CSCI-GA.2433-001

Database Systems

Lecture 1: Gentle Introduction to Database Systems

Mohamed Zahran (aka Z)
mzahran@cs.nyu.edu
http://www.mzahran.com
Who Am I?

- Mohamed Zahran (aka Z)
- http://www.mzahran.com
- Office hours: Wed 4:00-6:00 pm
  – or by appointment
- Room: WWH 320
**Formal Goals of This Course**

- What exactly is a database? Basically, gaining fundamental concepts of state-of-the-art database management systems.
- How to use it ... professionally? We will experiment with industrial strength tools.
- How to design a database? You will get to do it.
- How can it be optimized?
Informal Goals of This Course

• To learn databases and enjoy it!
• To use what you have learned in MANY different contexts
• To get more than an A
• To enhance your way of thinking about how to manage data
The Course Web Page

- Lecture slides
- Info about mailing list, labs, ...
- Useful links (tools, articles, ...)

http://cs.nyu.edu/courses/Fall12/CSCI-GA.2433-001/index.html
The Textbook

Database Management Systems

By

Raghu Ramakrishnan and Johannes Gehrke

3rd Edition
Grading

- Homework assignments  20%
- Projects            15%
- Midterm             30%
- Final               35%
Did You Know That …

• Humans are creating at least 1.8 zettabytes of data a year
• That is enough to fill 57.5 billion 32GB ipads!

• How will we manage this?

You may not notice it, but databases are behind almost everything you do on the Web!
is raw. It simply exists and has no significance beyond its existence...

is data that has been given meaning by way of relational connection.

A nice tool to do that: DB
Database

- Collection of data
- Entities + relationship between entities

DB Management System (DBMS)

Software designed to assist in maintaining and utilizing large collections of data.
What if We Don’t Want to Use DBMS?

• Alternative:
  – Store data in files (traditional OS file system)
  – Write application-specific code to manage it.

• What’s bad about it?
  – Special program for every scenario
  – Must protect the data from inconsistent changes
  – Dealing with crashes
  – Security?
Advantages of DBMS

- Data independence: DBMS hides details and provide an abstract view of the data
- Efficient storage and retrieval of data
- Enforces integrity and security
- Reduces data redundancy
- Concurrent access
- Crash recovery
- Reduces application development time
A Question

Is there ever a reason NOT to use a DBMS?
Aren’t DBs nothing but a set of tables stored in files on a disk? Why all this sophistication we are discussing?
The Need for A Good Design

<table>
<thead>
<tr>
<th>Name</th>
<th>SSN</th>
<th>Age</th>
<th>Grade</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>121</td>
<td>52</td>
<td>2</td>
<td>80</td>
</tr>
<tr>
<td>A</td>
<td>132</td>
<td>38</td>
<td>3</td>
<td>70</td>
</tr>
<tr>
<td>B</td>
<td>101</td>
<td>61</td>
<td>4</td>
<td>70</td>
</tr>
<tr>
<td>C</td>
<td>106</td>
<td>24</td>
<td>2</td>
<td>80</td>
</tr>
</tbody>
</table>

Is there something we can do to make the above table better?
How to design a database and use DBMS efficiently?

It is necessary to understand how DBMS works

- DB Design & Application de vel.
- Data Analysis
- Concurrency & Robustness
- Efficiency & Scalability
Entities in the organization, relationships, ...

Very High Level

Bits, bytes, pages, files, ...

Levels of Abstraction

DBMS

DB
A Bit of Vocabulary

- **Data model**: collection of high-level data description constructs that hide many low-level storage details.
  - Closer to DBMS than the user
  - A description of data in terms of data model is called **schema**.
  - Example: Relational data model

- **Semantic data model**: more abstract high-level data model
  - Closer to user
  - Serves as a useful starting point
  - Example: entity-relationship (ER) model
Relational Data Model

• The central description is: relation
• The schema describes, for each relation
  – name
  – name of each field
  – type of each filed

Students(sid: string, name: string, login: string,
        age: integer, gpa: real)
Relational Data Model

Students\((sid:\ \text{string}, \ name:\ \text{string}, \ login:\ \text{string}, \ age:\ \text{integer}, \ gpa:\ \text{real})\)

A record (instance of a relation)

<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>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>madayan@music</td>
<td>11</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>guldu@music</td>
<td>12</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Why Relations?

• Very simple model.
• *Often* matches how we think about data.
• Abstract model that underlies SQL, the most important database language today.
Several external views

- Only one conceptual and one physical schemas.
- Information about the schemas is stored in the **system catalogs**
- A **Data Definition Language** (DDL) is used to define external and conceptual schemas
- Example of DDL: SQL
Levels of Abstraction in a DBMS

The Conceptual schema: Describes the stored data in terms of the data model of DBMS

Example:

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)
Rooms(rno: integer, address: string, capacity: integer)
Enrolled(sid: string, cid: string, grade: string)
Teaches(fid: string, cid: string)
Meets_In(cid: string, rno: integer, time: string)
Levels of Abstraction in a DBMS

The Physical schema: Summarizes how the relations described in the conceptual schema are stored on storage devices (file organizations, indexes, ...)

Physical Schema

Conceptual Schema

View 1  View 2  View 3

Disk
Levels of Abstraction in a DBMS

External schema:
- Allows data access to be customized at the level of individual users.
- Each external schema = collection of 1 or more views and relations from the conceptual schema
- A view = a relation but its records are not stored in DBMS
source: http://rfuentesb.wordpress.com/2010/12/
Interacting with DBMS

• Queries

• The tools we have:
  – DDL
  – Data Manipulation Language (DML): enables users to create, modify, and query data.

• Transaction: an atomic sequence of database actions (reads/writes).
  – DBMS needs to deal with concurrency
    • Locking protocol
  – DBMS needs to deal with system crashes
    • Maintaining a log (write ahead log)
Simplified View of DB Environment

Users/Programmers

Database System

Application Programs/Queries

DBMS Software

Software to Process Queries/Programs

Software to Access Stored Data

Stored Database Definition (catalog)

Stored Database
Structure of DBMS

- Query Optimization and Execution
- Relational Operators
- Files and Access Methods
- Buffer Management
- Disk Space Management

DB
People Working with DB

- End users
- DBMS vendors
- DB application programmers
- Database administrator (DBA)
  - Designs logical / physical schemas
  - Handles security and authorization
  - Data availability, crash recovery
  - Database tuning as needs evolve
Database Design Steps

1. Requirements analysis:
   – What data is to be stored?
   – What applications must be built on top?
   – What operations are most frequent?

2. Conceptual database design:
   – High-level description of data to be stored and constraints
   – Must be very precise
   – Facilitates discussion among all the people involved in the design process
   – Often carried out using ER model
Database Design Steps

3. Logical database design:
   - Choose DBMS
   - Conceptual design \(\rightarrow\) logical schema in the data model of the chosen DBMS

4. Schema refinement:
   - Identify potential problems and refine

5. Physical database design:
   - Refine the design to reach required performance under the expected workload
Database Design Steps

6. Application and security design:
   – Aspects of the application that go beyond the database itself
   – Enforce access rules
DB & DBMS Are Really Important!

Charles Bachman

Designed the first general purpose DBMS

Recipient of the first ACM’s Turing Award
DB & DBMS Are Really Important!

Edgar Codd

Proposed the relational data model

Recipient of the 1981 Turing Award
DB & DBMS Are Really Important!

Jim Gray

Contributed to DB Transaction management

Recipient of the 1999 Turing Award
Conclusions

• We are in the era of data explosion
• DB are pivotal for us to interact with these data
• From DB to users there are several levels of abstractions implemented by DBMS … This is what we will study in the rest of the lectures

Reading: 1.1, 1.4-1.6, 1.8