Introduction to Database Systems
CSE 444

Lecture 1
Introduction
About Me: General

Prof. Magdalena Balazinska (magda)

- At UW since January 2006
- PhD from MIT
- Born in Poland
- Grew-up in Poland, Algeria, and Canada
About Me: Research

• **Past: Stream Processing**
  – Distributed stream processing (Borealis)
  – Load management and fault-tolerance
  – RFID data management (RFID Ecosystem)
  – Probabilistic event processing (Lahar)

• **Now: Cloud computing and scientific data mgmt**
  – Collaboration with astronomers, oceanographers, etc.
  – Making large-scale data analysis easier and interactive
  – Helping scientists leverage cloud computing
Staff

• **Instructor: Magdalena Balazinska**
  – CSE 550, magda@cs.washington.edu
  – Office hours: Wednesdays 10:30am-12:20pm

• **Ugrad TA: Michael Rathanapinta**
  – michaelr@cs.washington.edu
  – Office hours: Thursdays 10:30am-12:00pm in CSE 006

• **Ugrad TA: Liem Dinh**
  – liemdinh@cs.washington.edu
Communications

• **Web page:** [http://www.cs.washington.edu/444](http://www.cs.washington.edu/444)
  – Lectures will be available there
  – The mini-projects description will be there
  – Homeworks will be posted there

• **Mailing list**
  – Announcements, group discussions
  – You are already subscribed

• **Message board**
  – Great place to ask assignment-related questions
Textbook

Main textbook, available at the bookstore:


Most important: COME TO CLASS! ASK QUESTIONS!
Other Texts

Available at the Engineering Library (not on reserve):

- *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, 9:30am-10:20am
- Quiz sections: Th 8:30-9:20, 9:30-10:20
  - Location: EEB 025

- 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

Check course website for due dates
Three Homework Assignments

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

Check course website for due dates
Exams

• Midterm: Monday, November 8, in class

• Final: Wednesday, December 15, 8:30-10:20am, in class
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

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 quarter
Market Shares

From 2006 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:**

<table>
<thead>
<tr>
<th>id</th>
<th>fName</th>
<th>lName</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>195428</td>
<td>Tom</td>
<td>Hanks</td>
<td>M</td>
</tr>
<tr>
<td>645947</td>
<td>Amy</td>
<td>Hanks</td>
<td>F</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Cast:**

<table>
<thead>
<tr>
<th>pid</th>
<th>mid</th>
</tr>
</thead>
<tbody>
<tr>
<td>195428</td>
<td>337166</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Movie:**

<table>
<thead>
<tr>
<th>id</th>
<th>Name</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>337166</td>
<td>Toy Story</td>
<td>1995</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Magda Balazinska - CSE 444, Fall 2010
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*

817K actors, 3.5M casts, 380K movies;
How long do we expect it to take?
How Can We Evaluate the Query?

<table>
<thead>
<tr>
<th>Actor:</th>
<th>Cast:</th>
<th>Movie:</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>fName</td>
<td>lName</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>Hanks</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Plan 1: . . . . [ in class ]

Plan 2: . . . . [ in class ]
Evaluating Tom Hanks
What an RDBMS Does Well (1/2)

• Indexes: on Actor.lName, on Movie.year
• Multiple implementations of joins
• Query optimization
  – Access path selection
  – Join order
  – Join implementation
• Statistics!
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);

What is the problem?
What a RDBMS Does Well (2/2)

Transactions!

- Recovery
- Concurrency control
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 Management Studio
  - Or psql (for postgres)
  - More realistically some Java 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
  – Sampling, data cleaning, etc.
• XML: Xpath, Xquery
Accessing SQL Server

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

Change your password!!

Then play with IMDB, start working on PROJ1