

Database Systems (資料庫系統)

9/17/2008

Lecture #1

1

Course Goals

- First course in database systems.
- Learning objective
 - Use a relational database
 - Build a relational database

2

Topics

- Fundamentals
 - ER (Entity-Relationship) Model
 - SQL (Structured Query Language)
- Storage and indexing
 - Disks & Files
 - Tree-structure indexing
 - Hash-based indexing
- Query evaluation
 - External sorting
 - Evaluating relational operators
- Transaction management:
 - Concurrency control
 - Crash recovery
- Other Topics

3

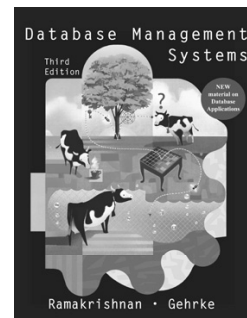
Prerequisite

- Data structure and algorithms
- English skill
 - Taught in English & Chinese
 - If I speak English too fast, please tell me to slow down.
 - You can ask questions in English or Chinese.

4

Textbook

- Required textbook: “Database Management Systems, Third Edition”, by Ramakrishnan and Gehrke.
- The textbook is available from 新月 and (and perhaps other) bookstores.



5

General Comments on Textbook

- Widely used among U.S. Universities 3~4 years ago.
- Bad
 - Ambiguous writing, inconsistent wording
 - “More like an experience report from researchers rather than introductory textbooks for beginners”

Ask me & TAs for clarifications

6

Course Format

- 7 Assignments
 - SQL + Written assignments
- Midterm Exam
 - Fixed date: Nov 12, 2008 (Wed) in class
 - Offer once only!
- Final Exam
 - Fixed date: Jan 14, 2009 (Wed) TBD
 - Offer once only!

7

Grading Breakdown (Tentative)

- Tentative means that I may change it later.
 - 7 Assignments (10% of Grade)
 - Midterm Exam (45% of Grade)
 - Final Exam (45% of Grade)

8

Office Hours & Contact Info

- Winston Hsu (徐宏民)
 - Office hour: Room 512
 - Email: winston@csie.ntu.edu.tw
- Instructor: 朱浩華 “Hao”
 - Room 518 or by appointment
 - Email: hchu@csie.ntu.edu.tw

9

Means of Communications

- Course homepage
 - http://mll.csie.ntu.edu.tw/course/database_f08/index.php
- BBS
 - ptt.cc, under “CSIE_DBMS” board
 - Post your questions on BBS.
 - Read posted messages before posting new questions.
 - No SPAM.
 - TAs respond to your questions as quickly as possible.
- Send email to TAs or me.
- Come to office hours

10

Lecture Notes

- Available on the course homepage before each lecture
 - Complements, not replacement of attending lecture and reading textbook.

11

**Any Question(s) on
Administrative Things?**

12

Introduce an interesting project in Ubiquitous Computing

(Won't be Tested)

13

Topobo (MIT media lab)

- Redefine programming
 - Create a program without “writing a program”.



14

Chapter I: Overview of Database Systems

15

Outline

- Why do we need a DBMS (Database Management System)?
- What can a DBMS do for an application?
- Why study database systems?
- Data Models: Overview of a Relational Model
- Levels of Abstraction in a DBMS
- Sample Queries in DBMS
- Transaction Management Overview
- Structure of a DBMS

16

Why DBMS?

- Suppose that you want to build an university database. It must store the following information:
 - Entities: Students, Professors, Classes, Classrooms
 - Relationships: Who teaches what? Who teaches where? Who teaches whom?

17

What can DBMS do for applications?

- Store huge amount of data (e.g., TB+) over a long period of time
- Allow apps to query and update data
 - Query: what is Mary's grade in the "Operating System" course?
 - Update: enroll Mary in the "Database" course
- Protect from unauthorized access.
 - Students cannot change their course grades.
- Protect from system crashes
 - When some system components fail (hard drive, network, etc.), database can be restored to a good state.

18

More on what can DBMS do for applications?

- Protect from incorrect inputs
 - Mary has registered for 100 courses
- Support concurrent access from multiple users
 - 1000 students using the registration system at the same time
- Allow administrators to easily change data schema
 - At a later time, add TA info to courses.
- Efficient database operations
 - Search for students with 5 highest GPAs

19

Alternative to Using a DBMS

- Store data as files in operating systems.
- Applications have to deal with the following issues:
 - 32-bit addressing (5GB) is insufficient to address 100GB+ data file
 - Write special code to support different queries
 - Write special code to protect data from concurrent access
 - Write special code to protect against system crashes
 - Optimize applications for efficient access and query
 - May often rewrite applications
- Easier to buy a DBMS to handle these issues

20

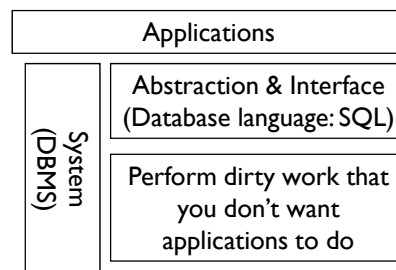
Database Management System (DBMS)

- DBMS is software to store and manage data, so applications don't have to worry about them.
- What can a DBMS do for applications?
 - Can you think of them?

21

What can a DBMS do for applications?

- Define data: Data Definition Language (DDL)
- Access and operate on data: Data Manipulation Language (DML)
 - Query language
- Storage management
- Transaction Management
 - Concurrency control
 - Crash recovery
- Provide good security, efficiency, and scalability



22

Why Study Database Systems?

- They are everywhere.
 - Online stores, real stores
 - Banks, credit card companies
 - Passport control
 - Police (criminal records)
 - Airlines and hotels (reservations)
- DBMS vendors & products
 - Oracle, Microsoft (Access and SQL server), IBM (DB2), Sybase, ...

23

Data Models

- A data model is a collection of concepts for describing data.
 - Entity-relation (ER) model
 - Relational model (main focus of this course)
- A schema is a description of data.
- The relational model is the most widely used data model.
 - A relation is basically a table with rows and columns of records.
 - Every relation has a schema, which describes the columns, or fields.

24

Relational Model

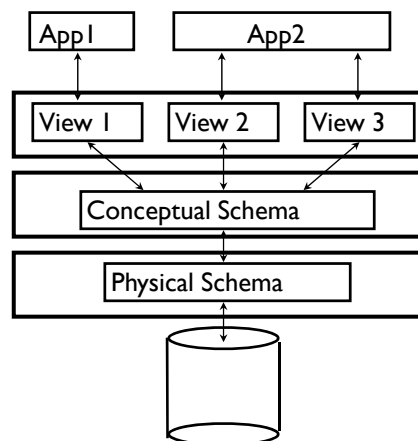
- The entire table shows an instance of the Students relation.
- The Students schema is the column heads
 - Students(Sid: String, Name: String, Login: String, age: Integer,...)

sid	name	email	age	gpa
53666	Jones	Jones@cs	18	3.4
53688	Smith	Smith@ee	18	3.2
53650	Joe	Joe@cs	19	2.5

25

Levels of Abstractions in DBMS

- Many views, one conceptual schema and one physical schema.
 - Conceptual schema defines logical structure
 - Relation tables
 - Physical schema describes the file and indexing used
 - Sorted file with B+ tree index
 - Views describe how applications (users) see the data
 - Relation tables but not store explicitly



26

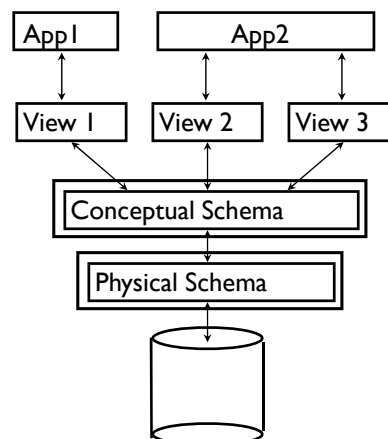
Example: University Database

- 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.
- View (External Schema):
 - Course_info(cid:string, enrollment:integer)
 - Why?

27

Data Independence

- Three levels of abstraction provides data independence.
 - Changes in one layer only affect one upper layer.
 - E.g., applications are not affected by changes in conceptual & physical schema.



28

Queries in DBMS

- Sample queries on university database:
 - What is the name of the student with student ID 123456?
- The key benefits of using a relational database are
 - Easy to specify queries using a query language: Structured Query Language (SQL)

```
SELECT S.name
FROM Students S
WHERE S.sid = 123456
```
 - Efficient query processor to get answer

29

Transaction Management

- A transaction is an execution of a user program in a DBMS.
- Transaction management deals with two things:
 - Concurrent execution of transactions
 - Incomplete transactions and system crashes

30

Concurrency Control

- Example: two travel agents (A, B) are trying to book one remaining airline seat (two transactions), only one transaction can succeed in booking.

```
// num_seats is 1
Transactions A and B: if num_seats > 0, book the seat & num_seat--;
// overbook!
```

- How to solve this?

31

Concurrency Control (Solution)

```
// num_seats is 1
Transactions A and B: if num_seats > 0, book the seat & num_seat--;
// overbook!
```

- Solution: use locking protocol

```
Transaction A: get exclusive lock on num_seats
Transaction B: wait until A releases lock on num_seats
Transaction A: if num_seats > 0, book & num_seat--;
// book the seat, num_seat is
set to 0
Transaction A: release exclusive lock on num_seats
Transaction B: num_seats = 0, no booking; // does not book the
seat
```

32

Crash Recovery

- Example: a bank transaction transfers \$100 from account A to account B

A = A - \$100

<system crashes> // good for the bank!

B = B + \$100

- How to solve this?

33

Crash Recovery (Solution)

A = A - \$100

<system crashes> // good for the bank!

B = B + \$100

- Solution: use logging, meaning that all write operations are recorded in a log on a stable storage.

A = A - \$100 // recorded A value (checkpoint) in a log

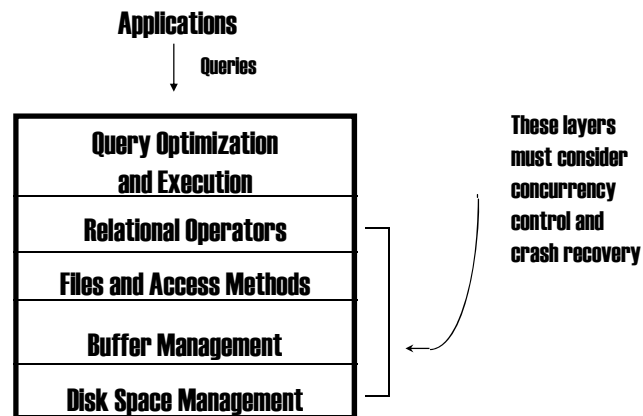
<system crashes>

// start recovery: read the log from disk

//analyze, undo, & redo

34

Layered Architecture



35

Administration

- Prof. Chu will be away at a conference next Wed
 - Please attend the lecture at CSIE 102
- Reading assignments
 - Read Chapters 1
 - Read Chapter 2 (except 2.7) for next lecture

36