15-826: Multimedia Databases and Data Mining

Christos Faloutsos
CMU
www.cs.cmu.edu/~christos

Outline
Goal: ‘Find similar / interesting things’
  • Intro to DB
  • Indexing - similarity search
  • Data Mining

Problem
Given a large collection of (multimedia) records, find similar/interesting things, ie:
  • Allow fast, approximate queries, and
  • Find rules/patterns

Sample queries
  • Similarity search
    – Find pairs of branches with similar sales patterns
    – find medical cases similar to Smith’s
    – Find pairs of sensor series that move in sync
    – Find shapes like a spark-plug
    – (nn: ‘case based reasoning’)

Sample queries –cont’d
  • Rule discovery
    – Clusters (of branches; of sensor data; …)
    – Forecasting (total sales for next year?)
    – Outliers (eg., unexpected part failures; fraud detection)

Outline
Goal: ‘Find similar / interesting things’
  • Intro to DB
  • Indexing - similarity search
  • Data Mining
Detailed Outline

Intro to DB
- Relational DBMS - what and why?
  - inserting, retrieving and summarizing data
  - views; security/privacy
  - (concurrency control and recovery)
- Object-Relational DBMS - what and why?

What is the goal of rel. DBMSs

Electronic record-keeping:
Fast and convenient access to information.
Eg.: students, taking classes, obtaining grades;
- find my gpa
- <and other ad-hoc queries>

Intro to DB
- views; security/privacy
- (MS Access, mSQL, mySQL, MySQL)

Why Databases?

- Flexibility
- data independence (can add new tables; new attributes)
- data sharing/concurrency control
- recovery

Why NOT Databases?

- Price
- additional expertise (SQL/DBA)
- over-kill for small data sets

Main vendors/products

Commercial
- Oracle
- IBM/DB2
- MS SQL-server
- Sybase
- (MS Access, ...

Open source
- Postgres (UCB)
- mySQL, mSQL
- miniBase (Wisc)
- Predator (Cornell)
(www.acm.org/sigmod)
Detailed Outline
Intro to DB
- Relational DBMS - what and why?
  - inserting, retrieving and summarizing data
  - views; security/privacy
  - (concurrency control and recovery)
- Object-Relational DBMS - what and why?

How do DBs work?

sql> insert into student
  name char(20);

More than one tables - joins
Eg., roster (names only) for 15-826

sql> select name
  from student, takes
  where student.ssn = takes.ssn
  and takes.c-id = 15-826
SQL-DML

General form:

\[
\text{select } a_1, a_2, \ldots, a_n \\
\text{from } r_1, r_2, \ldots, r_m \\
\text{where } P \\
\text{[order by ...]} \\
\text{[group by ...]} \\
\text{[having ...]}
\]

Aggregation

Find ssn and GPA for each student

```
<table>
<thead>
<tr>
<th>student</th>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
<td>c-id</td>
</tr>
<tr>
<td>123</td>
<td>603</td>
</tr>
<tr>
<td>123</td>
<td>412</td>
</tr>
<tr>
<td>234</td>
<td>603</td>
</tr>
</tbody>
</table>
```

Aggregation

sql> select ssn, grade
from takes;

```
<table>
<thead>
<tr>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
</tr>
<tr>
<td>123</td>
</tr>
<tr>
<td>123</td>
</tr>
<tr>
<td>234</td>
</tr>
</tbody>
</table>
```

Aggregation

sql> select ssn, avg(grade)
from takes;

```
<table>
<thead>
<tr>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
</tr>
<tr>
<td>123</td>
</tr>
<tr>
<td>234</td>
</tr>
</tbody>
</table>
```

Detailed Outline

Intro to DB
- Relational DBMS - what and why?
  - inserting, retrieving and summarizing data
  - views; security/privacy
- (concurrency control and recovery)
- Object-Relational DBMS - what and why?
Views - what and why?

- suppose you ONLY want to see ssn and GPA (eg., in your data-warehouse)
- suppose secy is only allowed to see GPAs, but not individual grades
- -> VIEWS!

Views

Views = ‘virtual tables’

sql> create view fellowship as (select ssn, avg(grade) from takes group by ssn);

<table>
<thead>
<tr>
<th>ss</th>
<th>c-id</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>603</td>
<td>4</td>
</tr>
<tr>
<td>123</td>
<td>412</td>
<td>3</td>
</tr>
<tr>
<td>234</td>
<td>603</td>
<td>3</td>
</tr>
</tbody>
</table>

sql> select * from fellowship;

<table>
<thead>
<tr>
<th>ss</th>
<th>c-id</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>603</td>
<td>4</td>
</tr>
<tr>
<td>123</td>
<td>412</td>
<td>3</td>
</tr>
<tr>
<td>234</td>
<td>603</td>
<td>3</td>
</tr>
</tbody>
</table>

Detailed Outline

Intro to DB
- Relational DBMS - what and why?
  - inserting, retrieving and summarizing data
  - views; security/privacy
- (concurrency control and recovery)
- Object-Relational DBMS - what and why?
Detailed Outline

Intro to DB
• Relational DBMS - what and why?
  – inserting, retrieving and summarizing data
  – views; security/privacy
  – (concurrency control and recovery)
• Object-Relational DBMS - what and why?

Why more than RDBMSs?
• RDBMS: tuples, of numbers + strings
• What apps need only those?
  – Banks
  – Airlines
  – Retailer stores
  – ...
• Q: Other apps, with more req’s?

Why more than RDBMS’s
• Q: Other apps, with more req’s?
  • A:
    – text
    – multimedia; financial apps/forecasting
    – Geographic Inf. Sys.
    – CAD/CAM
    – Network management

Ideally, we’d like to:
• create a new data type (eg., ‘image’, ‘time-sequence’)
• define functions on it (like (dist(im1, im2))
• be able to ask queries like
  select * from employee
  where dist(employee.face, given-face) <= 10;

OR DBMSs
traditional DBMS + attempts to provide
• user defined data types
• support for large / complex objects
• (inheritance)
**SQL-3 proposed extensions**

- complex types (sets, lists, multisets)
- inheritance (IS-A hierarchies)
- User Defined Functions (UDFs)

**Complex types**

eg.

```sql
create type MyDate (  
  day decimal(2),  
  month char(3),  
  year decimal (4)  
);
```

**BLObs etc:**

- Large objects, eg., video, images, 3d-MRI scans
- new data types: LOB (=Large OBject)
  - BLOB: (up to 4Gb; binary: jpeg, mpeg, …)
  - CLOB: (up to 2Gb; character: english text)
  - NCLOB: (……………; multi-byte characters)

**Stored procedures**

```sql
SQL> create or replace procedure del-st-rec  
  (s-id number) as  
  begin  
    delete from student  
    where s-id = ssn;  
  end del-st-rec;  
SQL> execute del-st-rec ( 123 );
```

**Conclusions**

- (relational) DBMSs: electronic record keepers
- customize them with `create table` commands
- ask SQL queries to retrieve info

**Conclusions cont’d**

main advantages over flat files & scripts:
- logical + physical data independence (ie., flexibility of adding new attributes, new tables and indices)
- concurrency control and recovery for free
Conclusions cont’d

- OR-DBMS: user-defined data types (e.g., images), and U.D. functions.

For more info:

- Microsoft Access: available on ANDREW clusters (PC)
- postgres: on ANDREW unix machines
  - www.cs.cmu.edu/~olston/15-415/F05/HW/PostgreSQL_Readme.htm
- Ramakrishna + Gehrke, 3rd edition
- 15-415 web page, e.g.,
  - www.cs.cmu.edu/~olston/15-415/F05/