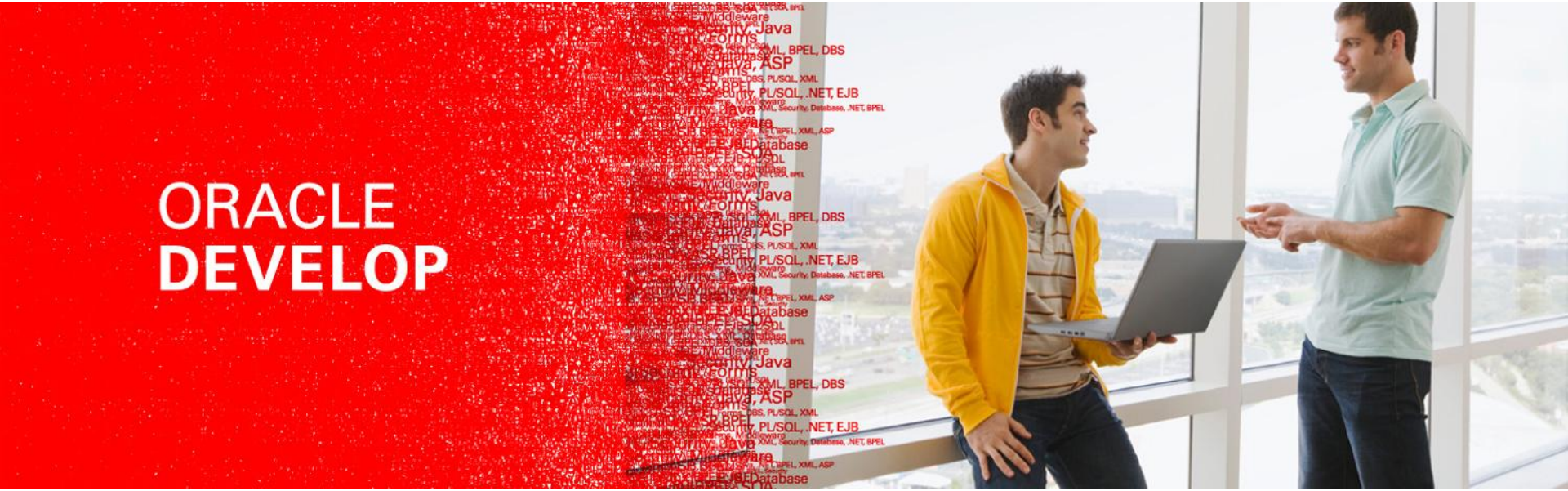


Oracle XMLDB : Case Study

Mihaela Lupu

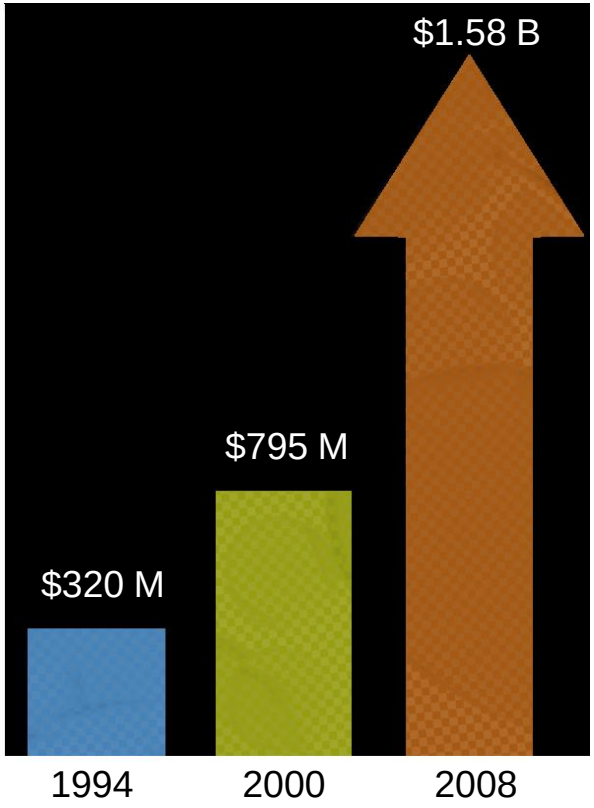
Senior Software Development Waters Corporation

Agenda

- Waters Corporation
- Xml Search Challenges
- Xml Search options
- Xml Index

Waters Corporation

- Global leader with complementary analytical technologies
  - Liquid chromatography, mass spectrometry, rheometry and microcalorimetry
- Year founded: 1958
- Publicly traded corporation (NYSE:WAT)
- Headquartered in Milford, Massachusetts
- Number of Employees: 5,000, including 2,400-strong sales and service to maintain direct link with end user



- Operating in 27 countries, including 11 manufacturing facilities, with products available in more than 50 countries

Waters Named to *Business Week* 50 Top Performing Companies

- Based on analysis of all companies in the Standard & Poor's 500
- Measured over 36 months
- Recognize sustained performance
- Leaders in innovation

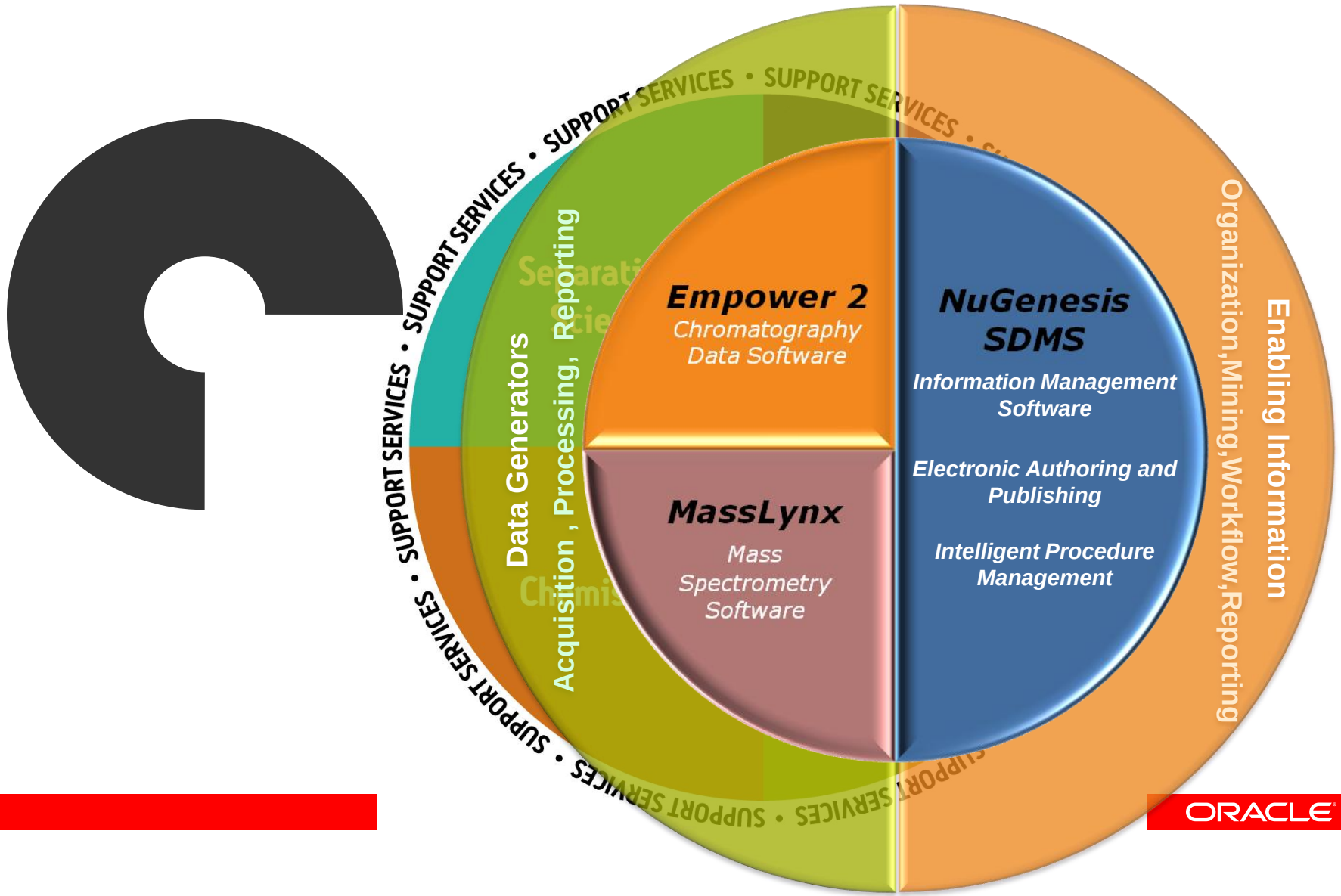
"This year's *BusinessWeek* 50 is chock-full of companies that changed the rules of engagement in their industries."

Dean Foust
BusinessWeek Magazine

"These companies are what I call the 'disrupters' of the economy."

Clayton H. Christensen
Harvard Business School professor, innovation expert

Technology Portfolio+ Expertise



Empower 2 Software Success

- 250,000+ Empower 2 Software Licenses sold
- Over 2,750+ CDS Network installations
- 50 of the Top 50 Pharmaceutical Companies have deployed Empower Networks
- 7 of the Top 10 Chemical Companies have deployed Empower Networks

NuGenesis SDMS Software Success

- 45,000+ SDMS 7.0 & 7.1 Software Licenses sold (incl. VP)
- >300 active customers
- 34 of the Top 50 Pharmaceutical Companies have deployed SDMS
- 4 of the Top 10 Chemical Companies have deployed SDMS

Business Drivers/Objectives

- Central place for storage/retrieval or data
- Robust, fast, data neutral and feature rich central data management
- Support for differently structured data records consisting of meta data (structured data stored in types) and file data (unstructured / binary data)
- Data stored in relational tables
- Specific data is stored as xml data.

Oracle XML DB Product/Project Specifics
Data volume

- Large structured data: over 10 million items
- Specific xml data structures can be even larger than 100 MB size
- Unstructured data can be larger than 40GB stream size
- Storage data rate:
  - 25MB/sec to 100MB/sec
- Real-time support (read/write in the same time)

Oracle XML DB Product/Project Specifics Application architecture

- Operating Support for Microsoft Windows 7, Windows Server 2008, and UNIX.

- Enterprise application built on a n-tier architecture, designed to scale and support n-number of users and n-number of instrument systems.

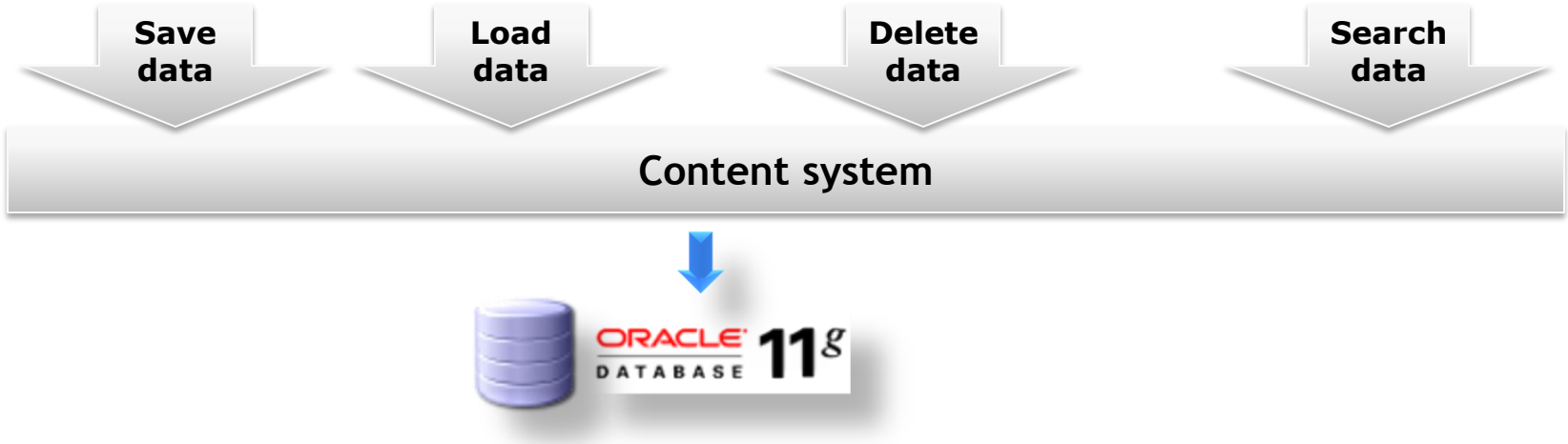
- Database layer: Oracle 11.2
- Presentation layer: WPF
- Application layer: .NET platform (C#)
- Server communication layer: WCF

Technical Strategies/Challenges

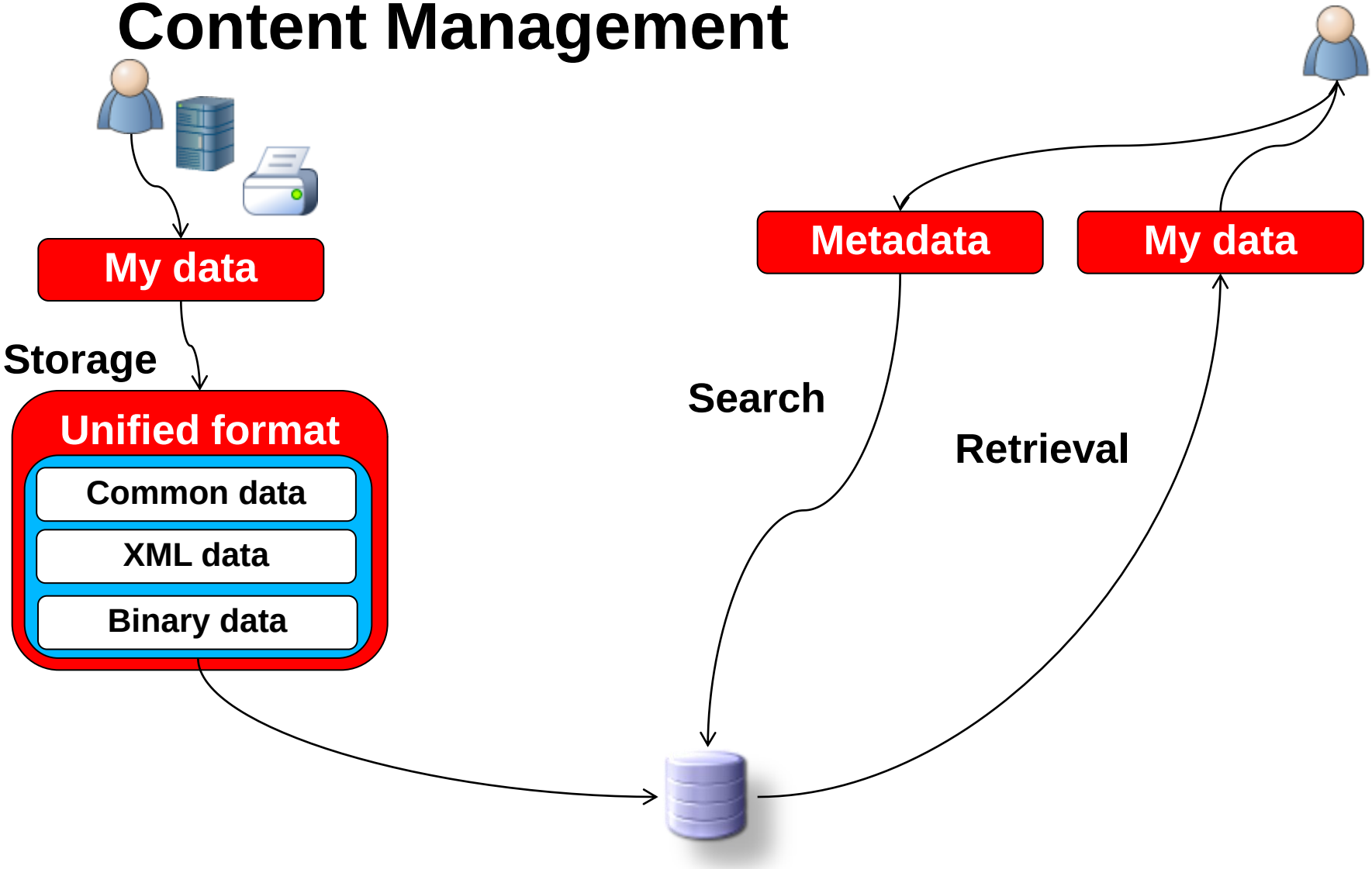
- Everybody likes to be generic
  - Generic load and Save mechanism
  - Generic search support
- Different unstructured data storage supports:
  - Database
  - File System
  - FTP
- Real time upload
- Large data support and management

Oracle XML DB Product/Project Specifics Content Management

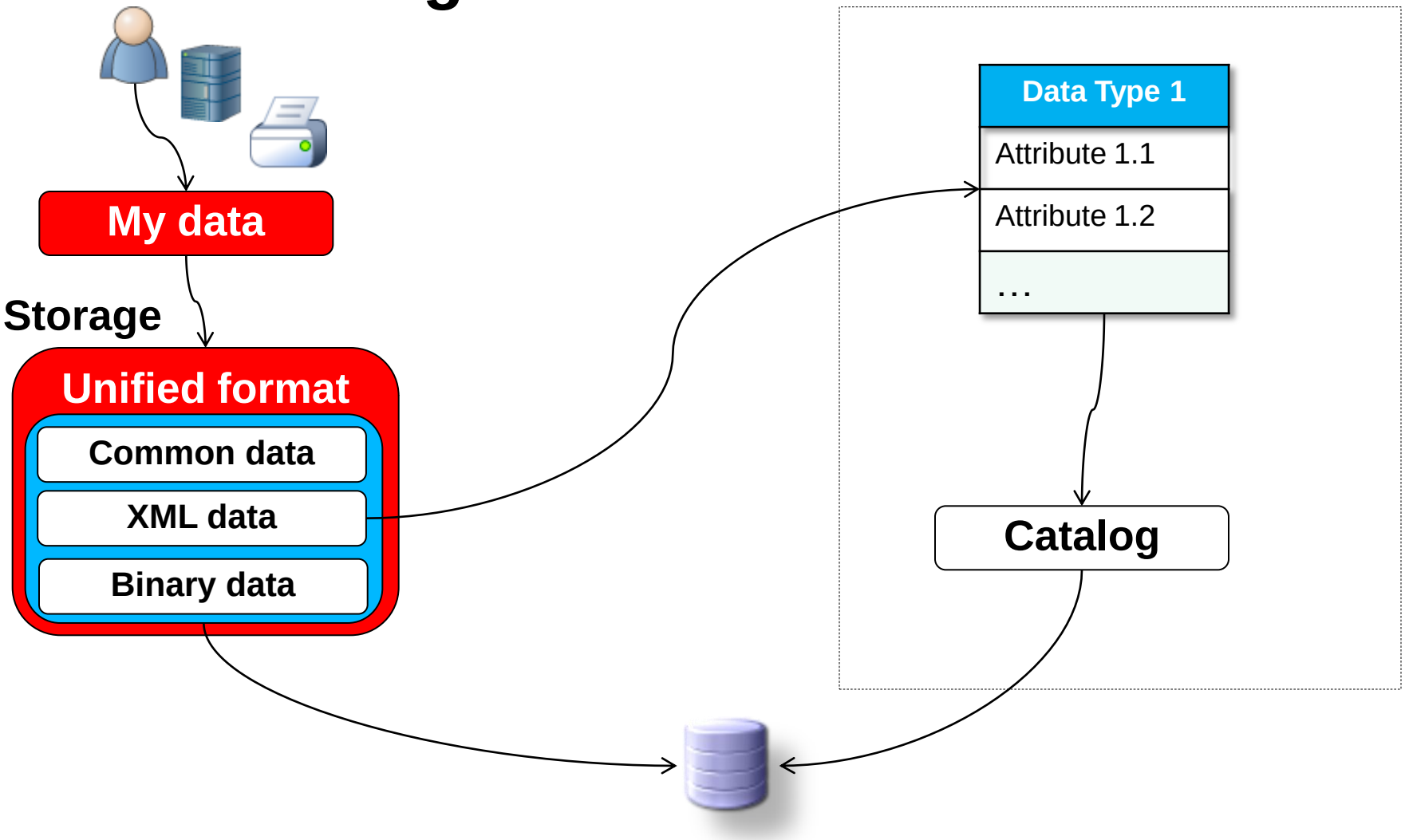
- Content Management system is the central repository of our data



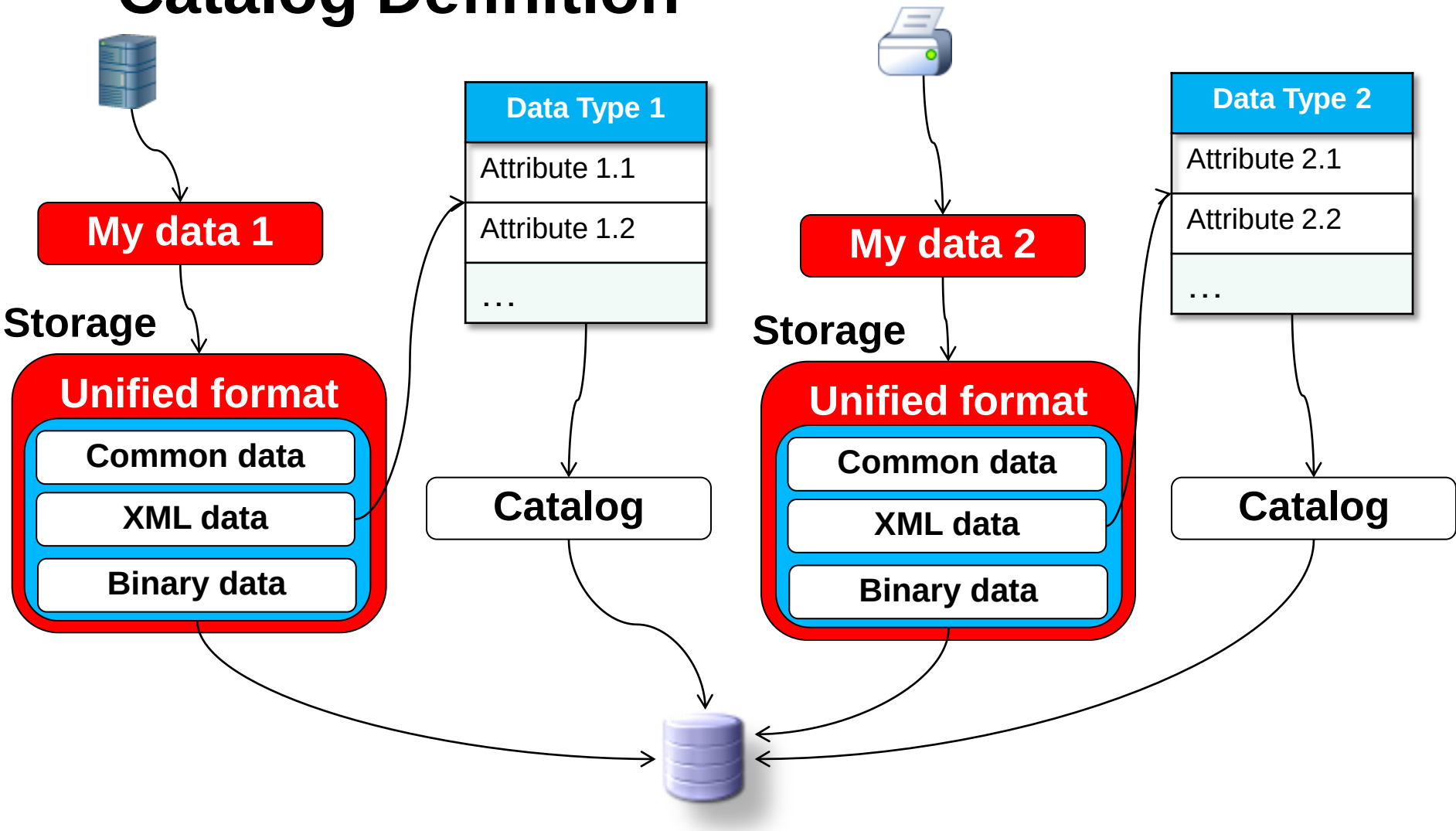
Oracle XML DB Product/Project Specifics
Content Management



Oracle XML DB Product/Project Specifics
Catalog Definition



Oracle XML DB Product/Project Specifics
Catalog Definition



Oracle XML DB Product/Project Specifics Generic Search Engine - Challenges

- The sql statement need to be built generically
- Join data from relational tables and xml content
- Xquery and Relational table search performance need to be equivalent
- DML performance of the hybrid XML/relational approach should be comparable to a pure relational approach
- Search across different data types

Oracle XML DB Product/Project Specifics
Generic Search Engine

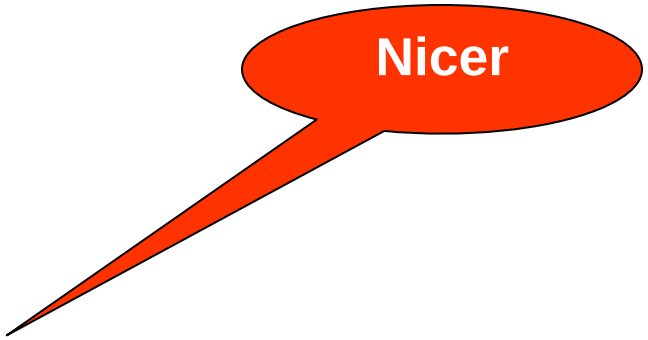
```
SELECT  relationalColumn1,
        relationalColumn2,
            …
        xmlTableColumn1,
        xmlTableColumn2,
            …
FROM RelTable A , RelTable B
LEFT OUTER XmlTable('declare namespace xxx="…";
   for $i1 in if (empty($data//xxx:…)) then <empty/> else $data//xxx:… return
   for $i2 in if (empty($i1//xxx:…)) then <empty/> else $i1//xxx:…  return
        …'  passing A.instanceData as "data"
   columns
      xmlTableColumn1 varchar2(2000) path '…') ON condition
WHERE relationalColumnX = :1
    AND xmlTableColumnY = :2
```

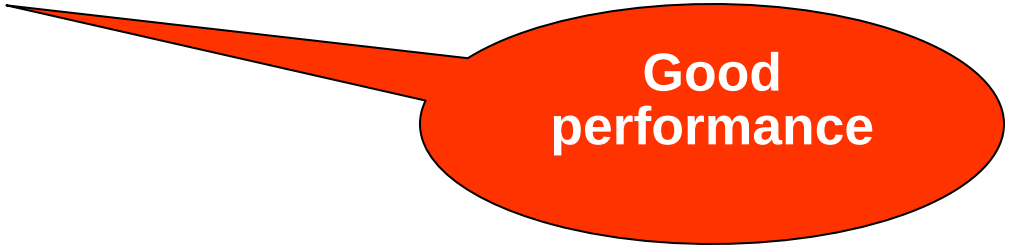
Oracle XML DB Product/Project Specifics Generic Search Engine

- Define a set of relational views (over XML) specific to each type of data
- Rewrite queries to go against the relational views
- Define structured xmlindex groups to correspond to the relational views
- Define a set of generic metadata (most frequently searched elements) that will belong to the catalog

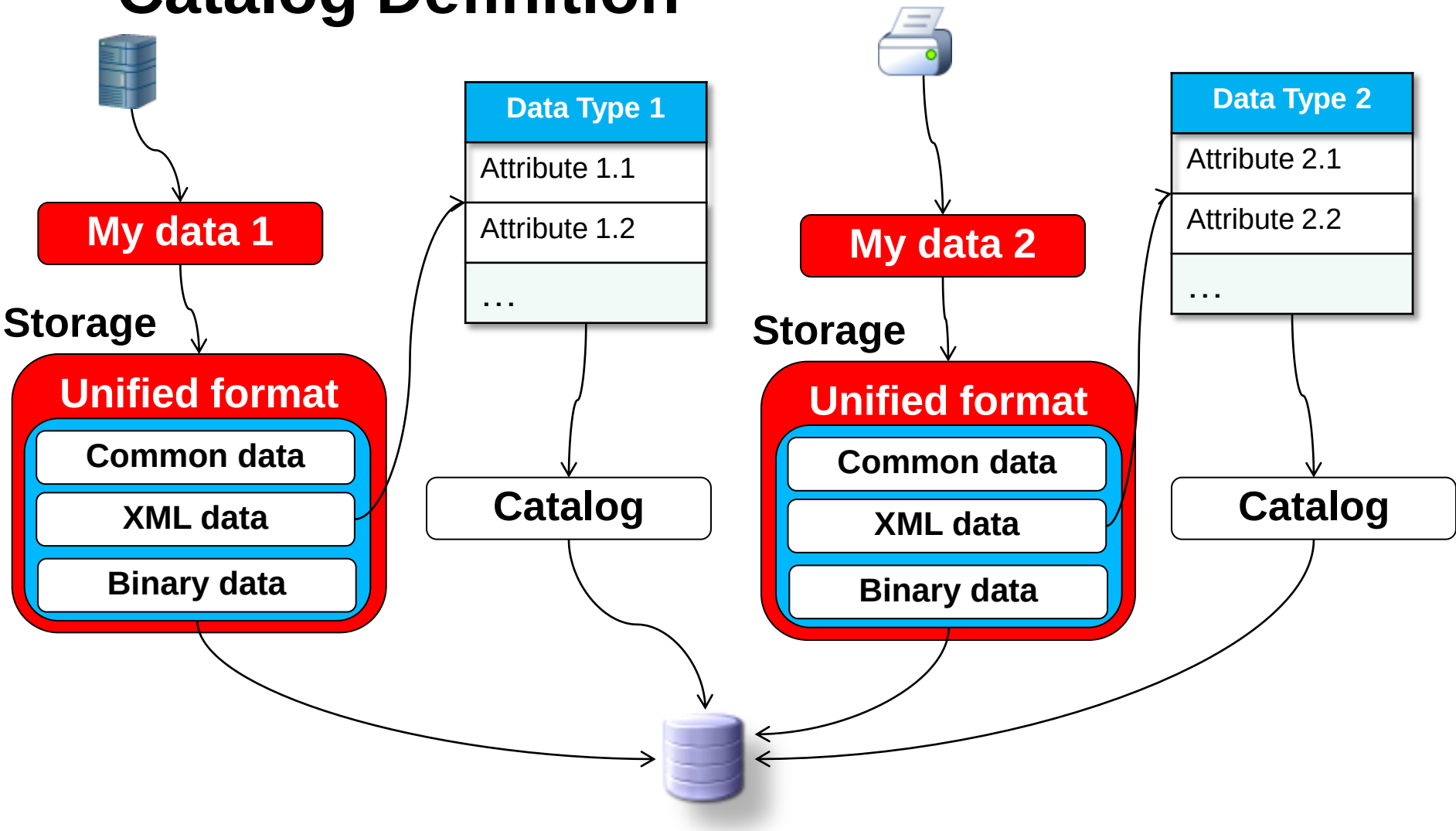
Oracle XML DB Product/Project Specifics
Generic Search Engine

```
SELECT  relationalColumn1,
        relationalColumn2,
            …
        xmlTableColumn1,
        xmlTableColumn2,
            …
FROM RelTable A , RelTable B

LEFT OUTER XmlIndexView1 ON condition
LEFT OUTER XmlIndexView2 ON condition

WHERE relationalColumnX = :1
   AND xmlTableColumnY = :2
```

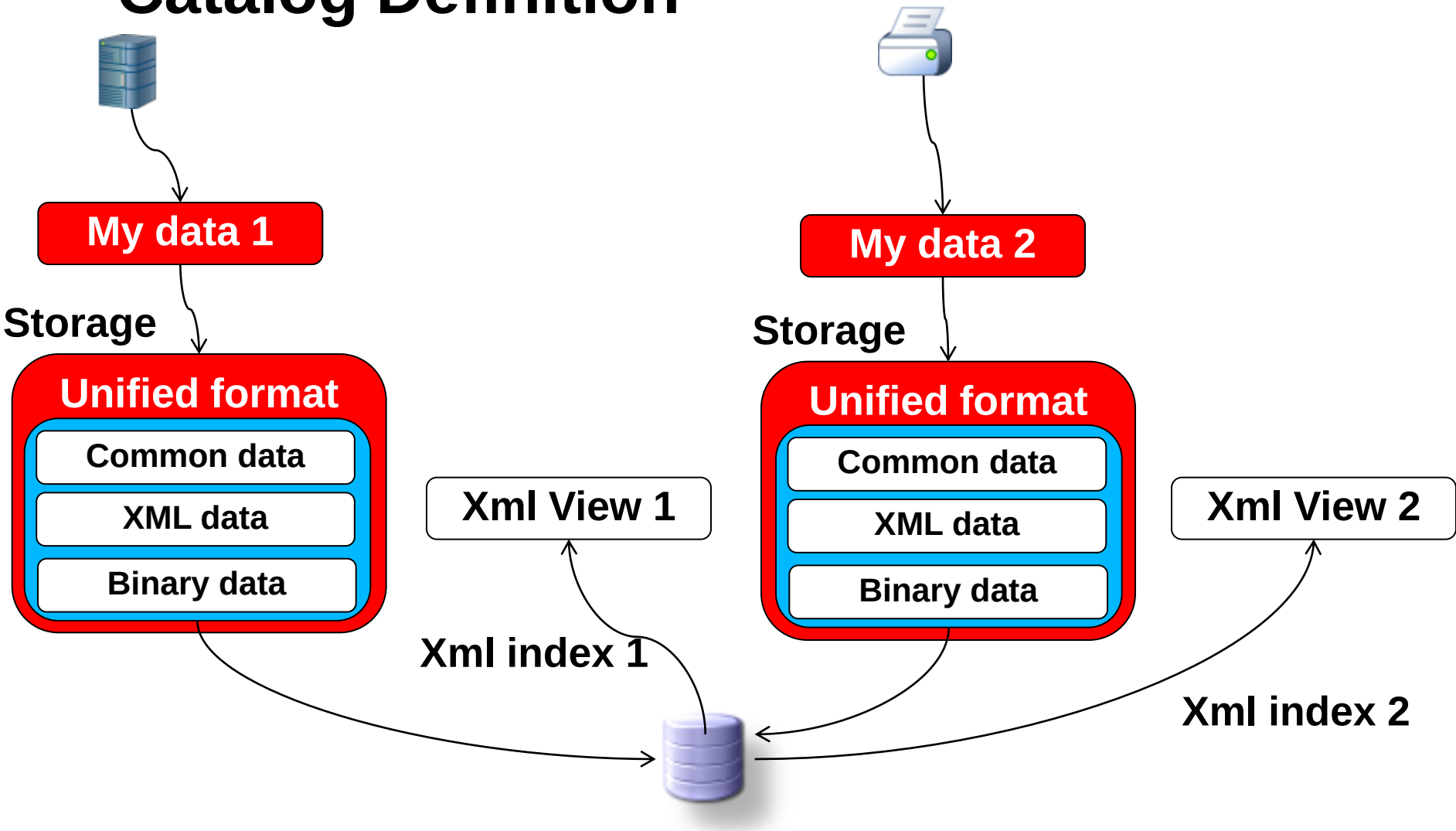


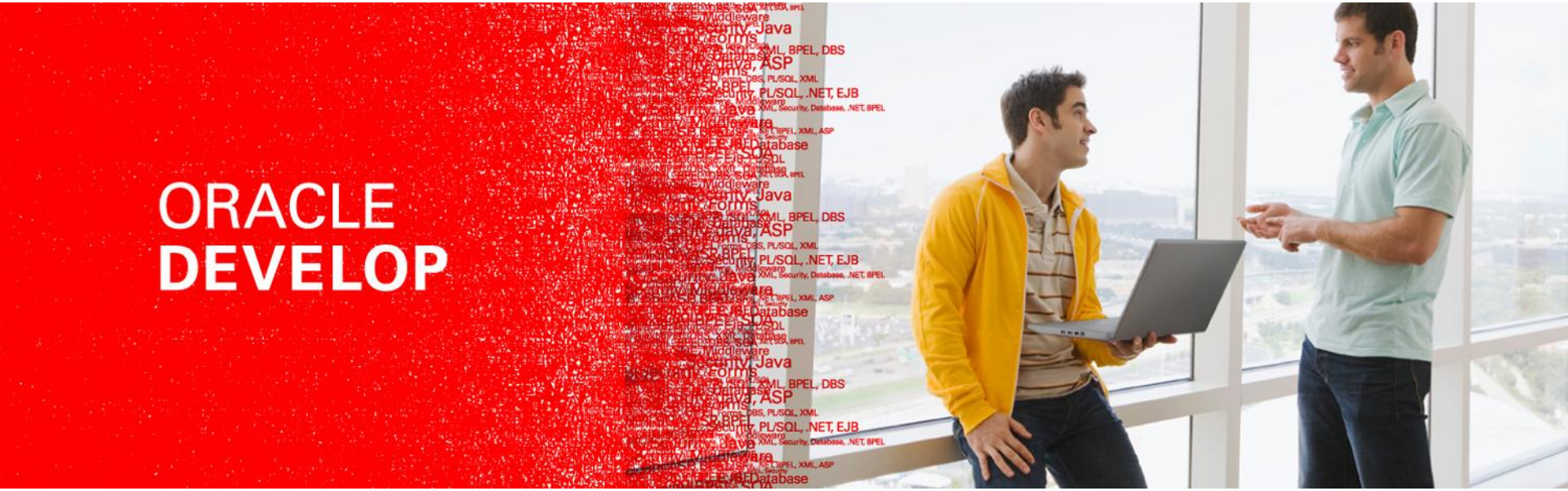


Oracle XML DB Product/Project Specifics
Catalog Definition



Oracle XML DB Product/Project Specifics
Catalog Definition





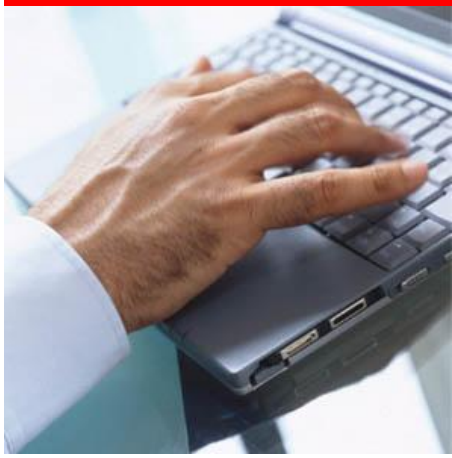
Oracle XMLDB : XML Index

Geeta Arora

Senior Software Development Manager

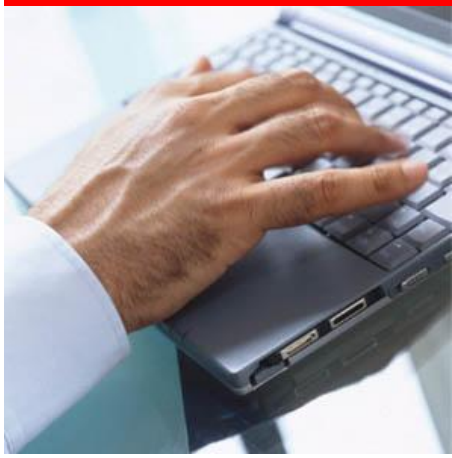
ORACLE

Agenda



- XML DB Background
- XMLIndex Background
- Unstructured XMLIndex
- Structured XMLIndex
- Conclusion

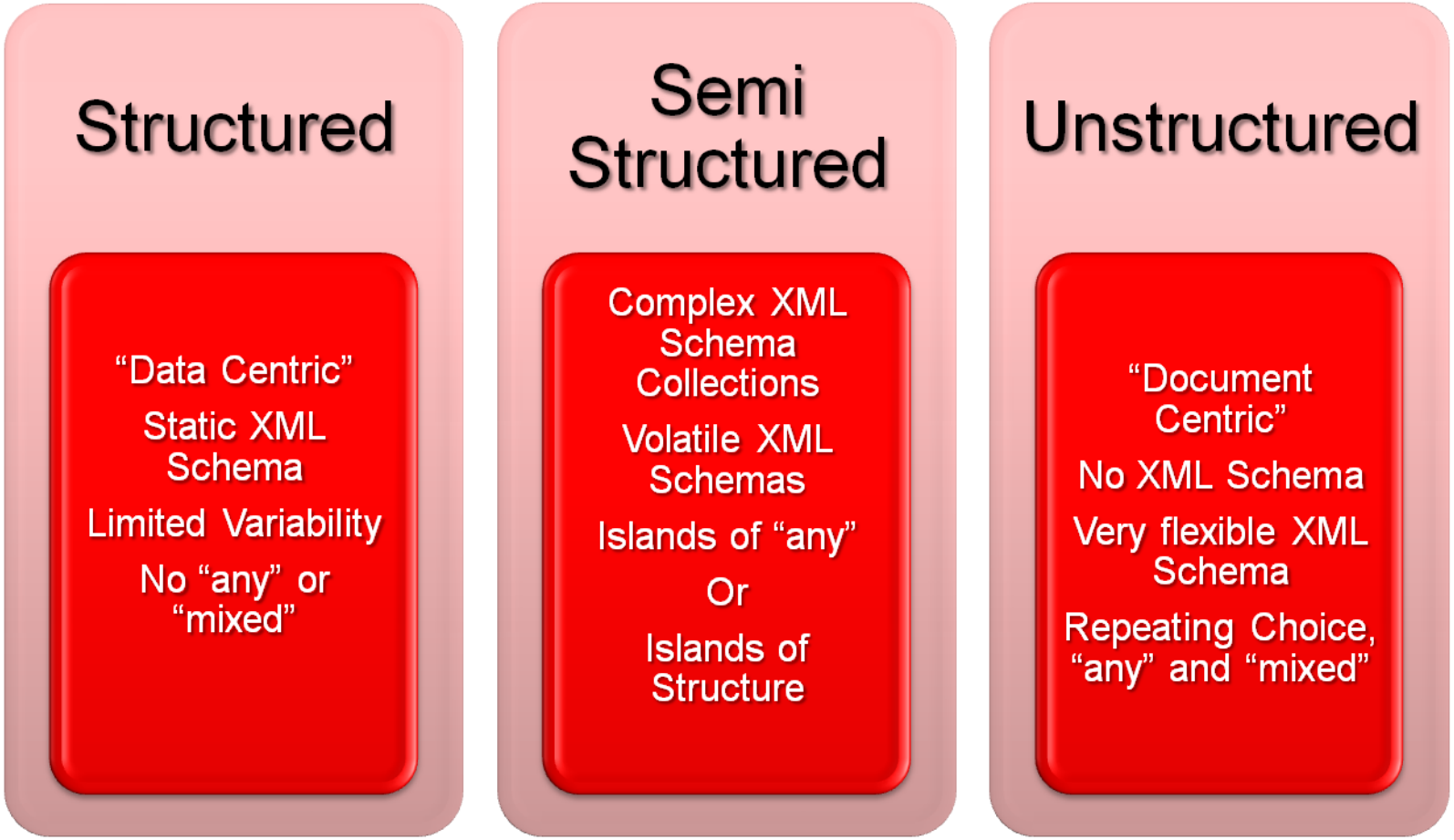
Agenda



- XML DB Background
- XMLIndex Background
- Unstructured XMLIndex
- Structured XMLIndex
- Conclusion

XMLDB Background

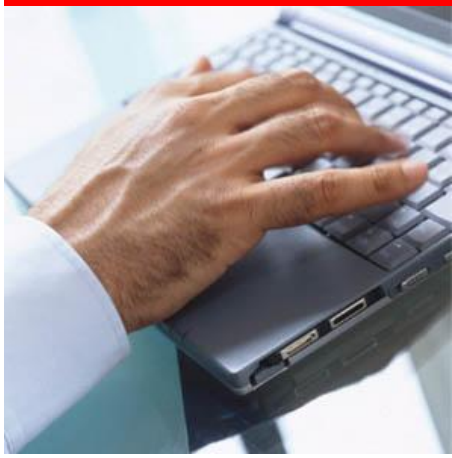
- Stored XML in database as XMLType
  - Unified management of structured data and semi-structured content
  - High scalability, faster queriability, optimized updates
- Multiple Storage options
  - Structured Object Relational Storage with B-tree index
  - Binary XML Storage
  - CLOB Storage
  - Hybrid (Object Relational with CLOB) storage
- XMLIndex : Unstructured and Structured components
- Maintains application transparency to physical storage choice


Structured
"Data Centric"
Static XML Schema
Limited Variability
No "any" or "mixed"
Semi Structured
Complex XML Schema Collections
Volatile XML Schemas
Islands of "any"
Or
Islands of Structure
Unstructured
"Document Centric"
No XML Schema
Very flexible XML Schema
Repeating Choice, "any" and "mixed"

XML usecases

DOCUMENT	*Unstructured*	Binary XML Storage + XMLIndex Unstructured Component	Binary XML Storage + XMLIndex Structured Component
	Structured	Hybrid Storage (Object Relational + embedded CLOB)	Structured Storage (Object Relational) + B-tree index
		Unstructured	*Structured*
		Parts of the document	

Agenda

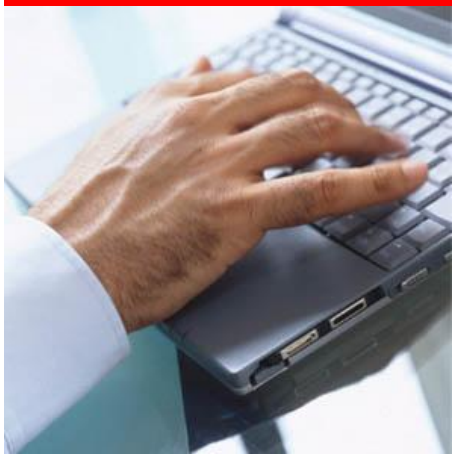


- XML DB Background
- XMLIndex Background
- Unstructured XMLIndex
- Structured XMLIndex
- Conclusion

XMLIndex

- Effective indexing strategy for XML documents
  - CLOB or Binary XML storage
- Improves XPath based fragment extraction
- Handles Path and Value based predicates
- Datatype aware
- Good DML performance with path subsetting, asynchronous maintenance etc.

Agenda



- XML DB Background
- XMLIndex Background
- Unstructured XMLIndex
- Structured XMLIndex
- Conclusion

XMLIndex: Unstructured Component

- Available since 11gR1
- Usecase: AdHoc XPaths not known in advance
- Organizes paths and values in single path table
- Allows easy indexing of interesting sub-trees
- Whole spectrum possible – single leaf element to everything
- Allows asynchronous maintenance
- Updates to document result in piece-wise index updates

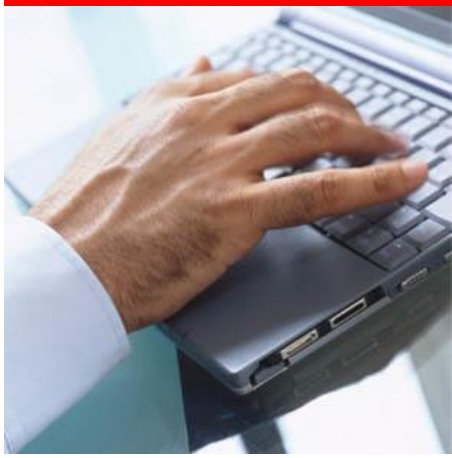
Unstructured XMLIndex Layout

RID	path	Order key	locator	value
10	/Document	1	Locator to get binary content	
10	/Document/Title	1.1	Locator to get binary content	Indexing XML Techniques
10	/Document/pubDate	1.3	Locator to get binary content	2009-04-10
20	/Book	1	Locator to get binary content	
20	/Book/Title	1.1	Locator to get binary content	Object relational storage

Unstructured XMLIndex - Path Subsetting

- Specify
  - nodes that will be used in common queries or
  - nodes that will rarely be used
- Can change the specified paths later
- Better DDL, DML performance
- Reduces size of primary and secondary indexes. Less storage overhead
- Transparent to queries!

Agenda



- XML DB Background
- XMLIndex Background
- Unstructured XMLIndex
- Structured XMLIndex
- Conclusion

Structured Index

- Available since 11gR2
- Usecase: Structured “islands” inside XML
- Provides Relational View over XML data
- Efficient Value Search of structured components
  - Relational query performance
  - XML Storage & Schema Independent
- Index size is small and light-weight
  - No path information is stored in the indexed tables
- Smooth Integration of XML with existing relational applications using XMLTable Design Pattern

SXI UseCase 1 : XML with structured component

- Overall XML in document is content driven
- Document has structured data component “metadata”, e.g. title, date, authors
- Typical query: find document with specific structured data value
- Example query:

```
SELECT *
FROM DOCUMENT_TAB  doc
WHERE XMLEXISTS(
'$doc//document [ title = “indexing XML Techniques” and
   pubdate > xs:date(“2009-03-01”) and pubdate < xs:date(“2009-
   12-31”)]' PASSING VALUE(doc) AS “doc”)
```

Solution using Structured XMLIndex

- Idea: Decompose the structured components relationally
- Create a side pivot table with
  - title, pubdate are pivoted as columns of the table.
- The example query can be rewritten using the side pivot table

```
SELECT *
FROM DOCUMENT_TAB doc
 WHERE EXISTS(
        SELECT 1
                FROM PIVOT_TAB p
                WHERE p.title = “indexing XML Technique” AND
                      p.pubdate >  to_date(“2009-03-01”) AND
                p.pubdate < to_date(“2009-12-31”) AND
                        p.ROWID = doc.ROWID)
```

SXI UseCase 2 : Relational Views

- Original Relational Table
  - *Document_tab_rel (title, pubdate, authors, rest_of_document CLOB)*
- Original application written against Relational Table
  - *SELECT * FROM document_tab_rel*
    WHERE title = "indexing XML Technique"
- Move data to XML
  - *CREATE TABLE document_tab of XMLType …*
  - *DROP TABLE document_tab_rel;*
- Define relational views over XML data
  - *CREATE VIEW document_tab_rel AS SELECT*
    title, pubdate, FROM document_tab, XMLTable ('//document'
    COLUMNS title varchar(100) PATH 'title',
    pubdate date PATH 'pubdate');
- XMLIndex to correspond to view

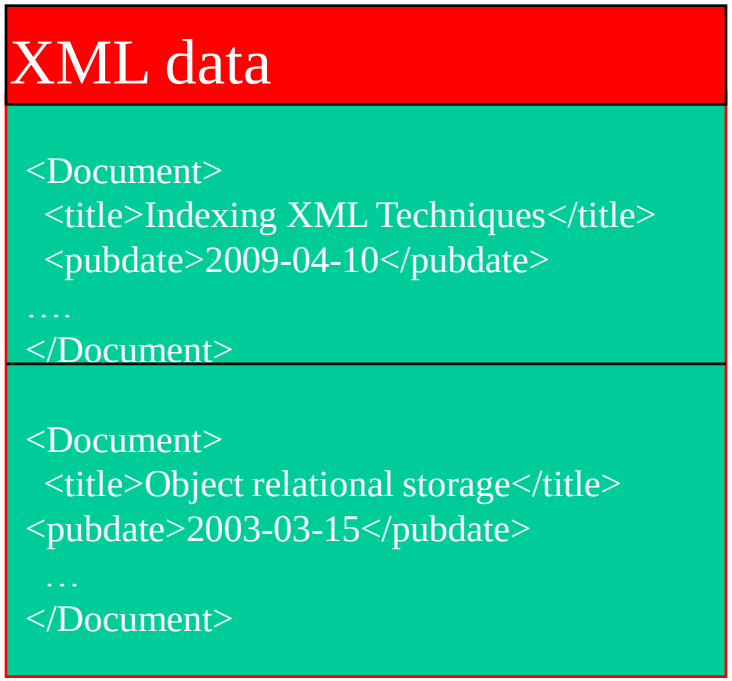
Structured XMLIndex Creation

- Example

```
CREATE INDEX paper_info ON DOCUMENT_TAB indextype is
   xdb.xmlindex
PARAMETERS(XMLTABLE('//document'  PIVOT_TAB
                    COLUMNS
                      title varchar(100) PATH 'title',
                      pubdate date PATH 'pubdate'))
```

- XPath *'//document'* used to identify nodes stored in each row of the table
- Multiple leaf data is projected out as columns of XMLTABLE
- Syntax similar to XMLTABLE construct in SQL/XML

Structured XMLIndex Layout



Structured XMLIndex		
RowID	Title	Pubdate
10	Indexing XML Techniques	2009-04-10
20	Object relational storage	2003-03-15

Mater-detail Aspect of Structured XMLIndex

- What about collection element Value ?
- Store them in a separate nested table
- Structured XMLIndex with chaining option

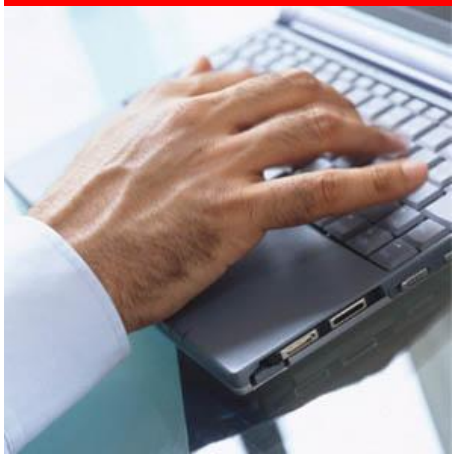
```
CREATE INDEX paper_info ON PAPER_TAB indextype is xdb.xmlindex
    XMLTABLE('//document' PIVOT_TAB
                COLUMNS
                  title varchar(100) PATH 'title',
                  pubdate date PATH 'pubdate' ,
                 authorList XML PATH '//authorList' VIRTUAL
    XMLTABLE '.'   PIVOT_NTAB
                COLUMNS
              authorname varchar(20) PATH 'authorName')
```

- Queries over the base XML storage "rewritten" to go against the XMLIndex Nested tables

XMLIndex with Secondary Index

- Unstructured Index:
  - Secondary relational indexes created automatically on Path and Value columns
  - Can create Text Index on Value column
- Structured Index:
  - Can create secondary relational indexes on structured xmlindex tables
    - Bitmap index or B+ tree
    - Statistics can be built and maintained for different indexes
  - Can create Text Index on projected text column

Agenda



- XML DB Background
- XMLIndex Background
- Unstructured XMLIndex
- Structured XMLIndex
- Conclusion

XMLIndex considerations

- Choice of index determined by
  - Structure in data
  - Query paradigm
- XMLIndex (unstructured component)
  - Can handle wide variety of queries, AdHoc XPaths
  - Scalar value lookups and fragment retrieval
  - Can index desired sub-trees including hierarchies
- XMLIndex (structured component)
  - Ideal for scalar value lookups
  - Speeding up queries on islands of structure
  - Author, Date, Title fields for example
  - Captures the “attributes” of an “entity” together using E/R Model

Oracle XML DB DEMOgrounds Booths

- **Come by our DEMOgrounds booths to have one-on-one conversation with our team members**
  - Moscone West: W-41, W-44, and W-61

Tuesday Sessions

S317480: Managing XML Content with Oracle XML: Getting the Best Bang for the Buck
Moscone South, Rm 200

2:00 PM – 3:00 PM

S317428: ProQuest Use Case
Moscone South, Rm 200

5:00 PM – 6:00 PM

Wednesday Sessions

S317650 : S&P Use Case
Hotel Nikko, Nikko Ballroom I

10:00 AM – 11:00 AM

S319105: Interfacing with Your Database via Oracle XML DB

Hotel Nikko/Bay View

11:30 AM – 12:30 PM

S317648: PolarLake Use Case, XDK, and XQJ
Hotel Nikko Nikko Ballroom I

1:00 PM – 2:00 PM

Thursday Sessions

S317504: Waters Use Case and Structured XMLIndex
Moscone South, Rm 200

10:30 AM – 11:30 AM

S317528: Working with Complex XML Schemas: Not as Hard as You Might Think
Hotel Nikko Nikko Ballroom I

2:00 PM – 3:00 PM

S317657: XBRL Expert Panel - Using Oracle Database as an XBRL Repository
Hotel Nikko Nikko Ballroom I

3:30 PM – 4:30 PM


Q&A