CSE544
Introduction

Monday, March 29, 2004

Staff

- Instructor: Dan Suciu
  - CSE 662, suciu@cs.washington.edu
  - Office hours: Tuesday, 1-2pm.
- TA: Nilesh Dalvi
  - Office hours: Fri 12:00 to 1:00.
- Mailing list: cse544@cs.washington.edu
  - http://mailman.cs.washington.edu/mailman/private/cse544
- Web page: (a lot of stuff already there) http://www.cs.washington.edu/544

Course Times

- Mon, Wed, 12-1:30pm

- Guest lecturer: Nilesh, on March 31st

- Final:
  - 8:30-10:20 a.m. Thursday, Jun. 10, 2004

Goals of the Course

- Purpose:
  - Using database systems
  - Foundations of data management.
  - Issues in building database systems.
  - Introduction to current research in databases.

Grading

- Homeworks: 25%
  - HW1: programming (SQL, Xquery, …)
  - HW2: theory
- Project: 30%
  - More later.
- Paper reviews: 20%
- Final: 25%

Textbook

- **Database Management Systems**,
  Ramakrishnan and Gehrke.

Also:

- **Foundations of Databases**,
  Abiteboul, Hull & Vianu

Also:

- Some papers to read (see website)

Other Useful Texts

- *Database systems: the complete book* (Ullman, Widom and Garcia-Molina)
- *Parallel and Distributed DBMS* (Ozsu and Valduriez)
- *Transaction Processing* (Gray and Reuter)
- *Data and Knowledge based Systems* (volumes I, II) (Ullman)
- *Data on the Web* (Abiteboul, Buneman, Suciu)
- *Readings in Database Systems* (Stonebraker and Hellerstein)
- Proceedings of SIGMOD, VLDB, PODS conferences.

Prerequisites

- Officially: none
- Real prerequisites:
  - Programming languages
  - Logic
  - Complexity theory
  - Algorithms and data structures

What *Is* a Relational Database Management System ?

Database Management System = DBMS

Relational DBMS = RDBMS

- A collection of files that store the data
- A big C program written by someone else that accesses and updates those files for you
- Examples: Oracle, DB2, SQL Server

Where are RDBMS used ?

- Backend for traditional “database” applications
- Backend for large Websites
- Backend for Web services

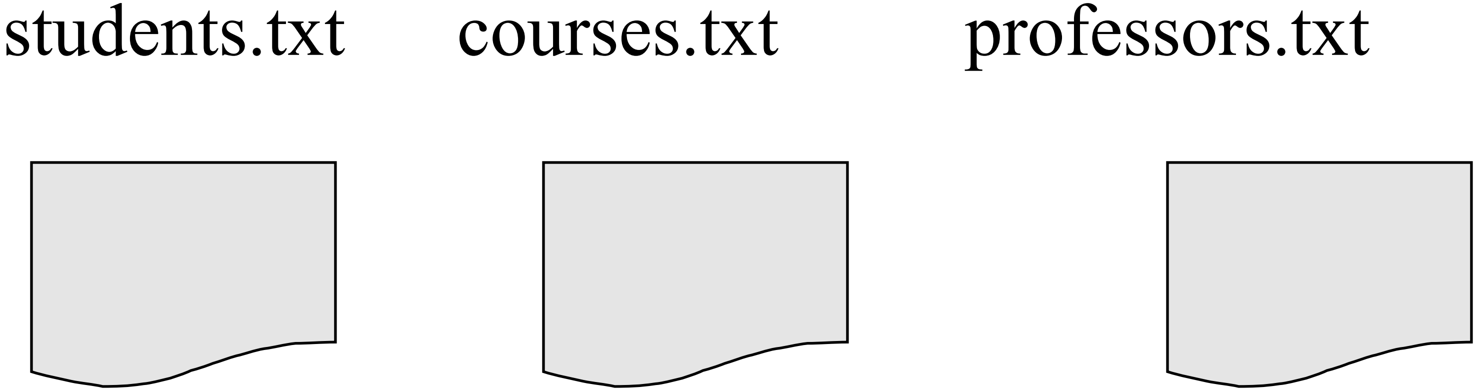
Example of a Traditional Database Application

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...

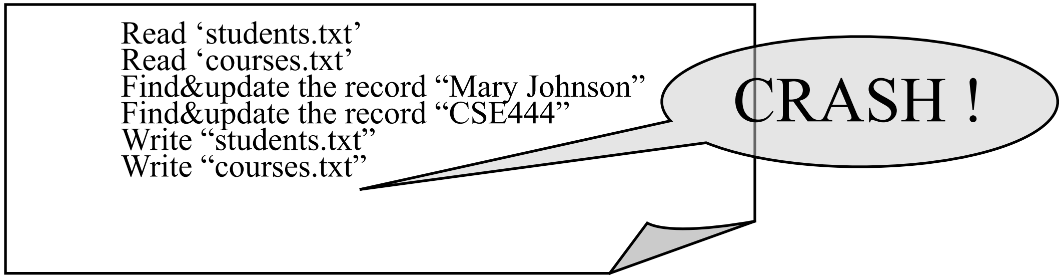
- Enroll “Mary Johnson” in “CSE444”:

Write a C 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:

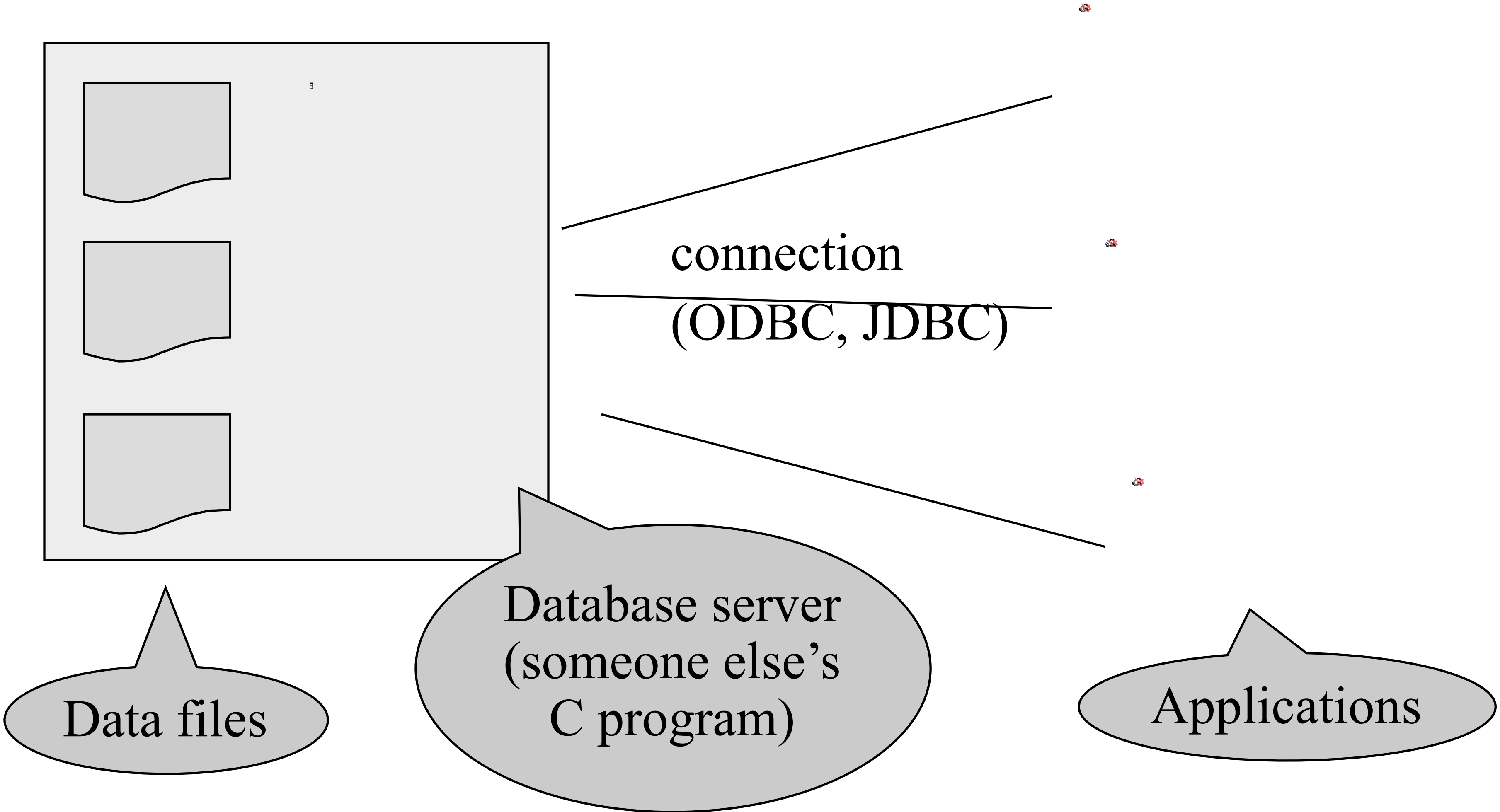


```
Read ‘students.txt’
Read ‘courses.txt’
Find&update the record “Mary Johnson”
Find&update the record “CSE444”
Write “students.txt”
Write “courses.txt”
```

  - 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 ?

Enters a DMBS

“Two tier system” or “client-server”



connection
(ODBC, JDBC)

Data files

Database server
(someone else’s
C program)

Applications

Functionality of a DBMS

The programmer sees SQL, which has two components:

- Data Definition Language - DDL
- Data Manipulation Language - DML
  - query language

Behind the scenes the DBMS has:

- Query engine
- Query optimizer
- Storage management
- Transaction Management (concurrency, recovery)

How the Programmer Sees the DBMS

- Start with DDL to *create tables*:

```
CREATE TABLE Students (
        Name CHAR(30)
        SSN CHAR(9) PRIMARY KEY NOT NULL,
        Category CHAR(20)
)  . . .
```

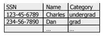
- Continue with DML to *populate tables:*

```
INSERT INTO Students
VALUES(‘Charles’, ‘123456789’, ‘undergraduate’)
.   .   .   .
```

How the Programmer Sees the DBMS

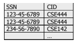
- Tables:

Students:



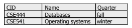
SSN	Name	Category
123-45-6789	Charles	undergrad
234-56-7890	Dan	grad
	…	…

Takes:



SSN	CID
123-45-6789	CSE444
123-45-6789	CSE444
234-56-7890	CSE142
	…

Courses:



CID	Name	Quarter
CSE444	Databases	fall
CSE541	Operating systems	winter

- Still implemented as files, but behind the scenes can be quite complex

“*data independence*” = separate *logical* view from *physical implementation*

Transactions

- Enroll “Mary Johnson” in “CSE444”:

```
BEGIN TRANSACTION;

INSERT INTO Takes
    SELECT Students.SSN, Courses.CID
    FROM Students, Courses
    WHERE Students.name = ‘Mary Johnson’ and
                   Courses.name = ‘CSE444’

-- More updates here....

IF everything-went-OK
      THEN COMMIT;
ELSE ROLLBACK
```

If system crashes, the transaction is still either committed or aborted

Transactions

- A *transaction* = sequence of statements that either all succeed, or all fail
- Transactions have the ACID properties:

  A = atomicity

  C = consistency

  I = isolation

  D = durability

Queries

- Find all courses that “Mary” takes

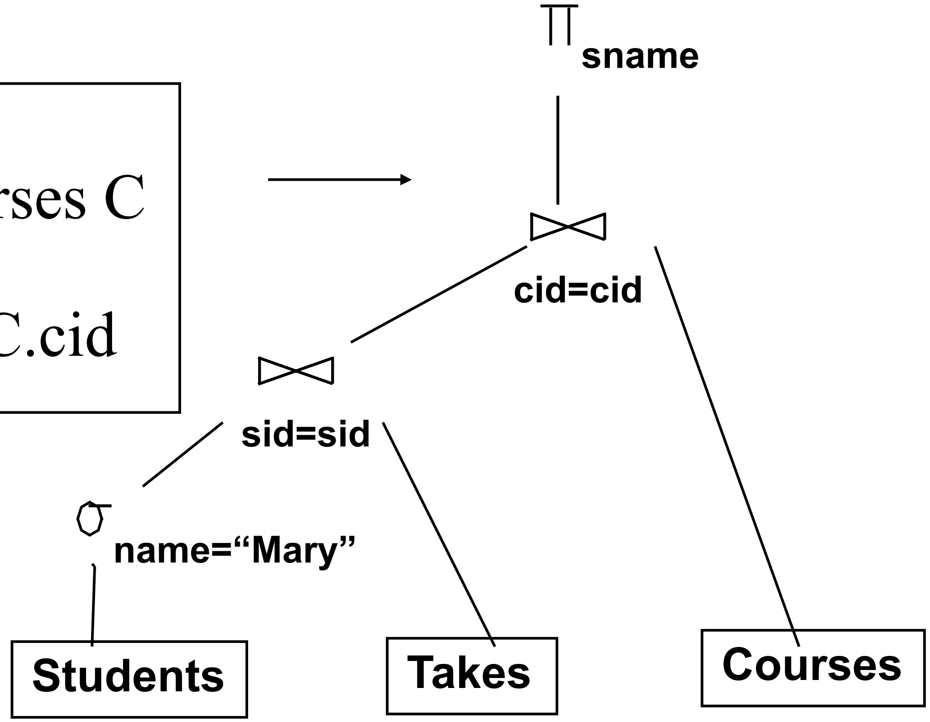
```
SELECT  C.name
FROM    Students S, Takes T, Courses C
WHERE   S.name=“Mary” and
        S.ssn = T.ssn and T.cid = C.cid
```

- What happens behind the scene ?
  - Query processor figures out how to answer the query efficiently.

Queries, behind the scene

Declarative SQL query → *Imperative query execution plan:*

```
SELECT  C.name
FROM Students S, Takes T, Courses C
WHERE S.name="Mary" and
          S.ssn = T.ssn and T.cid = C.cid
```



The **optimizer** chooses the best execution plan for a query

Database Systems

- The big commercial database vendors:
  - Oracle
  - IBM (with DB2)
  - Microsoft (SQL Server)
  - Sybase
- Some free database systems (Unix) :
  - Postgres
  - MySQL
  - Predator
- In CSE544 we use SQL Server and Postgres.

New Trends in Databases

- Object-relational databases
- Main memory database systems
- XML XML XML !
  - Relational databases with XML support
  - Middleware between XML and relational databases
  - Large-scale XML message systems
- Peer data management
- Stream data management
- Model management
- Security in data exchange
- Queries with uncertain matches

Database Industry

- Relational databases are a great success of theoretical ideas.
- Main players: Oracle, IBM, MS, Sybase
- Industry trends:
  - warehousing and decision support
  - data integration
  - XML, XML, XML.

What is the Field of Databases ?

- To a theoretical researcher (PODS/ICDT/LICS)
  - Focus on the query languages
  - Query language = logic = complexity classes
- To an applied researcher (SIGMOD/VLDB/ICDE)
  - Query optimization
  - Query processing (yet-another join algorithm)
  - Transaction processing, recovery (but most stuff is already done)
  - Novel applications: data mining, high-dimensional search
- To a systems programmer at Oracle:
  - Millions lines of code
- To an application builder:
  - E/R, SQL, ODBC/JDBC

Database Research

What is cool today:

- XML:
  - Theory: trees + logic + automata = ?
  - Implementation on top of a RDBMS
  - Native implementation, indexing
- Processing data streams
- Model management
- Security/privacy in global data sharing
- Queries with uncertainties

Course Outline

- Part 1: Query languages, conceptual design
  - What you need if you need a job
- Part 2: Database theory
  - What you need to know if you don’t need a job
- Part 3: Systems
  - What you need to know if you are hired by one of the four database companies

Homework: 25%

HW1: Get familiar with the technology:

- Design a small website powered by a db
- Process/query/manipulate XML data

- Will be handed out on Wednesday

HW2: Pure theory

Project: 30%

- General theme: apply database principles to a new problem
- Suggested topics will be on the Website in about a week.
- Groups of 2-3
- Groups assembled: Wednesday, 4/7
  - Email Nilesh the group name, and members
- Proposals: Wednesday, 4/14
- Touch base with me: every two weeks.
- Start Early.

Paper Reviews: 20%

- There will be reading assignments
  - More in the second half of the quarter

- You have to write the reviews before the day of class

Final: 25%

- June 10, 8:30-10:30, same room
- Will be challenging (and fun !)