CMSC424: Database Design

Instructor: Amol Deshpande
amol@cs.umd.edu

Today

- Motivation
  - Why study databases?
- Syllabus
- Administrivia
  - Workload etc
- Data management challenges in a very simple application

- We will also discuss some interesting open problems/research directions

- No laptop use allowed in the class!!
Motivation: Data Overload

- There is a *HUGE* amount of data in this world
- Everywhere you see…
- Personal
  - Emails, data on your computer
- Enterprise
  - Banks, supermarkets, universities, airlines etc etc
- Scientific
  - Biological, astronomical
- World wide web
  - Social networks etc…

Motivation: Data Overload

- Much more is produced every day

“More data will be produced in the next year than has been generated during the entire existence of humankind”

IBM: “… in 2005, the amount of data will grow from 3.2 million exabytes to 43 million exabytes”

[“total amount of printed material in the world is estimated to be 5 exabytes…”]
Motivation: Data Overload

- Much more is produced every day

Wal-mart: 583 terabytes of sales and inventory data
Adds a billion rows every day
“we know how many 2.4 ounces of tubes of toothpastes sold yesterday and what was sold with them”

Why?

[“library of congress --> 20 TBs”]

Motivation: Data Overload

- Much more is produced every day

Neilsen Media Research: 20 GB a day; total 80-100 TB

From where ???
12000 households or personal meters
Extending to iPods and TiVos in recent years
Motivation: Data Overload

- Scientific data is literally astronomical on scale
  “Wellcome Trust Sanger Institute's World Trace Archive database of DNA sequences hit one billion entries.”

Stores all sequence data produced and published by the world scientific community

22 Tbytes and doubling every 10 months

"Scanning the whole dataset for a single genetic sequence... like searching for a single sentence in the contents of the British Library”

Motivation: Data Overload

- Automatically generated data through instrumentation
  “Britain to log vehicle movements through cameras. 35 million reads per day.”

Wireless sensor networks are becoming ubiquitous.

RFID: Possible to track every single piece of product throughout its life (“Gillette boycott”)
Motivation: Data Overload

- How do we do anything with this data?

- Where and how do we store it?
  - Disks are doubling every 18 months or so -- not enough
  - In many cases, the data is not actually recorded as it is; summarized first

- What if the disks crash?
  - Very common, especially with 1000’s of disks for each app

- What to do with the data?
  - text search?
    - “find the stores with the maximum increase in sales in last month”
    - “how much time from here to pittsburgh if I start at 2pm?”
      - Data is there; more will be soon (live traffic data)
      - Requires predictive capabilities
    - “live traffic management”
      - GPS data, camera data, cellphone data
      - Complicated control issues
Motivation: Data Overload

- What to do with the data?
  - Find videos with this incident
    - Not even clear “how” to do this
  - Mine the “blogs” to detect “buzz”
  - More and more need to convert “information” to “knowledge”
    - “Data mining”
  - Most of these are open problems; we won’t discuss them much

- Speed !!
  - With TB’s of data, just finding something (even if you know what), is not easy
    - Reading a file with TB of data can take hours
  - Imagine a bank and millions of ATMs
    - How much time does it take you to do a withdrawal?
    - The data is not local

- How do we ensure “correctness”? 
  - Can’t have money disappearing
  - Harder than you might think
More questions...

- How do we guarantee the data will be there 10 years from now?
- Privacy and security !!!
  - Every other day we see some database leaked on the web
- Data integration (e.g. Web)
- New kinds of data
  - Scientific/biological, Image, Audio/Video, Sensor data etc
- Out of scope
  - Interesting research challenges!

DBMS to the Rescue

- Provide a systematic way to answer many of these questions...
- Aim is to allow easy management of data
  - Store it
  - Update it
  - Query it
- Massively successful for structured data
  - Structured?
Structured vs Unstructured

- A lot of the data we encounter is structured
  - Some have very simple structures
  - E.g. Data that can be represented in tabular forms
  - Significantly easier to deal with
  - We will focus on such data for much of the class

<table>
<thead>
<tr>
<th>Account</th>
<th>Customer</th>
</tr>
</thead>
<tbody>
<tr>
<td>bname</td>
<td>cname</td>
</tr>
<tr>
<td>acct_no</td>
<td>cstreet</td>
</tr>
<tr>
<td>balance</td>
<td>ccity</td>
</tr>
<tr>
<td>Downtown Mianus</td>
<td>Jones</td>
</tr>
<tr>
<td>Perry</td>
<td>Smith</td>
</tr>
<tr>
<td>R.H</td>
<td>Hayes</td>
</tr>
<tr>
<td>500 700 400 350</td>
<td>500 700 400 350</td>
</tr>
</tbody>
</table>

- Some data has a little more complicated structure
  - E.g. graph structures
    - Map data, social networks data, the web link structure etc
  - In many cases, can convert to tabular forms (for storing)
  - Slightly harder to deal with
    - Queries require dealing with the graph structure
Collaborations Graph

Query: Find my Erdos Number.

Structured vs Unstructured

- Increasing amount of data in a semi-structured format
  - XML – Self-describing tags (HTML ?)
  - Complicates a lot of things
  - We will discuss this toward the end
Structured vs Unstructured

- A huge amount of data is unfortunately unstructured
  - Books, WWW
  - Amenable to pretty much only text search... so far
    - Information Retrieval deals with this topic
  - What about Google?
    - Google is mainly successful because it uses the structure

- Video? Music?

DBMS to the Rescue

- Provide a systematic way to answer most of these questions...
  - ... for structured data
  - ... increasing for semi-structured data
    - XML database systems have been coming up

- Solving the same problems for truly unstructured data remains an open problem
  - Much research in Information Retrieval community
DBMS to the Rescue

- They are everywhere !!
- Enterprises
  - Banks, airlines, universities
- Internet
  - Searchsystems.net lists 35568 37220 public records DBs
  - Amazon, Ebay, IMDB
- Blogs, social networks...
- Your computer (emails especially)
- ...

What we will cover...

- representing information
  - data modeling
- languages and systems for querying data
  - complex queries & query semantics
  - over massive data sets
- concurrency control for data manipulation
  - controlling concurrent access
  - ensuring transactional semantics
- reliable data storage
  - maintain data semantics even if you pull the plug
What we will cover...

- We will see...
  - Algorithms and cost analyses
  - System architecture and implementation
  - Resource management and scheduling
  - Computer language design, semantics and optimization
  - Applications of AI topics including logic and planning
  - Statistical modeling of data

What we will cover...

- We will mainly discuss structured data
  - That can be represented in tabular forms (called Relational data)
  - We will spend some time on XML
- Still the biggest and most important business
  - Well defined problem with really good solutions that work
    - Contrast XQuery for XML vs SQL for relational
  - Solid technological foundations
- Many of the basic techniques however are directly applicable
  - E.g. reliable data storage etc
- Many other data management problems you will encounter can be solved by extending these techniques
Administrivia Break

- Instructor: Amol Deshpande
  - 3221 AV Williams Bldg
  - amol@cs.umd.edu
  - Class Webpage:
    - Off of http://www.cs.umd.edu/~amol,
    - Or http://www.cs.umd.edu/class

- Email to me: write CMSC424 in the title.

- TA: Fatih Kaya

Administrivia Break

- Textbook:
  - Database System Concepts
    - Fifth Edition
    - Abraham Silberschatz, Henry F. Korth, S. Sudarshan

- Lecture notes will be posted on the webpage, if used
  - http://forum.cs.umd.edu
    - We will use this in place of a newsgroup
    - First resort for any questions
    - General announcements will be posted there
    - Register today!
Administrivia Break

- Workload:
  - 4 homeworks (10%)
  - 2 Mid-terms, Final (50%)
  - An SQL assignment (10%)
  - A programming assignment (10%)
  - An application development project (20%)

- Schedule on the webpage
- First assignment out next week, due a week later

Administrivia Break

- Grading
  - Fixed
  - 80+: A
  - 70+: B
  - 60+: C
    - Most had 40+ on non-exams last two times (out of 50)
  - 60-: D/F
Next..

- Data management challenges in a very simple application
  - Why we can’t use a file system to do database management

- Data Modelling
  - Going from conceptual requirements of a application to a concrete data model

Example

- Simple Banking Application
  - Need to store information about:
    - Accounts
    - Customers
  - Need to support:
    - ATM transactions
    - Queries about the data

- Instructive to see how a naïve solution will work
A *file-system* based solution

- Data stored in files in ASCII format
  - # seperated files in /usr/db directory
  - /usr/db/accounts
    - Account Number # Balance
      - 101 # 900
      - 102 # 700
      - ...
  - /usr/db/customers
    - Customer Name # Customer Address # Account Number
      - Johnson # 101 University Blvd # 101
      - Smith # 1300 K St # 102
      - Johnson # 101 University Blvd # 103
      - ...

- Write application programs to support the operations
  - In your favorite programming language
  - Withdrawals by a customer for amount $X from account Y
    - Scan /usr/db/accounts, and look for Y in the 1st field
    - Subtract $X from the 2nd field, and rewrite the file
  - To support finding names of all customers on street Z
    - Scan /usr/db/customers, and look for (partial) matches for Z in the address field
  - ...

What’s wrong with this solution?

1. Data redundancy and inconsistency
   - No control of redundancy

   **Customer Name # Customer Address # Account Number**
   - Johnson # 101 University Blvd # 101
   - Smith # 1300 K St # 102
   - Johnson # 101 University Blvd # 103
   ... 
   Especially true as programs/data organization evolve over time

   - Inconsistencies
     - Data in different files may not agree
     - Very critical issue

What’s wrong with this solution?

2. Evolution of the database is hard
   - Delete an account
     - Will have to rewrite the entire file
   - Add a new field to the accounts file, or
     split the customers file in two parts:
     - Rewriting the entire file least of the worries
     - Will probably have to rewrite all the application programs
What’s wrong with this solution?

3. Difficulties in Data Retrieval
   ◦ No sophisticated tools for selective data access
     • Access only the data for customer X
     • Inefficient to scan the entire file
   ◦ Limited reuse
     • Find customers who live in area code 301
     • Unfortunately, no application program already written
     • Write a new program every time?

What’s wrong with this solution?

4. Semantic constraints
   ◦ Semantic integrity constraints become part of program code
     • Balance should not fall below 0
     • Every program that modifies the balance will have to enforce this constraint
   ◦ Hard to add new constraints or change existing ones
     • Balance should not fall below 0 unless overdraft-protection enabled
     • Now what?
       • Rewrite every program that modifies the balance?
What’s wrong with this solution?

5. Atomicity problems because of failures

Jim transfers $100 from Acct #55 to Acct #376

1. Get balance for acct #55
2. If balance55 > $100 then
   a. balance55 := balance55 - 100
   b. update balance55 on disk
   c. get balance from database for acct #376
   d. balance376 := balance376 + 100
   e. update balance376 on disk

CRASH

Must be atomic
Do all the operations or none of the operations

What’s wrong with this solution?

6. Durability problems because of failures

Jim transfers $100 from Acct #55 to Acct #376

1. Get balance for acct #55
2. If balance55 > $100 then
   a. balance55 := balance55 - 100
   b. update balance55 on disk
   c. get balance from database for acct #376
   d. balance376 := balance376 + 100
   e. update balance376 on disk
   f. print receipt

CRASH

After reporting success to the user, the changes better be there when he checks tomorrow
What’s wrong with this solution?

7. Concurrent access anomalies

Joe@ATM1: Withdraws $100 from Acct #55
1. Get balance for acct #55
2. If balance55 > $100 then
   a. balance55 := balance55 – 100
   b. dispense cash
   c. update balance55

Jane@ATM2: Withdraws $50 from Acct #55
1. Get balance for acct #55
2. If balance55 > $50 then
   a. balance55 := balance55 – 50
   b. dispense cash
   c. update balance55

Balance would only reflect one of the two operations
Bank loses money
What’s wrong with this solution?

8. Security Issues
   ◦ Need fine grained control on who sees what
     • Only the manager should have access to accounts with balance more than $100,000
     • How do you enforce that if there is only one accounts file?

Summary: What’s wrong with this solution?

   ◦ Hard to control redundancy
   ◦ Hard to evolve the structure
   ◦ Data retrieval requires writing application programs
   ◦ Semantic constraints all over the place
   ◦ Not fast enough!
   ◦ Data consistency issues
     • Disk crashes etc
   ◦ Security

   Database management provide an end-to-end solution to all of these problems
How?

- The key insight is what's called *data abstraction*.

Data Abstraction

- Probably *the* most important purpose of a DBMS.
- Goal: Hiding *low-level details* from the users of the system.
- Through use of *logical abstractions*.
Data Abstraction

What data users and application programs see?

What data is stored?
- describe data properties such as data semantics, data relationships

How data is actually stored?
- e.g. are we using disks? Which file system?

Data Abstraction: Banking Example

- Logical level:
  - Provide an abstraction of *tables*
  - Two tables can be accessed:
    - *accounts*
      - Columns: account number, balance
    - *customers*
      - Columns: name, address, account number

- View level:
  - A teller (non-manager) can only see a part of the *accounts* table
    - Not containing high balance accounts
Data Abstraction: Banking Example

- Physical Level:
  - Each table is stored in a separate ASCII file
  - # separated fields

- Identical to what we had before?
  - BUT the users are not aware of this
    - They only see the tables
    - The application programs are written over the tables abstraction

- Can change the physical level without affecting users

- In fact, can even change the logical level without affecting the teller

DBMS at a Glance

1. Data Modeling
2. Data Retrieval
3. Data Storage
4. Data Integrity
Data Modeling

- A data model is a collection of concepts for describing data properties and domain knowledge:
  - Data relationships
  - Data semantics
  - Data constraints

- We will discuss two models extensively in this class
  - Entity-relationship Model
  - Relational Model

- Probably discuss XML as well

Data Retrieval

- Query = **Declarative** data retrieval program
  - describes *what* data to acquire, not *how* to acquire it
  - Non-declarative:
    - scan the *accounts* file
    - look for number 55 in the 2nd field
    - subtract $50 from the 3rd field
  - Declarative (posed against the *tables* abstraction):
    - Subtract $50 from the column named *balance* for the row corresponding to *account number* 55 in the *accounts* table
    - How to do it is not specified.

- Why?
  - Easier to write
  - More efficient to execute (why?)
Data Storage

- Where and how to store data?
  - Main memory?
  - What if the database larger than memory size?
  - Disks?
  - How to move data between memory and disk?
  - Etc etc...

Data Integrity

- Manage concurrency and crashes
  - **Transaction**: A sequence of database actions enclosed within special tags
  - Properties:
    - **Atomicity**: Entire transaction or nothing
    - **Consistency**: Transaction, executed completely, take database from one consistent state to another
    - **Isolation**: Concurrent transactions appear to run in isolation
    - **Durability**: Effects of committed transactions are not lost
  - Consistency: Transaction programmer needs to guarantee that
    - DBMS can do a few things, e.g., enforce constraints on the data
  - Rest: DBMS guarantees
Data Integrity

- Semantic constraints
  - Typically specified at the logical level
  - E.g. balance > 0

DBMS at a glance

- Data Models
  - Conceptual representation of the data
- Data Retrieval
  - How to ask questions of the database
  - How to answer those questions
- Data Storage
  - How/where to store data, how to access it
- Data Integrity
  - Manage crashes, concurrency
  - Manage semantic inconsistencies

- Not fully disjoint categorization !!
Summary

- Why study databases?
  - Shift from computation to information
    - Always true in corporate domains
    - Increasing true for personal and scientific domains
  - Need has exploded in recent years
    - Data is growing at a very fast rate
  - Solving the data management problems is going to be a key

Summary

- Database Management Systems provide
  - Data abstraction
    - Key in evolving systems
  - Guarantees about data integrity
    - In presence of concurrent access, failures...
  - Speed !!