

COMP67321

Emerging DB technologies

Goran Nenadic

School of Computer Science
University of Manchester

1

Evolution of DB technologies

- Early technologies
 - hierarchical, network
- Current mainstream technologies
 - relational and distributed
- Other database technologies
 - deductive, functional, object-relational, object-oriented
- Emerging technologies
 - XML databases, multimedia databases
 - data mining, text mining, data warehousing

2

Current challenges and issues

- New data types
- New application areas
- Ultra-large databases
- Databases and the Web

3

New data types

- Traditional DB data
 - structured, formatted, limited data types
 - largely numeric, passive and static
 - no complex structures, abstract data types, inheritance etc.
- New data types
 - semi-structured and unstructured data
 - text and multimedia (video, audio, images)
 - temporal data
 - spatial data
- New representation, storage, indexing and query languages needed for manipulation

4

New application areas

- Traditional DBs
 - personnel DBs, accounts, supermarkets, etc.
 - on-line transaction processing (OLTP)
- New areas
 - engineering design
 - multimedia publishing and entertainment industry
 - national health systems
 - geographic information systems (GIS)
 - digital libraries
 - arts and scientific applications
 - biomedicine, genetics, physics, astronomy

5

Ultra-large databases

- In some areas, huge repositories
 - retail and financial transactions
 - digital libraries (e.g. biomedical literature)
 - health care
- Can we get some **knowledge** out of this data
 - historical overview
 - data mining: knowledge discovery in DBs
 - text mining: extracting knowledge from documents

6

Internet and database systems

- Database systems built upon Internet are distributed database systems
- Web data introduces a new challenges
 - access to heterogeneous, multimedia data
 - Semantic Web
 - amount of data is huge
 - quality, security and privacy issues

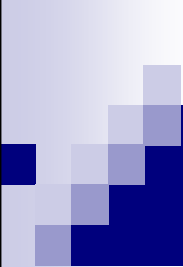
7

Emerging database technologies

- Database architectures
- Data warehousing and analysis
- Multimedia databases
- XML databases
- Web databases


8

9



Database architectures


10



Database system architectures

- Centralised
 - DBMS + user interface + applications on one machine
- Client/server
 - two-tier
 - three-tier
- Distributed and federated DBs

11



Client/server architecture: two-tier approach

- A database system is regarded as having a simple two-part structure, consisting of a **server** and a set of **clients**
- **Server** is the "core" DBMS
 - supports basic DBMS functions: data definition, data manipulation, data security and integrity etc.
 - query/transaction/SQL server
 - provides all the three levels of the ANSI/SPARC architecture
- **Clients** are various applications that run on top of DBMS
 - both user-written applications and built-in applications
 - user interface programs

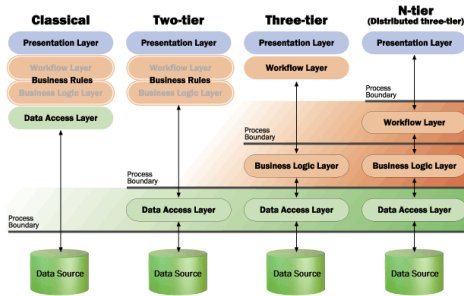
12

Client/server architecture: three-tier approach

- Adds an intermediate (middle) layer that
 - processes requests from clients and passes to server
 - **application** server
 - “describes” business rules
 - checks security credentials
- Levels:
 - user interface
 - application/business rules
 - data access

13

Variations of client/server architecture



14

Distributed database systems

- A collection of multiple logically interrelated DBs
 - organisational decentralisation
- Data and DBMS software are distributed over many sites, connected by a network
 - distributed DBMS
- “*Transparency*” for the clients:
 - one **logical** system
 - client can access many servers simultaneously as if there were only one server
 - client does not know which machines hold which pieces of data

15

continued

Distributed database systems

- Advantages
 - reflects organisational structure
 - local autonomy (maintenance of data)
 - improved availability
 - improved performance
 - modularity
 - economics of scale

16

continued

Distributed database systems

- Degree of homogeneity
 - homogeneous (uses one DBMS)
 - heterogeneous (multiple DBMSs, e.g. Oracle, MySQL)
- Level of autonomy varies
 - centralised DB (at one of the sites)
 - truly distributed
 - federated DBMS

17

Federated database systems


- Loosely coupled collection of (autonomous) DBs
 - can be within the same organisation
- Multi-database system
 - provides access to several independent DBs
 - each has local users, transactions, etc.
- Do not have a global common schema, but can provide a global common **view**
- Problems
 - differences in data models (relational, object)
 - differences in query languages, constraints, etc.
 - semantic heterogeneity (different names, representations, etc.)

18



Data warehousing and analysis


19



Data/information management

- Store all useful data and information
 - day-to-day operational data (e.g. transactions)
 - external data (e.g. market data)
 - human resources data
- Provide effective information storage and easy information access to support management at all levels
- Various types of data/information are needed by different management levels

20



What kind of data we have?

- Traditional “**transactional**” information, i.e. **operational** data that documents everyday life in an enterprise/organisation
 - retail (e.g. sales in supermarket stores)
 - financial services (e.g. ATM withdrawals)
 - transport (e.g. flight bookings)
 - telecommunications (e.g. mobile billing, Internet)
 - healthcare (e.g. drug prescriptions)
- Recording and processing this type of data is known as “online transaction processing”
 - operational/transactional databases

21

Online transaction processing

- OLTP: online transaction processing
- OLTP: processing and recording transactions that create *new* data and/or update existing information in operational DBs
 - insertions, updates, deletions
- Typically a small number of rows are affected in each transaction
- Traditional DBMS optimised to perform well in OLTP, but not in comprehensive exploration, aggregation and decision making

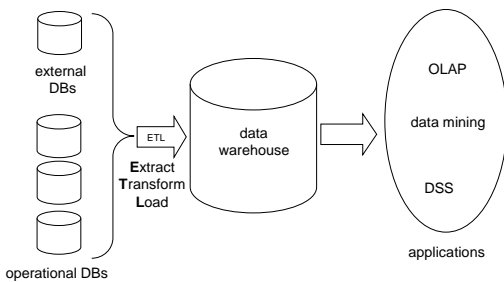
22

Data warehouse (DW)

- DW: an integrated database designed to support business intelligence, and better and faster decision making
- DWs *integrate* and *aggregate* data from various operational and external DBs maintained by different business units
- DW needs to provide
 - more complex aggregation of business data
 - mining “new” data (e.g. spending trends)

23

General DW schema



24

Data warehouse applications

- Online analytical processing (OLAP)
 - complex analysis of data from DW
 - e.g. trend analysis, time series, etc.
- Data mining (DM)
 - support for "knowledge discovery"
 - search for unanticipated knowledge
- Decision support systems (DSS)
 - high level data processing for management
 - executive information systems (EIS)

25

OLAP

- Interactive process of creating, managing, analysing and reporting on data
 - support for complex Boolean conditions, statistical functions and time-series analysis
- Main goal: support ad-hoc but complex querying performed by business analysts
 - data exploration & aggregation in various ways
- Allows a sophisticated user to analyse data using complex views
 - OLAP frameworks need to make this easy to do

26

example

Typical OLAP queries

- Write a multi-table join to compare sales for each product line year-to-date (YTD) this year vs. last year.
- Repeat the above process to find the top 5 product contributors to margin.
- Repeat the above process to find the sales of a product line to new vs. existing customers.
- Repeat the above process to find the customers that have had negative sales growth.

27

Data mining

- Process of extracting valid, previously unknown, comprehensible and actionable information from extremely large databases, in order to make crucial decisions or prediction
 - also known as “knowledge discovery in databases” (KDD)
- Process of identifying valid, non-trivial, novel, potentially useful **patterns** in data
 - use such patterns to predict or classify specific events, or act accordingly

Recall IBM lecture 28

Data mining – example methods

- **Association rule mining**
 - establish an “association” link, e.g. if a customer buys milk and tea, they also buy cookies
- **Classification**
 - mapping data into predefined classes (classes are known i.e. business determined)
- **Clustering**
 - partition data into groups of similar objects (groups i.e. clusters not know) – identify patterns

29

What is (not) data mining?

| | |
|---|---|
| <p>What is <u>not</u> data mining:</p> <ul style="list-style-type: none"> – Look up phone number in a phone directory – Query a Web search engine for information about “Amazon” – Find the average value of market basket during a given period – Rank web-pages based on access statistics | <p>What is data mining:</p> <ul style="list-style-type: none"> – Discover that certain names are more prevalent in certain locations – Group together similar documents returned by search engine according to their context (e.g. Amazon rainforest, Amazon.com) – Identify patterns in buying a PC and related equipment – Identify patterns in accessing a web site |
|---|---|

30

Data mining & OLAP

- Both are exploratory data analysis techniques: identify “interesting” trends and patterns
- OLAP
 - user-controlled and user-driven process
 - limited number of dimension and measure types
- Data mining
 - automatic results: patterns that a user may not be aware of; hidden knowledge
 - typically data-driven
 - mine information in any combination of dimensions

31

Decision support systems (DSS)

- Supports business or organizational decision-making activities.
- DSSs serve the management, operations, and planning levels of an organization and help to make decisions
- Typically an interactive system that gathers all important business data

32

Decision support systems (DSS)

- Typical information that a DSS application might gather and present are:
 - inventories of information assets (including legacy and relational data sources, data warehouses)
 - comparative sales figures between one period and the next,
 - projected revenue figures based on product sales assumptions

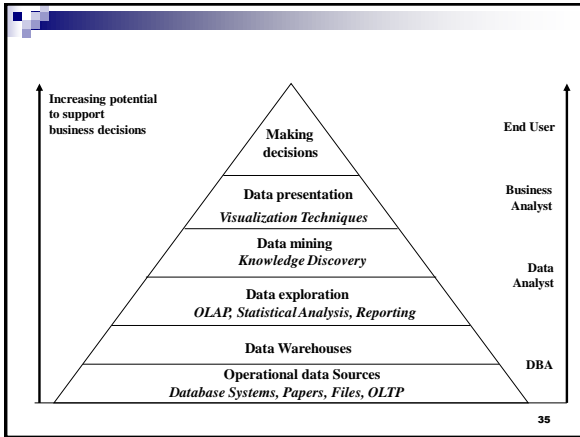
33

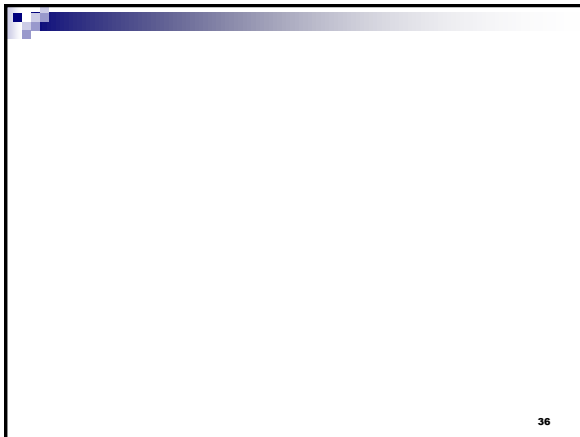
Chart Grid

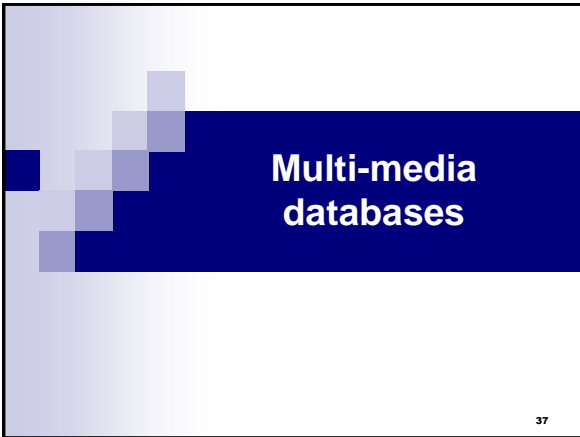
Internet Order Count and Customer Count by Customer by Date

| | | | Internet Order Count | Customer Count |
|--------------|------------|-----------------|----------------------|----------------|
| FY 2003 | H1 FY 2003 | New South Wales | 209 | 209 |
| | | Queensland | 116 | 116 |
| | | South Australia | 27 | 27 |
| | H2 FY 2003 | Tasmania | 11 | 11 |
| | | Victoria | 106 | 106 |
| | | Total | 469 | 469 |
| Total | | 469 | 469 | |
| FY 2003 | H1 FY 2003 | New South Wales | 277 | 277 |
| | | Queensland | 122 | 122 |
| | | South Australia | 63 | 63 |
| | H2 FY 2003 | Tasmania | 23 | 23 |
| | | Victoria | 161 | 161 |
| | | Total | 646 | 646 |
| Total | | 646 | 646 | |
| Total | | 1,115 | 1,115 | |
| Total | | 1,115 | 1,115 | |

34

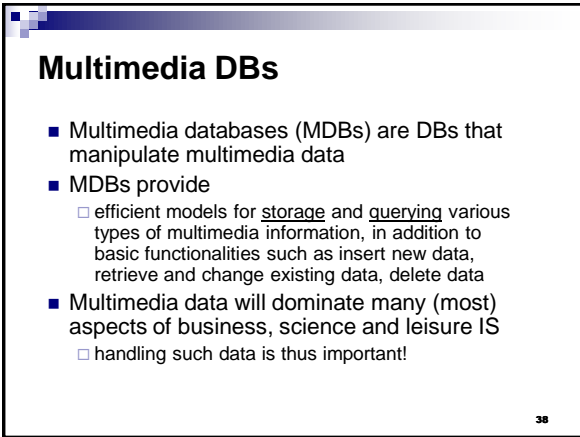






Multi-media databases

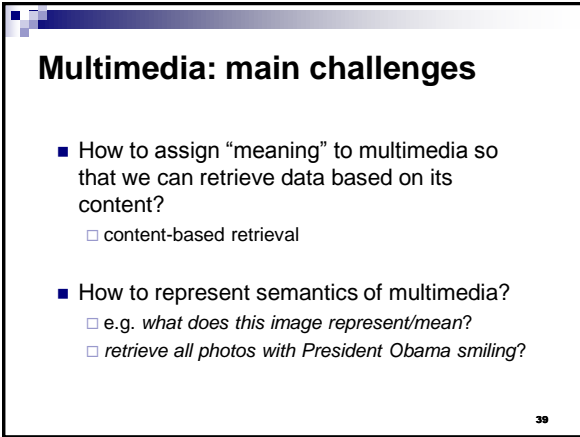
37



Multimedia DBs

- Multimedia databases (MDBs) are DBs that manipulate multimedia data
- MDBs provide
 - efficient models for storage and querying various types of multimedia information, in addition to basic functionalities such as insert new data, retrieve and change existing data, delete data
- Multimedia data will dominate many (most) aspects of business, science and leisure IS
 - handling such data is thus important!

38



Multimedia: main challenges

- How to assign “meaning” to multimedia so that we can retrieve data based on its content?
 - content-based retrieval
- How to represent semantics of multimedia?
 - e.g. *what does this image represent/mean?*
 - *retrieve all photos with President Obama smiling?*

39



Multimedia: main challenges

- Two main problems
 - information is mainly implicit
 - interpretation can be subjective: an object means different things to different people
- How to extract explicit features so that we can do querying, integration and analysis
 - meta-data

41

Meta-data: data about data

- In traditional relational DBMS, metadata describes "simple" features assigned to relations/tables
 - schema definition, meaning of attributes, attribute ranges, integrity, primary/foreign keys, ...
- In multimedia DBMS, meta-data describes semantics of relations and individual objects
 - meta-data captures semantics and makes it as explicit as possible
 - how to generate meta-data?

42

continued

Meta-data: data about data

- Use of metadata is essential for querying and integration of multimedia databases
 - query multimedia databases by querying meta-data
- Generate meta-data when storing an object
 - this is one of the main issues in multimedia DBs
- Requirements for generating meta-data
 - apply a model known/used by expected users
 - support "near" matches
 - cater for interpretation and subjectivity (both of those who provide and those who use the meta-data)

43

continued

Meta-data: data about data

- *Subjectivity*: different interpretations of the content both
 - when providing meta-data and
 - when querying
- Model and content of meta-data depends on the application area
- Two types of meta-data
 - [textual descriptions](#)
 - [annotations](#)

44

continued

Meta-data: data about data

- [Textual descriptions](#): short summaries of a given multimedia object
- [Annotations or tagging](#): descriptors (i.e. categories or features or attributes) assigned from a set of values
 - open-ended keywords
 - controlled vocabularies (pre-defined, no structure)
 - taxonomies and ontologies (pre-defined, structured vocabularies)

45

continued

Storage models - examples

- BLOB: binary large object
 - a collection of binary data stored as a single entity
 - for images, videos, and sound, programs (2-4GB)
- CLOB: character large object
 - for storing text data; e.g. TEXT (mysql)
- Storing objects externally (as files, URL, etc.)
 - data types: DATALINK, BFILE
 - typically read-only access
 - standard data formats for multi-media

49

XML databases

50

XML

- XML = eXtensible Mark-up Language
- Entity = Data + explicit Mark-up
 - data entities can be strings, images, video, pubs, trains, visits, etc.
 - mark-up = assign annotation to data
 - explicit "structure", "semantics", "meaning"
- XML 1.0
 - recommendation from W3C, 1998

51

HTML vs. XML

- HTML uses tags for formatting (e.g., "italic")
 - describes the layout
- XML uses tags for structure and semantics (e.g., "this is an address, and it contains house number, street, and postcode")

52

XML example (1)

```

<addrBook>
  <person NI = "111-22-3333">
    <name> Caesar </name>
    <greet> Caesar Imperator </greet>
    <addr> The Capitol </addr>
    <addr> Rome, OH 98765 </addr>
    <tel> (321) 786 2543 </tel>
    <fax> (321) 786 2543 </fax>
    <email> jc@forum.rome.org </email>
  </person>
</addrBook>

```

53

XML: main concepts

- XML document
- Schema descriptions
 - defined by
 - DTD = Document Type Definition
 - XSD = XML Schema Definition
- Elements, including their
 - structure, and
 - attributes (describe elements)

54

XML terminology

- tags: **book, title, author, ...**
- start tag: **<book>**, end tag: **</book>**
- elements: **<book>...</book>**, **<author>...</author>**
- **elements can be nested**
- empty element: **<red></red>** or **<red/>**
- attributes: **add additional features to data**
 <book year="1992">...</book>
 alternative: <book><year>1992</year>...</book>

55

XML example (2)

```
<book year="1992">
  <title>Advanced Programming in the Unix environment</title>
  <author><last>Stevens</last><first>W.</first></author>
  <publisher>Addison-Wesley</publisher>
  <price>65.95</price>
</book>

<book year="2000">
  <title>Data on the Web</title>
  <author><last>Abiteboul</last><first>Serge</first></author>
  <author><last>Buneman</last><first>Peter</first></author>
  <author><last>Suciu</last><first>Dan</first></author>
  <publisher>Morgan Kaufmann Publishers</publisher>
  <price>39.95</price>
</book>
```

56

Document type definition (DTD)

- Defines XML tags and their nesting
- Each domain of interest (e.g., electronic components, bars-beers-drinkers) creates a DTD that describes all the documents (data) this group will share
- An XML document may have a DTD
 - not obligatory
 - but, validation is useful in data exchange

57

example

A simple DTD

```

<!DOCTYPE company [
  <!ELEMENT company ((person|product)*)>
  <!ELEMENT person (NI, name, office, phone?)>
  <!ELEMENT NI      (#PCDATA)>
  <!ELEMENT name    (#PCDATA)>
  <!ELEMENT office  (#PCDATA)>
  <!ELEMENT phone   (#PCDATA)>
  <!ELEMENT product (pid, name, description?)>
  <!ELEMENT pid     (#PCDATA)>
  <!ELEMENT description (#PCDATA)>
]>
    
```

58

XML schema definition (XSD)

- DTDs define structure, but do not provide any complex type constraints
- XSD = XML schema definition
- XML schema is a W3C standard for data modelling using XML
- XSD can specify
 - which elements/attributes are mandatory/optional
 - element/attribute types
 - cardinalities
 - relative ordering

59

XSD example

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="Employee"
    minOccurs="0"
    maxOccurs="unbounded">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="NI" type="xsd:string"/>
        <xsd:element name="Name" type="xsd:string"/>
        <xsd:element name="DateOfBirth" type="xsd:date"/>
        <xsd:element name="EmployeeType" type="xsd:string"/>
        <xsd:element name="Salary" type="xsd:long"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
    
```

60

XML databases

Why do we need XML?

61

Semi-structured data

- Semi-structured data
 - no strict format, but can have some structure with optional elements and attributes
 - some attributes may be missing, some repeated, and new ones can be added later: change unpredictably
 - therefore, need explicit information about elements
 - ad-hoc, diverse data sources
 - examples
 - HTML document with titles (e.g. <H1>) and paragraphs (<p>), but no information about author(s), date, etc.
 - multimedia objects

62

example

Semi-structured data

- Missing attributes


```

<person> <name> John</name>
          <phone>1234</phone>
</person>

<person> <name>Joe</name>
</person>
                
```

no phone !
- Could be represented in a relational table with nulls (-)

| name | phone |
|------|-------|
| John | 1234 |
| Joe | - |

63

example

Semi-structured data

- Repeated attributes


```
<person> <name> Mary</name>
           <phone>2345</phone>
           <phone>3456</phone>
</person>
```

two phones !
- Difficult in a table

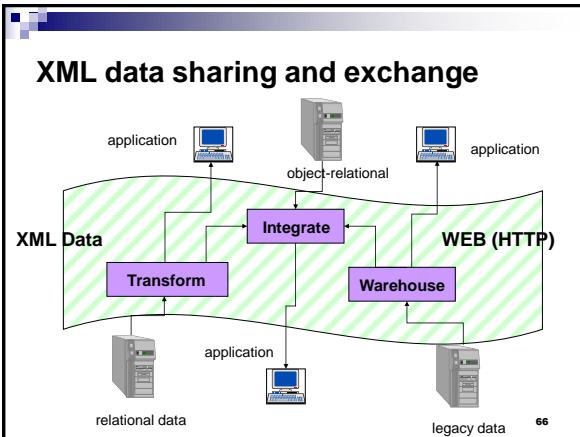
| name | phone | | ??? |
|------|-------|------|-----|
| Mary | 2345 | 3456 | ??? |
| | | | |

64

Data exchange and integration

- Relational data does not have a "syntax"
 - I can't "give" you my relational database (if we don't use the same DBMS and share the schema)
 - need to import it from other syntax, like CSV (comma-separated-values; or SQL commands)
- We need a flexible model that supports rich syntax for complex data, and
 - can map any (existing) data into it
 - can store data in files, so exchange on the Web, etc.
 - query it directly

65



XML databases

Data modelling with XML

67

example

Semi-structured data

- Relational schema and data are separated
 - relational schema: `persons(name, phone)`
- Typically, semi-structured data is self-describing
 - mixing data and meta-data (schema)
- Schema elements become part of the data
 - in XML, `<person>`, `<name>`, `<phone>` are part of the data, and are repeated for each entity (as required)
- Semi-structured data can be modeled by trees and/or (directed) graphs

68

Example: tree/graph model

69

Data modeling with XML

- XML is **self-describing**
 - consequence: XML is much more flexible
- Well-Formed XML with nested tags is exactly the same idea as trees of semi-structured data
- XML also enables non-tree structures, as does the semi-structured data model

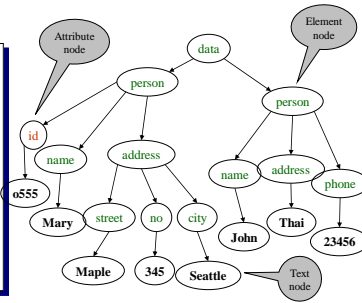
XML is a natural choice to model semi-structured data

70

XML data: tree representation

```

<data>
  <person id="0555" >
    <name> Mary </name>
    <address>
      <street> Maple </street>
      <no> 345 </no>
      <city> Seattle </city>
    </address>
  </person>
  <person>
    <name> John </name>
    <address> Thailand </address>
    <phone> 23456 </phone>
  </person>
</data>
    
```



71

example

From relational to XML data

employees

| name | phone |
|------|-------|
| John | 3634 |
| Sue | 6343 |
| Dick | 6363 |

```

<employees>
  <row> <name>John</name>
    <phone>3634</phone></row>
  <row> <name>Sue</name>
    <phone>6343</phone>
  <row> <name>Dick</name>
    <phone>6363</phone></row>
</employees>
    
```

72

XML databases

Storing and querying XML data

73

Storing XML data

- XML-data can be stored in the following 3 types of DBMS
 - **traditional relational** database
 - DBMS basically does not 'deal' with XML
 - storage
 - without mapping (XML documents as CLOBs)
 - map an XML schema to relational schema
 - **XML-enabled** database
 - **XML-native** database

74

XML-enabled databases

- Use object or object-relational model to map XML-schema
 - model XML-elements as classes
 - e.g. element *Person* is modelled as a class *Person*
- Also, new data types provided to support XML
 - SQL:2003 (SQL/XML) introduced XML extensions including a new native data type (*XML*) and a set of operators
 - DBMS now understands how to deal with XML
 - XMLCLOB, XMLFile (DB2); XMLTYPE (Oracle 9i)

75

example

XML-enabled databases

```
CREATE TABLE mail_user (
  user_name  VARCHAR2(20),
  mailbox    SYS.XMLTYPE
);
```

```
<mailbox>
<inbox>
<email id=1><subject text="Urgent info"><from .../><attach .../> </email>
<email id=2><subject text="Meeting"/><from .../><attach .../></email>
...
</inbox>
<outbox>
<email id=7><subject text="RE:Urgent"/><from .../><attach .../></email>
...
</outbox>
</mailbox>
```

76

example

XML-enabled databases

Querying using SQL-like expressions

return names of users that have any email in their inbox that contains string "Manchester" in its subject

```
SELECT   user_name
FROM     mail_user
WHERE    mailbox.extract('/mailbox/inbox/email/subject/text()').getStringVal()
LIKE '%Manchester%'
```

↑ Elements
(full path)
↑ Attribute

77

XML-native databases

- Designed to store XML documents
 - do not rely on a relational or object model
 - basic unit of logical design is an XML-document
 - easy/natural storage and access to data
- Support for XML query languages
 - XPath = simple path through the tree
 - XQuery = the SQL of XML
- Example XML-native DBMS
 - Tamino, X-Hive (commercial)
 - dbXML, eXist (open-source)

78

XML query languages: XPath and XQuery

79

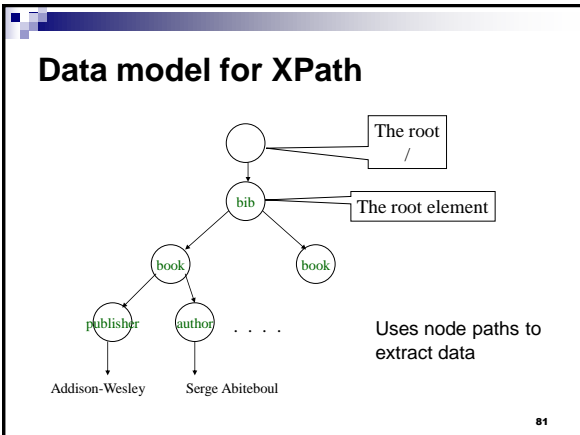
file: bib.xml

Sample data for queries

```

<bib>
  <book> <publisher> Addison-Wesley </publisher>
  <author> Serge Abiteboul </author>
  <author> <first-name> Rick </first-name>
  <author> <last-name> Hull </last-name>
  </author>
  <author> Victor Vianu </author>
  <title> Foundations of Databases </title>
  <year> 1995 </year>
</book>
<book price="55">
  <publisher> Freeman </publisher>
  <author> Jeffrey D. Ullman </author>
  <title> Principles of Database and Knowledge Base Systems
  </title>
  <year> 1998 </year>
</book>
</bib>

```



XPath: simple expressions

- retrieve *titles* of all books from file "bib.xml"


```
doc("bib.xml")/bib/book/title
```

result

```
<title>Foundations of Databases</title>
<title>Principles of Database and Knowledge Base Systems</title>
```
- ```
/bib/book/year
```

result

```
<year> 1995 </year>
<year> 1998 </year>
```

82

---

---

---

---

---

---

---

---

---

---

examples

### XPath: predicates

- Specify restrictions/conditions/constraints
  - retrieve all books published before 2000
 

```
doc("bib.xml")/bib/book[year<2000]
```
  - retrieve *titles* of all books published before 2000
 

```
doc("bib.xml")/bib/book[year<2000]/title
```

83

---

---

---

---

---

---

---

---

---

---

### Examples

|                 |                                         |
|-----------------|-----------------------------------------|
| bib             | matches a bib element                   |
| @price          | matches a price attribute               |
| /               | matches the root element                |
| /bib            | matches a bib element under root        |
| bib/paper       | matches a paper in bib                  |
| bib/book/@price | matches price attribute in book, in bib |

Also,

|            |                           |
|------------|---------------------------|
| paper book | matches a paper or a book |
| *          | matches any element       |

84

---

---

---

---

---

---

---

---

---

---

## XQuery: motivation

- XPath's expressivity insufficient
  - no join queries (as in SQL)
  - no quantifiers (as in SQL)
  - no aggregation and functions
  - no changes to the XML structure possible
- XQuery uses XPath to express more complex queries

85

---

---

---

---

---

---

---

---

## XQuery: basic structure

```
FOR ...
LET...
WHERE...
ORDER BY...
RETURN...
```

86

---

---

---

---

---

---

---

---

## XQuery: example

```
FOR $x IN document("bib.xml")/bib/book
WHERE $x/year/text() > 1995
RETURN $x/title
```

Retrieve all book titles published after 1995

87

---

---

---

---

---

---

---

---

## XQuery: example

```
FOR $x IN document("bib.xml")/bib/book[year/text()>1995]/title
RETURN $x
```

Retrieve all book titles published after 1995

88

---

---

---

---

---

---

---

---

---

---

## XQuery: joins and nesting

For each author of a book by *Addison-Wesley*, list all books they published:

```
FOR $b IN document("bib.xml")/bib,
 $a IN $b/book[publisher/text()='Addison-Wesley']/author
RETURN { $a,
 FOR $t IN $b/book[author/text()=$a/text()]/title
 RETURN $t
 }
```

In the RETURN clause, comma concatenates XML fragments

89

---

---

---

---

---

---

---

---

---

---

## XQuery: aggregates

**count** = a function that counts  
**avg** = computes the average  
**sum** = computes the sum  
**distinct-values** = eliminates duplicates

Find all books with more than 3 authors

```
FOR $x IN document("bib.xml")/bib/book
WHERE count($x/author)>3
RETURN $x
```

```
FOR $x IN document("bib.xml")/bib/book[count(author)>3]
RETURN $x
```

90

---

---

---

---

---

---

---

---

---

---

**Summary - use XML for**

- Representation, storage and content-based retrieval of semi-structured data
  - e.g. storing and querying multimedia, text, Semantic Web
- Data exchange
  - transform relational/object models into XML models
  - however, unlike relational/object databases, an XML database need not have a schema!
  - an XML schema may not be very prescriptive in terms of what can or cannot be stored

91

---

---

---

---

---

---

---

---

**Web databases**

92

---

---

---

---

---

---

---

---

**Web as complex multimedia DB**

- Heterogeneous data on a single page
  - text + images + graphics + animations + videos
- Usage/user models are different
  - not used by specialists/professionals
  - used for many purposes
- Scalability issues
  - search space includes billions of objects
- Web search engines
  - search by text features only (IR engines)
  - software agents (Web crawlers)

93

---

---

---

---

---

---

---

---



From Google CEO Eric Schmidt

## Web & data mining and privacy

- If I look at enough of your messaging and your location, and use Artificial Intelligence, we can predict where you are going to go. Show us 14 photos of yourself and we can identify who you are. You think you don't have 14 photos of yourself on the internet? You've got Facebook photos! People will find it's very useful to have devices that remember what you want to do, because you forgot...But society isn't ready for questions that will be raised as result of user-generated content."

<http://www.itworld.com/internet/116426/google-brace-yourselves-data-explosion>

97

---

---

---

---

---

---

---

---

## Summary

- Current challenges
  - new data types
  - new application areas
  - ultra-large databases
- Various database architectures
  - Client-server, distributed, federated
- Various databases
  - Operational/transactional DBs
  - DW: an integrated database for supporting higher-level analysis of data

98

---

---

---

---

---

---

---

---

continued

## Summary

- Data analysis: making sense from large datasets
  - OLAP: ad-hoc exploration
  - Data mining: systematic exploration
  - Decision support systems
- Managing multi-media data
  - implicit and subjective semantics
- XML data model
  - semi-structured data: explicit instance-level tags
- Web as multi-media database
  - large scale
  - data linking and privacy

99

---

---

---

---

---

---

---

---