Introduction to Database Systems
CSE 444

Lecture 1

Introduction

Staff

- Instructor: Hal Perkins
  - CSE 548, perkins@cs.washington.edu
    Office hours: CSE labs tba, office drop-ins and appointments welcome
- TA: Paramjit Sandhu
  - paramsan at cs
  - Office hours: CSE labs tba
- TA: Fanny Halim
  - fannyh at u
  - Office hours: CSE labs tba

Communications

- Web page: http://www.cs.washington.edu/444
  - Lectures, homework, projects will be available there
- Discussion list
  - See the web page
  - Discussions about the course, databases, etc. Stay in touch outside class
- Mailing list
  - Mostly announcements, intent is fairly low traffic
  - You are already subscribed if you are registered

Textbook

Main textbook, available at the bookstore:

- *Database Systems: The Complete Book*
  Hector Garcia-Molina,
  Jeffrey Ullman,
  Jennifer Widom

Most important: COME TO CLASS ! ASK QUESTIONS !

Other Texts

Available at the Engineering Library

(not on reserve – would anyone care if they were?):

- *Database Management Systems*, Ramakrishnan
- *XQuery from the Experts*, Katz, Ed.
- *Fundamentals of Database Systems*, Elmasri, Navathe
- *Foundations of Databases*, Abiteboul, Hull, Vianu
- *Data on the Web,* Abiteboul, Buneman, Suciu

Course Format

- Lectures MWF, 12:30 – 1:20 pm
- Quiz sections: Th 9:30-10:20, 10:30-11:20
  - Currently EEB 003 / EEB 045, but we're trying to move them to a single room – stay tuned!

- 4 Mini-projects
- 3 homework assignments

- Midterm and final

Grading

• Homeworks	30%
• Mini-projects	30%
• Midterm	15%
• Final	25%

Four Mini-Projects

1. SQL
2. SQL in Java
3. Database tuning
4. Parallel processing: MapReduce

Due: Wednesdays every other week

Three Homework Assignments

1. Conceptual Design
2. Transactions
3. Query execution and optimization

Due: Wednesdays every other week

Late Policy

- You have 4 late days to use during the quarter however you wish
  - No more than 2 on any single assignment or project
  - Used in 24 hour chunks
  - No other late assignments accepted

    (And we may specify no late days for particular assignments if needed to hand out solutions before exams or at the end of the quarter)

Academic Conduct

- We all learn best when we work with others, talk to colleagues, etc., and you definitely should do that, **but…**
- Anything you submit for credit is expected to be your individual work (or your group's work if the assignment specifically allows for that)
  – Enough said?

Exams

- Midterm: in class; tentatively Friday, November 13.
- Final: Thursday, Dec. 17, 8:30 am!

Outline of Today’s Lecture

1. Overview of a DBMS
2. A DBMS through an example
3. Course content

Database

What is a database ?

Give examples of databases

Database

What is a database ?

- A collection of files storing related data
- Our interest is mostly in “structured” data

Give examples of databases

- Accounts database; payroll database; UW’s students database; Amazon’s products database; airline reservation database

Database Management System

What is a DBMS ?

Give examples of DBMSs

Database Management System

What is a DBMS ?

- *A big C program written by someone else that allows us to manage efficiently a large database and allows it to persist over long periods of time*

Give examples of DBMSs

- DB2 (IBM), SQL Server (MS), Oracle, Sybase
- MySQL, PostgreSQL, …

We will focus on relational DBMSs most of the quarter

Market Shares

From 2007 Gartner report:

- IBM: 21% market with $3.2BN in sales
- Oracle: 47% market with $7.1BN in sales
- Microsoft: 17% market with $2.6BN in sales

An Example

The Internet Movie Database

http://www.imdb.com

- Entities:
  Actors (800k), Movies (400k), Directors, …

- Relationships:
  who played where, who directed what, …

Required Data Management Functionality

1. Describe real-world entities in terms of stored data
2. Create & persistently store large datasets
3. Efficiently query & update
    1. Must handle complex questions about data
    2. Must handle sophisticated updates
    3. Performance matters
4. Change structure (e.g., add attributes)
5. Concurrency control: enable simultaneous updates
6. Crash recovery
7. Security and integrity

DBMS Benefits

- Expensive to implement all these features inside the application
- DBMS provides these features (and more)
- DBMS simplifies application development

How to decide what features should go into the DBMS?

Back to Example: Tables

Actor:

id	fName	lName	gender
195428	Tom	Hanks	M
645947	Amy	Hanks	F
. . .			

Cast:

pid	mid
195428	337166
. . .	

Movie:

id	Name	year
337166	Toy Story	1995
. . .	. . .	. ..

SQL

```
SELECT *
FROM  Actor
```

SQL

```
SELECT count(*)
FROM  Actor
```

This is an *aggregate query*

SQL

```
SELECT *
FROM  Actor
WHERE lname = ‘Hanks’
```

This is a *selection query*

SQL

```
SELECT *
FROM  Actor, Casts, Movie
WHERE lname='Hanks' and Actor.id = Casts.pid
  and Casts.mid=Movie.id and Movie.year=1995
```

This query has *selections* and *joins*

We will learn SQL in all its glory in 4 lectures !

How Can We Evaluate the Query ?

Actor:

id	fName	lName	gender
. . .		Hanks	
. . .			

Cast:

pid	mid
. . .	
. . .	

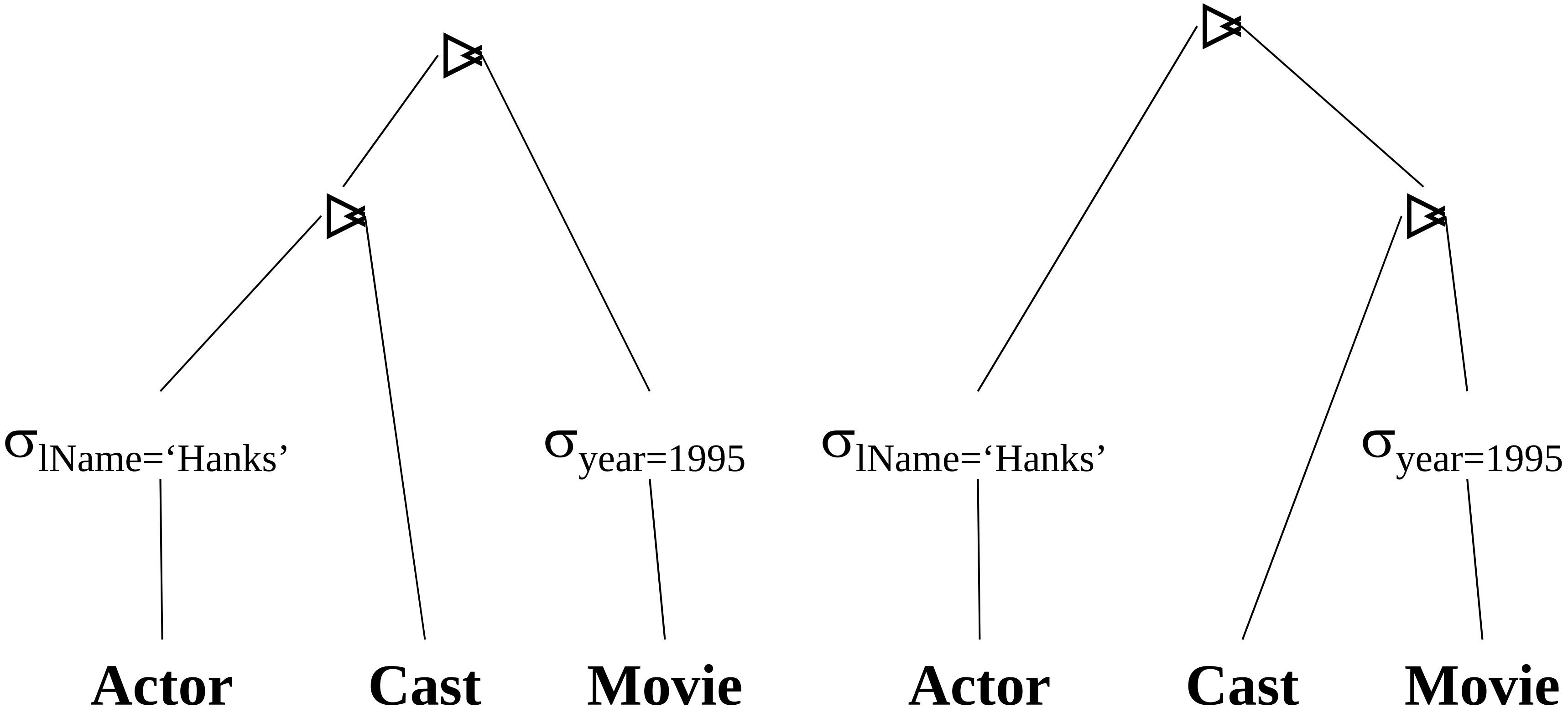
Movie:

id	Name	year
. . .		1995
. . .		

Plan 1: . . . . [in class]

Plan 2: . . . . [in class]

Evaluating Tom Hanks



⋈

⋈

$\sigma_{\text{lName='Hanks'}}$

$\sigma_{\text{year=1995}}$

Actor

Cast

Movie

⋈

⋈

$\sigma_{\text{lName='Hanks'}}$

$\sigma_{\text{year=1995}}$

Actor

Cast

Movie

What an RDBMS Does Well (1/2)

- Indexes: on Actor.lName, on Movie.year
- Multiple implementations of joins
- Query optimization (which join order ?)
- Statistics !

We’ll learn all about this in November

Now Let’s See Database Updates

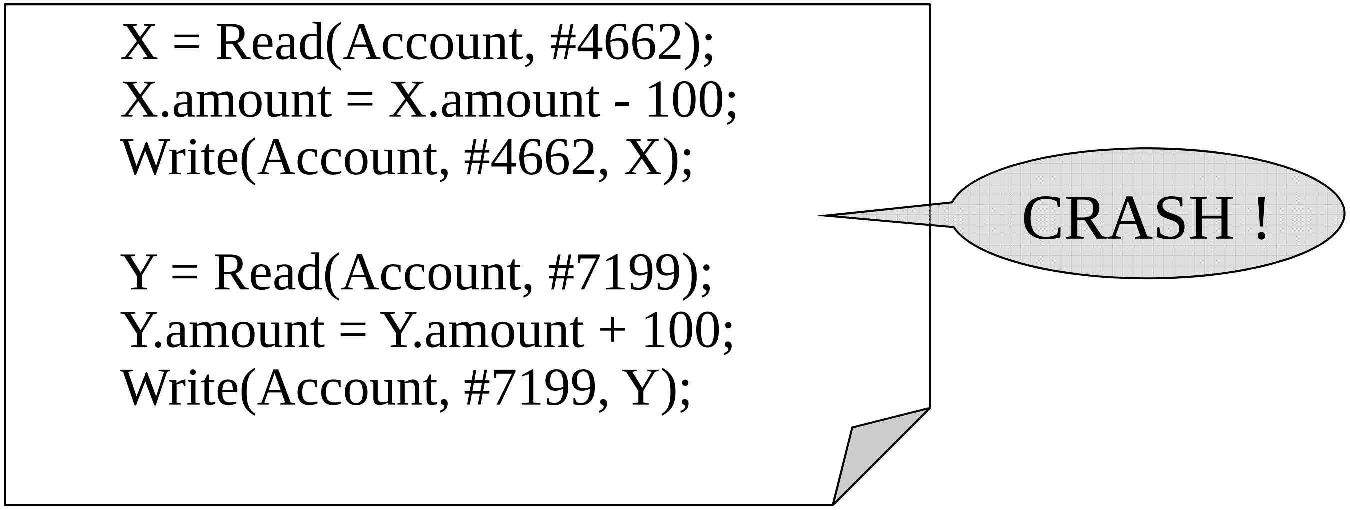
- Transfer $100 from account #4662 to #7199:

```
X = Read(Account, #4662);
X.amount = X.amount - 100;
Write(Account, #4662, X);

Y = Read(Account, #7199);
Y.amount = Y.amount + 100;
Write(Account, #7199, Y);
```

Now Let’s See Database Updates

- Transfer $100 from account #4662 to #7199:



```
X = Read(Account, #4662);
X.amount = X.amount - 100;
Write(Account, #4662, X);

Y = Read(Account, #7199);
Y.amount = Y.amount + 100;
Write(Account, #7199, Y);
```

CRASH !

What is the problem ?

What a RDBMS Does Well (2/2)

Transactions !

- Recovery
- Concurrency control

We will learn all that in October

Client/Server Architecture

- There is a single *server* that stores the database (called DBMS or RDBMS):
  - Usually a beefy system, e.g. IISQLSRV1
  - But can be your own desktop…
  - … or a huge cluster running a parallel dbms
- Many *clients* run apps and connect to DBMS
  - E.g. Microsoft’s SQL Server Management Studio
  - Or psql (for postgres)
  - More realistically some Java, C#, or C++ program
- Clients “talk” to server using JDBC protocol

What This Course Contains

- SQL
- Conceptual Design
- Transactions
- Database tuning and internals (very little)
- Distributed databases: a taste of *MapReduce*
- More data management if we have time
  - Sampling, data cleaning, etc.
- XML: Xpath, Xquery

Accessing SQL Server

(after tomorrow)

SQL Server Management Studio

- Server Type = Database Engine
- Server Name = IISQLSRV
- Authentication = SQL Server Authentication
  - Login = your UW email address (*not* CSE email)
  - Password = seattle

Change your password !!

Then play with IMDB, start working on PROJ1