Management System (IMS) for airline reservations.
► 1970s Edgar Codd proposed a relational data model
► 1980s database query language SQL was standardized
► 1990s Data warehouses, consolidating data from multiple data stores for analysis
► 2000s Web applications
► Now – Even larger volumes of data NoSQL databases
Imagine writing a program for a bank
   - Customers, Accounts, MoneyTransfers
   - More than 500 GB (does not fit in memory)

Application must stage large datasets between main memory and secondary storage (500GB RAM is not still cheap!)

Must protect data from inconsistency (update in ATM should be consistent with bank branch)

Crash recovery

Security and access control

Concurrency (Transaction management)
Why Use a DBMS?

- Data independence and efficient access
- Reduced application and development time
- Data integrity and security
- Uniform data administration
- Concurrent access
- Recovery from crashes

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke
Example of a Traditional Database Application

Suppose we are building a system to store the information about:

- students
- courses
- professors
- who takes what, who teaches what
Can we do it without a DBMS?

Sure we can! Start by storing the data in files:

- students.txt
- courses.txt
- professors.txt

Now write C or Java programs to implement specific tasks.
Doing it without a DBMS...

- Enroll “Mary Johnson” in “CSE444”:

  Write a C/Java program to do the following:

  Read ‘students.txt’
  Read ‘courses.txt’
  Find&update the record “Mary Johnson”
  Find&update the record “CSE444”
  Write “students.txt”
  Write “courses.txt”
Problems without an DBMS...

► System crashes:
   - What is the problem?
   - Large data sets (say 50GB)
     - Why is this a problem?
   - Simultaneous access by many users
     - Lock students.txt – what is the problem?

CRASH!
DBMS

“Client-server”

Data files

Database server (someone else’s C program)

connection

Applications
Why Study Databases?

► Shift from computation to information
  ► Low-end users: Web Applications needs to organize information (a mess will not be effective)
  ► High-end users: Scientific applications now have data management problems!

► Datasets increasing in diversity and volume
  ► Digital libraries, interactive video, Human Genome project etc.

► DBMS encompasses most of CS
  ► OS, languages, AI, multimedia etc.
Data Models

- A **data model** is a collection of concepts for describing data. (high-level)
- A **schema** is a description of a particular collection of data, using the given data model
- The **relational model of data** is the most widely used model today.
  - Main concept: **relation**, basically a table with rows and columns
  - Every relation has a **schema**, which describes the columns, or fields.
  - Schema is defined by: name of schema, the name of each **field** (or **attribute** or **column**) and type of each field
  
  e.g. **Students**(*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real)
### Entity: Student

**Students** (sid: string, name: string, login: string, age: integer, gpa: real)

<table>
<thead>
<tr>
<th>Sid</th>
<th>name</th>
<th>login</th>
<th>age</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>jones@cs</td>
<td>18</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>smith@ee</td>
<td>18</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>smith@math</td>
<td>19</td>
<td>3.8</td>
</tr>
</tbody>
</table>

**Attribute**

(field or column)

**Integrity Constraints:**

We can define the field sid to be unique or age to be larger than 0. Rules for records to satisfy

**Using age as a field is not a good idea, why?**
Levels of Abstraction

- Unlike programmers of early systems, programmer of relational system does not need to implement lower level details
- Many views, single conceptual (logical) schema and physical schema.
  - Views describe how users see the data.
  - Conceptual schema defines logical structure
  - Physical schema describes the files and indexes used
Physical View

- The DBMS must know
  - exact physical location
  - precise physical structure

Employee record

A.B.C. De Silva          |222, Galle Road, Colombo |
Name (20 characters)     Address (40 characters)

650370690V|Senior Lecturer
NID (10 char)  Designation (15 char)
The conceptual model is a logical representation of the entire contents of the database.

The conceptual model is made up of base tables.

Base tables are “real” in that they contain physical records.
External View

- The user/application see
  - authorised data
  - own format
External views allow to

- hide unauthorised data
  - e.g. salary, dob
- provide user view
  - e.g. view employee name, designation, department data taken from employee and department files
- derive new attributes
  - e.g. age derived from dob or nid
Example: University Database

- Conceptual schema:
  - Students(sid:string, name:string, login:string, age:integer, gpa:real)
  - Courses(cid:string, cname:string, credits:integer)
  - Enrolled(sid:string, cid:string, grade:string)

- Physical schema:
  - Relations stored as unordered files
  - Index on first column of Students

- External Schema (View):
  - Course_info(cid:string, enrollment:integer)
Data Independence

- Applications 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!
Concurrent execution of user programs is essential for good DBMS performance
  - Because disk accesses are frequent and relatively slow, it is important to keep the CPU humming by working on several user programs concurrently.

Interleaving actions of different user programs can lead to inconsistency

DBMS ensures such problems don’t arise.

Users can pretend they are using a single-user system.
- Two users performing operations on a joint account at the same time.
- If one reads before the other writes back, the first to write will be cancelled.
- It will work ok if read and insert is atomic (not interrupted).
- To make sure, we can lock the account.
A prepaid mobile phone user will transfer 10 credit to User 2.

This operation needs two steps

If trying to remove 10 credits from User 1 fails for some reason, we have added 10 credits to U2 out of the blue

If we perform the operation in a transaction, we can roll-back the changes.
A typical DBMS has a layered architecture.

The figure does not show the concurrency control and recovery components.

This is one of several possible architectures; each system has its own variations.
An overview of Database Concepts

Querying
- Relational Algebra/Calculus
- Structured Query Language (SQL)
- Entity Relationship (ER) Diagrams
- Normalization / Functional Dependencies
- Data Definition Language (DDL)

Database Design
- Locking Mechanisms (Avoiding Deadlocks)
- Crash Recovery
- Utilizing the Memory Hierarchy (Buffering)
- How to Store Data in Files
- Finding data fast; Indexing Structures
- External Sorting

Covered in BBM 471

Covered in this Course
Databases make these folks happy...

- End users and DBMS vendors
- DP application programmers
  - E.g. smart webmasters
- Database administrator
  - Design logical / physical schemas
  - Handles security and authorization
  - Data availability, crash recovery

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke
End of the first lecture...