'Modern' Databases

Database Systems Lecture 18
Natasha Alechina

In This Lecture
- 'Modern' Databases
- Distributed DBs
- Web-based DBs
- Object Oriented DBs
- Semistructured Data and XML
- Multimedia DBs
- For more information
  - Connolly and Begg chapters 22-28
  - Ullman and Widom chapter 4

Other Sorts of DB
- We have looked mainly at relational databases
  - Relational model
  - SQL
  - Design techniques
  - Transactions
  - Many of these topics relied on relational concepts

Distributed Databases
- A distributed DB system consists of several sites
  - Sites are connected by a network
  - Each site can hold data and process it
  - It shouldn’t matter where the data is - the system is a single entity

- Distributed database management system (DDBMS)
  - A DBMS (or set of them) to control the databases
  - Communication software to handle interaction between sites

Client/Server Architecture
- The client/server architecture is a general model for systems where a service is provided by one system (the server) to another (the client)
  - Server
    - Hosts the DBMS and database
    - Stores the data
  - Client
    - User programs that use the database
    - Use the server for database access
Web-based Databases

- Database access over the internet
  - Web-based clients
  - Web server
  - Database server(s)
  - Web server serves pages to browsers (clients) and can access database(s)

- Typical operation
  - Client sends a request for a page to the web server
  - Web server sends SQL to database
  - The web server uses results to create page
  - The page is returned to the client

Web-based Databases

Advantages
- World-wide access
- Internet protocols (HTTP, SSL, etc) give uniform access and security
- Database structure is hidden from clients
- Uses a familiar interface

Disadvantages
- Security can be a problem if you are not careful
- Interface is less flexible using standard browsers
- Limited interactivity over slow connections

Object Oriented Databases

- Relational DBs
  - The database can't see data's internal structure so can't use complex data
  - Relational model gives a simple, and quite powerful, structure - but is quite rigid

- Object Oriented DBs
  - Use concepts from object oriented design/programming
  - OO concepts
    - Encapsulation
    - Inheritance
    - Polymorphism
  - OODBMS

Object Oriented Databases

- An object oriented database (OODB) is a collection of persistent objects
  - Objects - instances of a defined class
  - Persistent - object exist independently of any program

- An object oriented DBMS
  - Manages a collection of objects
  - Allows objects to be made persistent
  - Permits queries to be made of the objects
  - Does all the normal DBMS things as well

OODB example

- In lecture 10 we had a store with different sorts of products
  - Books
  - CDs
  - DVDs

- This lead to missing data among the various types

- OODB solution
  - We make an abstract Product class
  - Book, CD, and DVD are each a concrete subclass of Product
  - The database is a persistent collection of Products
**Object Oriented Databases**

- **Advantages**
  - Good integration with Java, C++, etc.
  - Can store complex information
  - Fast to recover whole objects
  - Has the advantages of the (familiar) object paradigm

- **Disadvantages**
  - There is no underlying theory to match the relational model
  - Can be more complex and less efficient
  - OODB queries tend to be procedural, unlike SQL.

**Object Relational Databases**

- **Extend a RDBMS with object concepts**
  - Data values can be objects of arbitrary complexity
  - These objects have inheritance etc.
  - You can query the objects as well as the tables

- **An object relational database**
  - Retains most of the structure of the relational model
  - Needs extensions to query languages (SQL or relational algebra)

**Semistructured data**

- **Semistructured Data**: A new data model designed to cope with problems of information integration.
- **XML**: A standard language for describing semistructured data schemas and representing data.

**The Information-Integration Problem**

- Related data exists in many places and could, in principle, work together.
- But different databases differ in:
  - Model (relational, object-oriented?).
  - Schema (normalised/ not normalised?).
  - Terminology: are consultants employees? Retirees? Subcontractors?
  - Conventions (meters versus feet?).
  - How do we model information residing in heterogeneous sources (if we cannot combine it all in a single new database)?

**Example**

- Suppose we are integrating information about bars in some town.
- Every bar has a database.
  - One may use a relational DBMS; another keeps the menu in an MS-Word document.
  - One stores the phones of distributors, another does not.
  - One distinguishes ales from other beers, another doesn’t.
  - One counts beer inventory by bottles, another by cases.
Semistructured Data

- Purpose: represent data from independent sources more flexibly than either relational or object-oriented models.
- Think of objects, but with the type of each object its own business, not that of its "class."
- Labels to indicate meaning of substructures.
- Data is self-describing: structural information is part of the data.

Graphs of Semistructured Data

- Nodes = objects.
- Labels on arcs (attributes, relationships).
- Atomic values at leaf nodes (nodes with no arcs out).
- Flexibility: no restriction on:
  - Labels out of a node.
  - Number of successors with a given label.

Example: Data Graph

- Nodes = objects.
- Labels on arcs (attributes, relationships).
- Atomic values at leaf nodes (nodes with no arcs out).
- Flexibility: no restriction on:
  - Labels out of a node.
  - Number of successors with a given label.

Example: Well-Formed XML

```xml
<?xml version = "1.0" standalone = "yes" ?>
<BARS>
  <BAR>
    <NAME>Joe's Bar</NAME>
    <BEER>
      <NAME>Bud</NAME>
      <PRICE>2.50</PRICE>
    </BEER>
    <BEER>
      <NAME>Miller</NAME>
      <PRICE>3.00</PRICE>
    </BEER>
  </BAR>
  ...
</BARS>
```

XML

- XML = Extensible Markup Language.
- While HTML uses tags for formatting (e.g., "italic"), XML uses tags for semantics (e.g., "this is an address").
- Key idea: create tag sets for a domain (e.g., bars), and translate all data into properly tagged XML documents.
- Well formed XML - XML which is syntactically correct; tags and their nesting totally arbitrary.
- Valid XML - XML which has DTD (document type definition); imposes some structure on the tags, but much more flexible than relational database schema.

XML and Semistructured Data

- Well-Formed XML with nested tags is exactly the same idea as trees of semistructured data.
- XML also enables non-tree structures (with references to IDs of nodes), as does the semistructured data model.
Example

- The `<BARS>` XML document is:

```
<NAME> Joe's Bar</NAME>
<BEER name="Bud" price="2.50"/>
<BEER name="Miller" price="3.00"/>
```

XPATH and XQUERY

- XPATH is a language for describing paths in XML documents.
  - Really think of the semistructured data graph and its paths.
  - Why do we need path description language: can’t get at the data using just Relation.Attribute expressions.
- XQUERY is a full query language for XML documents with power similar to OQL (Object Query Language, query language for object-oriented databases).

Multimedia Databases

- Multimedia DBs can store complex information
  - Images
  - Music and audio
  - Video and animation
  - Full texts of books
  - Web pages
- They can be used in a wide range of application areas
  - Entertainment
  - Marketing
  - Medical imaging
  - Digital publishing
  - Geographic Information Systems

Querying Multimedia DBs

- Metadata searches
  - Information about the multimedia data (metadata) is stored
  - This can be kept in a standard relational database and queried normally
  - Limited by the amount of metadata available
- Content searches
  - The multimedia data is searched directly
  - Potential for much more flexible search
  - Depends on the type of data being used
  - Often difficult to determine what the ‘correct’ results are

Metadata Searches

- Example - indexing films we might store
  - Title
  - Year
  - Genre(s)
  - Actor(s)
  - Director(s)
  - Producer(s)
- We can then search for things like
  - Films starring Kevin Spacey
  - Films directed by Peter Jackson
  - Dramas produced in 2000
  - We don’t actually search the films

Advantages

- Metadata can be structured in a traditional DBMS
- Metadata is generally concise and so efficient to store
- Metadata enriches the content

Disadvantages

- Metadata can’t always be found automatically, and so requires data entry
- It restricts the sorts of queries that can be made
Content Searches

• An alternative to metadata is to search the content directly
  • Multimedia is less structured than metadata
  • It is a richer source of information but harder to process

Example of content based retrieval

• Find images similar to a given sample
• Hum a tune and find out what it is
• Search for features, such as cuts or transitions in films

Content-Based Retrieval

Example of content based retrieval

• Find images similar to a given sample
• Hum a tune and find out what it is
• Search for features, such as cuts or transitions in films

Content-Based Retrieval

• Image retrieval is hard
  • It is often not clear when two images are ‘similar’
  • Image interpretation is unsolved and expensive
  • Different people expect different things

Do we look for?

• Images of roses
• Images of red things?
• Images of flowers?
• Images of red flowers?
• Images of red roses?

Other Topics

• Temporal DBs
  • Storing data that changes over time
  • Can ask about the history of the DB rather than just the current state
  • System time vs real time

• Logic DBs
  • A database is a set of facts and rules for manipulating them (like a Prolog program)
  • The DBMS maintains and controls these facts and rules
  • A ‘query’ is made by applying the rules to the facts

Next week

• Wednesday 12-1: Revision and module evaluation lecture
• If you have suggestions for revision questions, please email me!
• Monday the 28th 9-10: Java and SQL lecture (optional – not in the exam)