

Information Management Resource Kit

Module on Management of Electronic Documents

UNIT 5. DATABASE MANAGEMENT SYSTEMS

LESSON 4. TEXTUAL, RELATIONAL AND XML DATABASES

NOTE

Please note that this PDF version does not have the interactive features offered through the IMARK courseware such as exercises with feedback, pop-ups, animations etc.

We recommend that you take the lesson using the interactive courseware environment, and use the PDF version for printing the lesson and to use as a reference after you have completed the course.



© FAO, 2003

Objectives

At the end of this lesson, you will be able to:

- understand the differences between **relational and textual databases**, and
- understand how **XML** can be used in a database system.



Introduction

Textual or relational database: which choice will better meet our needs?



Once you have defined your requirements for document management and delivery, you have to choose the type of database that can meet your needs.

To make the right choice, it is useful to understand the basic principles and benefits provided by the two main types of databases: textual and relational.

Flat file databases

The **flat file database** can be considered the **first basic type of database**.

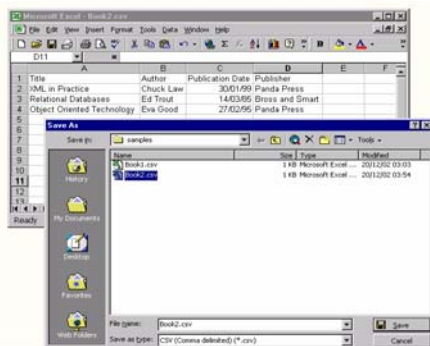
A flat file database is a textual file that can be created using a simple text editor.

Each information field (e.g. title, author, publisher, etc.) is separated from others using a delimiter character (usually a comma) and each record is separated from others using another character or by pressing the ENTER key.

```
XML in Practice,Chuck Law,30/01/99,Panda Press,345
Relational Databases,Ed Trout,14/03/85,Bross and Smart,267
Object Oriented Technology,Eva Good,27/02/95,Panda Press,456
```

If you use a comma as the separator, this is called a **CSV** file (Comma Separated Values).

Flat file databases



You can also easily create a CSV file **using a spreadsheet**. In fact, most spreadsheet packages and some relational database products give you the option to 'Save As .csv'.

In this example we used Microsoft Excel.

It is very easy to write your own code to read, write, delete and update records in a flat file database, or you can use open source code written by other people; one of the most widespread flat file databases is called **DBM**.

Instead of using flat files with field separators and tools such as DBM, **we could use XML** to represent the fields in our database and use open source XML parsers and processors to access them.

DBM has open source implementations available in many languages. Most Unix and Linux operating systems ship with a set of DBM tools. You can get an implementation called GDBM from the Gnu Project (www.gnu.org) or a Perl implementation called SDBM from www.perl.org.

Flat file databases

Flat file databases work fine for **simple data structures**, but problems start for example when...



Mmmh...this book was written by three authors: I have to store the three of them in the same field...

Ouch! The publisher Panda Press was taken over by Bross and Smart: I have to change its name in all the fields!

A field must contain more than one item of information. This means that all fields are not homogeneous (e.g. the content in the field "author" can be a single author or a list of authors).

The same information is repeated in the database. This means we have **redundant** data storage and this can cause problems with **consistency** when we want make changes to data: apart from the additional effort involved, there would be a risk that we might miss out one of the changes and make our data inaccurate.

Flat file databases

```
XML in Practice,Chuck Law,30/01/99,Panda Press,345
Relational Databases,Ed Trout,14/03/85,Bross and Smart,267
Object Oriented Technology,Eva Good,27/02/95,Panda Press,456
```

For example, in this database...

- some fields contain more information than others.
- some information is redundant.

Please click on the answer of your choice

Relational databases

With a relational database these problems are solved.

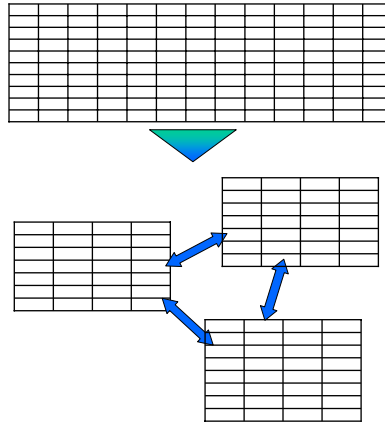
A relational database is a **database which uses the relational data model for storing data.**

The basic idea is simple: instead of creating a single logical unit which contains the entire database, the database is **split into several tables.**

Each table contains a set of records with logically structured data.

Relationships between the data in different records are used to join the tables together to form a single logical database.

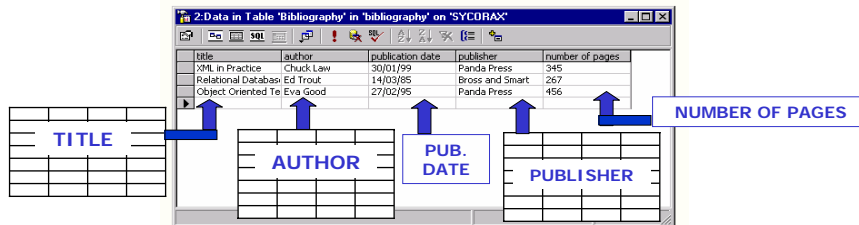
Let's look at an example...



Relational databases

To store **bibliographic information** in our library we could create a Bibliography table with five columns (**fields**): title, author, publication date, publisher, number of pages.

Each row corresponds to a specific book (**record**). Here's what the table looks like when we create it in Microsoft SQL Server and load up three records:



The fields in the 'publication date' column are all of type 'Date' and the fields in the 'number of pages' column are all integers.

The other fields could be transformed into as many **separate tables**. Let's see how...

Relational databases

title	author	publication date	publisher	number of pages
XML in Practice	Chuck Law	30/01/99	Panda Press	345
Relational Databases	Ed Trout	14/03/85	Bross and Smart	267
Object Oriented Te	Eva Good	27/02/95	Panda Press	456

PUBLISHER	
1	Panda Press
2	Bross and Smart
3
...
...
...
n

For example, we can make a **separate table** called 'Publishers' that contains the names of all the publishers and **then refer to records in that table** from fields in the bibliography.
 In that way we **only have one record** for Panda Press, which is used by reference everywhere else that we need it.

Relational databases

Publishers	
* (All Columns)	
pubId	publisher
1	Panda Press
2	Bross and Smart
*	

To make the reference without ambiguity you need to be able to **uniquely identify each record** in the Publishers table.
 To do that we define a **primary key** in the Publishers table: this is a one or more columns which uniquely identify a record in the table.
 Sometimes it is necessary to create a column with an id value: for example, **pubId**.

Relational databases

Now we can change our **Bibliography** table so that each record has a primary key and the 'publisher' column no longer holds the name of the publisher, but the **pubId** of a publisher in the new Publishers table.

bibid	title	author	publication date	publisherKey	number of pages
1	XML in Practice	Chuck Law	30/01/99	1	345
2	Relational Databases	Ed Trout	14/03/05	2	267
3	Object Oriented Technology	Eva Good	27/02/95	1	456

In the Bibliography table the publisher is now something called a **foreign key**: it takes the value of a primary key in another table and is used to make reference to records in that other table.

To indicate this change we will change the name of the publisher column to **publisherKey**.

Relational databases

Now we are sure that there is no data redundancy, but we don't have the direct relationship between a book and its publisher expressed in the record in a single table; it is encapsulated in the reference between the two tables.

```

SELECT Bibliography.title, Bibliography.author, Publishers.publisher
FROM Publishers INNER JOIN
      Bibliography ON Publishers.pubId = Bibliography.publisherKey
  
```

title	author	publisher
XML in Practice	Chuck Law	Panda Press
Relational Databases	Ed Trout	Bross and Smart
Object Oriented Technology	Eva Good	Panda Press

If we want to get the relationship back directly in a single record we need to **join the two tables** back together again (using a query expressed in the relational database query language SQL).

Note. Access SQL is used in this example. It would not necessarily work on other databases.

Relational databases

Panda Press was taken over by Bross and Smart: no problem, I can update the database without changing every occurrence in the bibliography table!



One of the benefits of the relational data model is that it allows you to create a normalized data model, where **no data are repeated**.

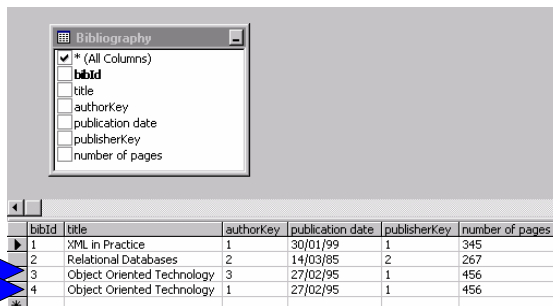
What we have created is a **one-to-many relationship** between a publisher and books, that is to say one publisher may publish many books.

We could do the same with authors.

So far our bibliography has a single author for each publication, but what if we now want to allow publications with more than one author?

Relational databases

We want to allow any author to write many books and any book to be written by many authors. This is called a **many-to-many relationship** between authors and books.



bibid	title	authorKey	publication date	publisherKey	number of pages
1	XML in Practice	1	30/01/99	1	345
2	Relational Databases	2	14/03/85	2	267
3	Object Oriented Technology	3	27/02/95	1	456
4	Object Oriented Technology	1	27/02/95	1	456

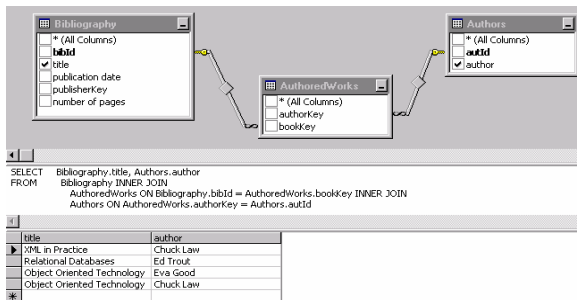
So far, the only way we can allow a book to have more than one author, using the Bibliography and Authors tables that we have, is to **repeat rows for each publication** with a different author in each row.

So here we have repeated the row for 'Object Oriented Technology' so that it can reference both Eva Good and Chuck Law as authors.

Once again we have a redundancy problem!

Relational databases

In fact, although we are only talking about two entities (e.g. authors and books) we can't model the many-to-many relationship between them properly in a relational database unless we **introduce a third table**.



We call this table **AuthoredWorks**: it will hold foreign keys to records in the Bibliography and Authors tables.

We can now get a list of publication titles and their authors by executing an SQL query that joins the Bibliography and Authors tables as shown in the figure.

Note. Access SQL is used in this example. It would not necessarily work on other databases.

Relational databases

Relational databases are often used as the basis for document or content management systems, which provide several benefits for the management and delivery of information.

Features of Document Management systems

Document management features	Access and retrieval features
<ul style="list-style-type: none"> - Import/Export - Check in/Check out - Access control - Version control - Variant management - Workflow (process management) - Back up/Restore/Logging - Metadata management - Support for cross references and link management - Integration with editing and processing tools - Document configuration 	<ul style="list-style-type: none"> - Full text index and search - Metadata index and search - XML (or HTML) structural search - Paging or search results - Sorting/filtering or search results - Format transformation - User profiling and preferences - Customised views and configurations by user or role

On the other hand, you do not always need all these features; it depends on your requirements.

Textual databases

We need a database which links to the full text of each document stored.



Let's have a look at this example.

We have to choose a database for a **simple bibliographic reference database**.

The main requirements for our system are:

- quick **search of the full text** of the documents,
- **metadata search**,
- controlled **update** of the document collection (infrequently), and
- **browsing** of the document collection, based on metadata.

Textual databases

In our example, which are the main features needed in the database?

- Integration with editing and processing tools.
- Metadata index and search.
- Full text index and search.
- Version control.

Please click on the answers of your choice

Textual databases

The type of metadata we want to hold for each document is shown in this XML fragment, which uses the metadata standard called RDF:

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">

  <rdf:Description>
    <dc:creator>Chuck Law</dc:creator>
    <dc:publisher>Panda Press</dc:publisher>
    <dc:description>A basic introduction to XML with
hands-on exercises</dc:description>
    <dc:identifier>ISBN-129-12992</dc:identifier>
    <dc:format>PDF</dc:format>
    <dc:title>XML in Practice</dc:title>
    <dc:date>30/01/99</dc:date>
    <dc:language>EN</dc:language>
  </rdf:Description>

</rdf:RDF>
```

To satisfy our need **we can use a textual database.**

If we already had a relational database installed and some programming resources available, then it would be possible to implement a system to meet our requirements.

We could also meet the requirements using a document management system based on a relational database, although the system would include many features that we don't require in this instance.

Textual databases



Textual Databases (often called Textbases) store collections of documents or texts as a series of records and fields. The data structures in a textbase are fixed and hold the text, metadata and indexes.

Textual databases are designed to manage **text-based resources**, providing users with fast search and retrieval and some control over the assembly and formatting of text components.

The **defining features** of a textual database are:

- Management of text as **discrete records**
- **Indexing** of text in the records
- Fast **search and retrieval** functionality
- **Sorting and assembly** of document records
- **Packaging, transformation or formatting** of text documents

Textual databases

Most textual databases break a collection of documents down into a **sequence of records** (some also allow a sub-division of records into fields).

	MFN	Author(s)	Title	...
Record 1	1	Salih, A.G.
Record 2	2	
.....	...			

The **structures supported** by text databases are limited (to these simple linear collections) and are **generally fixed** in the database.

Textual databases may also provide **text management tools**.

Textual databases may also provide **text management tools** that include:

- Hyperlink management
- Document assemble (putting together compound documents from a set of smaller components)
- Document comparison
- Hyperlink verification
- Support for multiple languages
- Metadata search
- Thesaurus search and integration
- Annotation (so that individual users can associate their own notes with text records or fields)

Textual databases



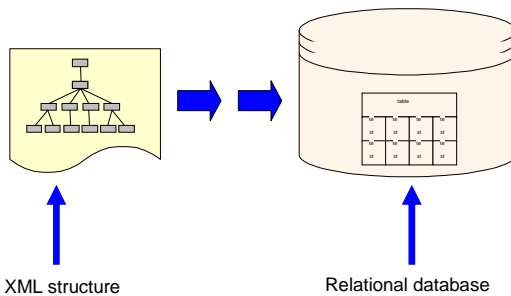
The most powerful features of text bases are centered around their ability to **rapidly search large document collections** and **retrieve records** which are then displayed to the user (if necessary with some formatting added).

Commercial textbase products have included BASISplus, BRS/SEARCH and Folio VIEWS. CDS/ISIS is a text database maintained by the UNESCO General Information Programme.

In recent years **relational databases** such as Oracle and SQL Server **have added the capability of full text index and search** and this, combined with the **emergence of XML** as a standard for structured text, has led to something of a decline of more specialist textbase products.

XML and databases

Recently there has been much technical work on the linkage of **XML and databases**.

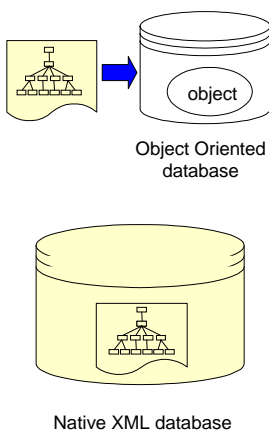


Relational databases can be used as **XML database**, but the relational model of tables is **not naturally suited** to modelling the hierarchical structures of XML documents.

So one or more **layers of transformation** between the XML data structures and the structures stored persistently on disk are needed.

With the rise in adoption of XML, the leading relational database vendors have moved to **add XML support** into their products.

XML and databases



During the late 1980s and early 1990s this problem was addressed by a new breed of products: **object-oriented databases**. Because the model used by object-oriented databases is very similar to the hierarchical model of XML documents, these databases have often been used to implement XML databases.

It turns out that the object caching mechanisms of some object-oriented databases don't work very well with XML structures (particularly when fine-grained reuse of XML elements is implemented).

So by the time XML became firmly established as a technology, the race was on to develop a new generation of **native XML databases**, which lay down XML structures directly to persistent disk storage.

Some commercial products include **Tamino** (www.softwareag.com), **X-Hive** (www.x-hive.com) and **TextML** (www.ixiasoft.com). There are also some good open source implementations, most notably **Exist** (exist.sourceforge.net).

The native XML data type added by Oracle at Oracle9i also turns that database into the equivalent of a native XML database. With XML support in all the leading relational database products there is a danger that the specialist native XML databases vendors will suffer the same fate as the object-oriented vendors of the 1990s. This is a factor that should be considered when buying products.

Summary

- The **flat file database** is the **first basic type of database**; it can be a textual file created using a simple text editor.
- A **relational database** is a database which uses the relational data model for storing structured data.
- Relational databases support a normalized data model, where **there is no redundant information storage**.
- If you only need to manage **text-based resources**, providing users with fast search and retrieval and some control over the assembly and formatting of text components, you can use a **textual database**.
- Different types of **XML databases** can be implemented using relational, object-oriented or native XML databases.



Exercises

The following five exercises will allow you to test your understanding of the concepts described up to now.

Good luck!



Exercise 1

Which of the following files is a flat file database?

- A simple text file which doesn't contain record and fields.
- An XML file which represents records and fields.
- A simple text file using field and record separators.

Please click on the answer of your choice

Exercise 2

	bibId	title	publication_date	publisherKey	number_of_pages
1	1	XML in Practice	30/01/99	1	345
2	2	Relational Databases	14/03/85	2	267
3	3	Object Oriented Technology	27/02/95	1	456
*					

Which is the primary key in this table?

- bibId
- publication_date
- publisherKey

Please click on the answer of your choice

Exercise 3



Imagine that you need to manage the documentation at each phase of a project (design, development and implementation), with particular requirements to:

- make documents available in read-only mode to all project participants;
- allow document owners to create and update documents;
- manage the versions of all documents;
- link documents with project-related information and metadata.

In your opinion, the main requirements are for...

- indexing documents and provide an indexed search to help users find them.
- managing the production and access to documents.

Please click on the answer of your choice

Exercise 4



Your main requirements are managing the production and access to documents.

You also need to manage versions very carefully, since it is likely that there will be very frequent updates to the documents.

Which of the following types of database can meet your needs?

- A relational database.
- A textual database.

Please click on the answer of your choice

Exercise 5

Could you associate each type of database with the relevant feature?

Relational database

It uses a very similar model to that of XML documents

Object-oriented database

It needs an XML support to manage XML documents.

Native XML database

It is ideal to implement fine-grained reuse of XML elements.

Click on each option and drag it to the correspondent box.

If you want to know more...

DBM Flat File database. You can get an implementation called GDBM from the Gnu Project (www.gnu.org) or a Perl implementation called SDBM from www.perl.org.

Date, C.J. An Introduction to Database Systems, UK, Addison Wesley; ISBN: 0201787229.

www.B2Business.net - an online portal with information on products for electronic business, including listings of document and content management systems.

CDS/ISIS is a text database maintained by the UNESCO General Information Programme: <http://www.unesco.org/isis>

Resource Description Framework (RDF) Model and Syntax Specification. Eds. Ora Lassila, Ralph R. Swick. <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>

www.rpbouret.com/xml/XMLDatabaseProds.htm

A list of XML Database products, maintained by independent consultant Ronald Bourret.

Tamino XML Server, a Software AG product offering for storage, maintenance, publishing and exchange of XML documents (<http://www.softwareag.com/tamino/default.htm>)

X-Hive Corporation, developing and marketing X-Hive/DB, a native XML database (<http://www.x-hive.com/>)

IXIASOFT, the company which have developed TEXTML Server, an XML Content Server for storing, indexing, and retrieving XML documents (<http://www.ixiasoft.com/>)

SourceForge.net, a website for the development of Open Source software, including Exist, a native XML database (<http://sourceforge.net/>)

