IBM Software Group

How to Boost Performance of your XML Data in DB2

Matthias Nicola
DB2/XML Performance
IBM Silicon Valley Lab
mnicola@us.ibm.com

@business on demand software

DB2 Data Management Software

Aug 16, 2005

Agenda

- Why XML? Why XML in Databases?
- Existing XML storage options and problems
- Native XML in DB2: Performance & Flexibility
  - Overview
  - Native XML Storage
  - XML Indexes
  - XQuery, and the Integration with SQL
- Summary

Why XML?

- **Flexibility, Flexibility, Flexibility !**
- **XML is vendor and platform independent**
- **XML is a very flexible data model**
  - **for structured data, semi-structured data, schema-less data**
- **Easy to extend => define new tags as needed**
- **XML is self-describing - any XML parser can "understand" it !**
- **Easy to "validate" XML, i.e. to check compliance with a schema**
  - any XML parser can do it!
- **Easy to transform XML documents into other formats (HTML, etc.)**
- **Easy to share XML between applications, businesses, processes, …**

Why use XML with Databases?

- **Managing large volumes of XML data is a DB problem!**
  - Efficient Search & Retrieval of XML
  - Persistency, Recovery, Transactions, ACID
  - Performance, Scalability
  - *…all the same reasons as for relational data!*
- **Integration**
  - Integrate new XML data with existing relational data
  - Publish (relational) data as XML
  - Database support for web applications, SOA, web services (SOAP)

XML Databases

- **XML-enabled Databases**
  - The core data model is not XML (but usually relational)
  - Mapping between XML data model and DB's data model is required
  - E.g.: DB2 XML Extender (V7, V8)

- **Native XML Databases**
  - Use the hierarchical XML data model to store and process XML
  - No mapping, no storage as text
  - E.g.: Forthcoming version of DB2

Agenda

- Why XML? Why XML in Databases?
- **Existing XML storage options and problems**
- Native XML in DB2: Performance & Flexibility
  - Overview
  - Native XML Storage
  - XML Indexes
  - XQuery, and the Integration with SQL
- Summary

Option 1: Storing XML as CLOB/Varchar

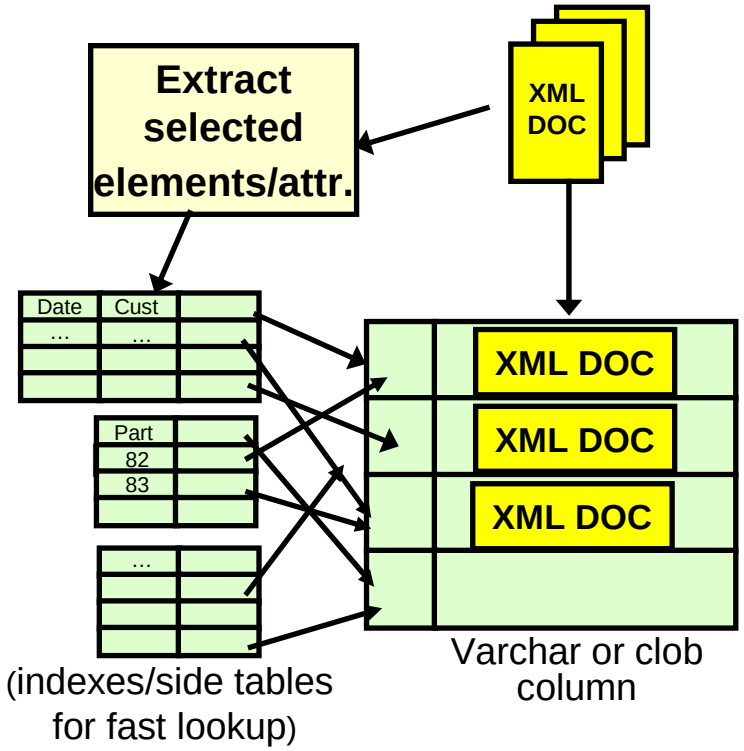
```
<order date="2004-11-18">
    <customer>Thompson</customer>
    <part key="82"> .... </part>
    <part key="83"> .... </part>
    ...
</order>
```

Table

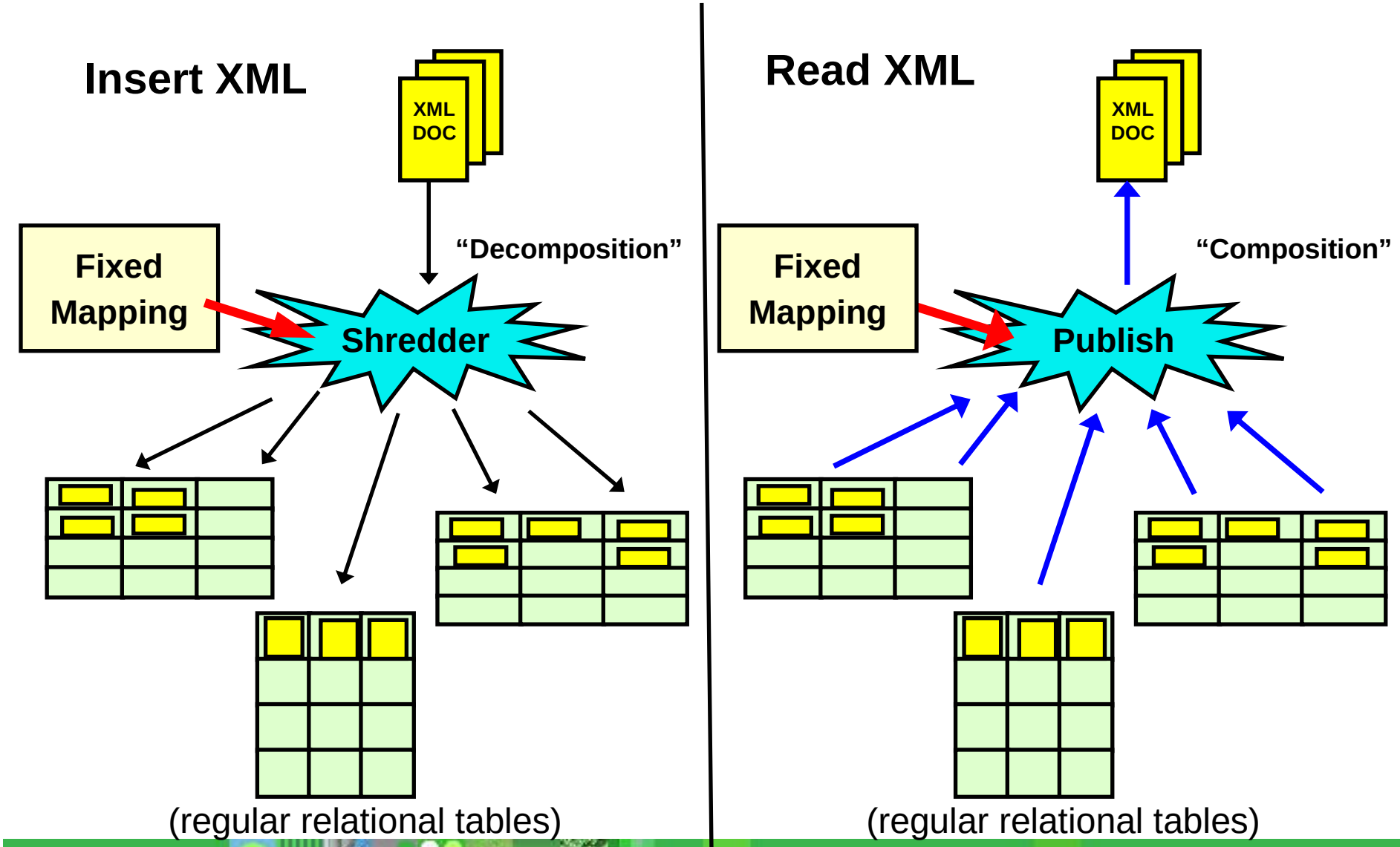
Order (CLOB)	ID
`<order date="2004-11-18"> <customer>Thompson</customer> <part key="82"> .... </part> <part key="83"> .... </part> .... </order>`	1
…	…
…	…

The XML structure is ignored !

Limited support for query performance:



Option 2: Mapping XML to Relational (Shredding)



CLOB/Varchar: Performance Characteristics

- Fast insert/retrieval of *full* documents, but:
- Any sub-document level access requires costly XML Parsing:
  - Evaluating XQuery/Xpath
  - Retrieving partial documents
  - Sub-document level updates (e.g. element values)
- XML parsing is very CPU intensive, esp. for large data volumes: Performance Bottleneck

Problems with Shredded Storage

- Mapping from XML to relational schema can be very complex
- May require dozens (if not hundreds) of tables to represent a single XML schema
- Complex multi-way joins to reconstruct XML docs
- Translation of complex XQueries to SQL:
  - Can result in inefficient SQL
  - Sometimes even prohibitively complex
- Mapping is fixed, no schema flexibility

Shredding: A simple case

```
<DEPARTMENT  deptid="15" deptname="Sales">
    <EMPLOYEE>
        <EMPNO>10</EMPNO>
        <FIRSTNAME>CHRISTINE</FIRSTNAME>
        <LASTNAME>SMITH</LASTNAME>
        <PHONE>408-463-4963</PHONE>
        <SALARY>52750.00</SALARY>
    </EMPLOYEE>
    <EMPLOYEE>
        <EMPNO>27</EMPNO>
        <FIRSTNAME>MICHAEL</FIRSTNAME>
        <LASTNAME>THOMPSON</LASTNAME>
        <PHONE>406-463-1234</PHONE>
        <SALARY>41250.00</SALARY>
    </EMPLOYEE>
</DEPARTMENT>
```

Department

DEPTID	DEPTNAME
15	Sales

Employee

DEPTID	EMPNO	FIRSTNAME	LASTNAME	PHONE	SALARY
15	27	MICHAEL	THOMPSON	406-463-1234	41250
15	10	CHRISTINE	SMITH	408-463-4963	52750

Shredding: A schema change…

"Employees are now allowed to have multiple phone numbers…"

```
<DEPARTMENT  deptid="15" deptname="Sales">
      <EMPLOYEE>
         <EMPNO>10</EMPNO>
         <FIRSTNAME>CHRISTINE</FIRSTNAME>
         <LASTNAME>SMITH</LASTNAME>
         <PHONE>408-463-4963</PHONE>
         <PHONE>415-010-1234</PHONE>
         <SALARY>52750.00</SALARY>
      </EMPLOYEE>
      <EMPLOYEE>
          <EMPNO>27</EMPNO>
          <FIRSTNAME>MICHAEL</FIRSTNAME>
          <LASTNAME>THOMPSON</LASTNAME>
          <PHONE>406-463-1234</PHONE>
          <SALARY>41250.00</SALARY>
       </EMPLOYEE>
</DEPARTMENT>
```

Requires:
- **Normalization of existing data !**
- **Modification of the mapping**
- **Change of applications**

Phone

EMPNO	PHONE
27	406-463-1234
10	415-010-1234
10	408-463-4963

Department

DEPTID	DEPTNAME
15	Sales

Costly!

Employee

DEPTID	EMPNO	FIRSTNAME	LASTNAME	PHONE	SALARY
15	27	MICHAEL	THOMPSON	406-463-1234	41250
15	10	CHRISTINE	SMITH	408-463-4963	52750

Key Advantages of DB2 native XML Support

- No XML parsing for sub-doc level access
- No mapping to a different data model
- Schema flexibility (0, 1, or many XML schemas)
- No translation from XQuery to SQL
- No multi-way joins to reconstruct documents

→ **Better Query Performance, More Flexibility**

Agenda

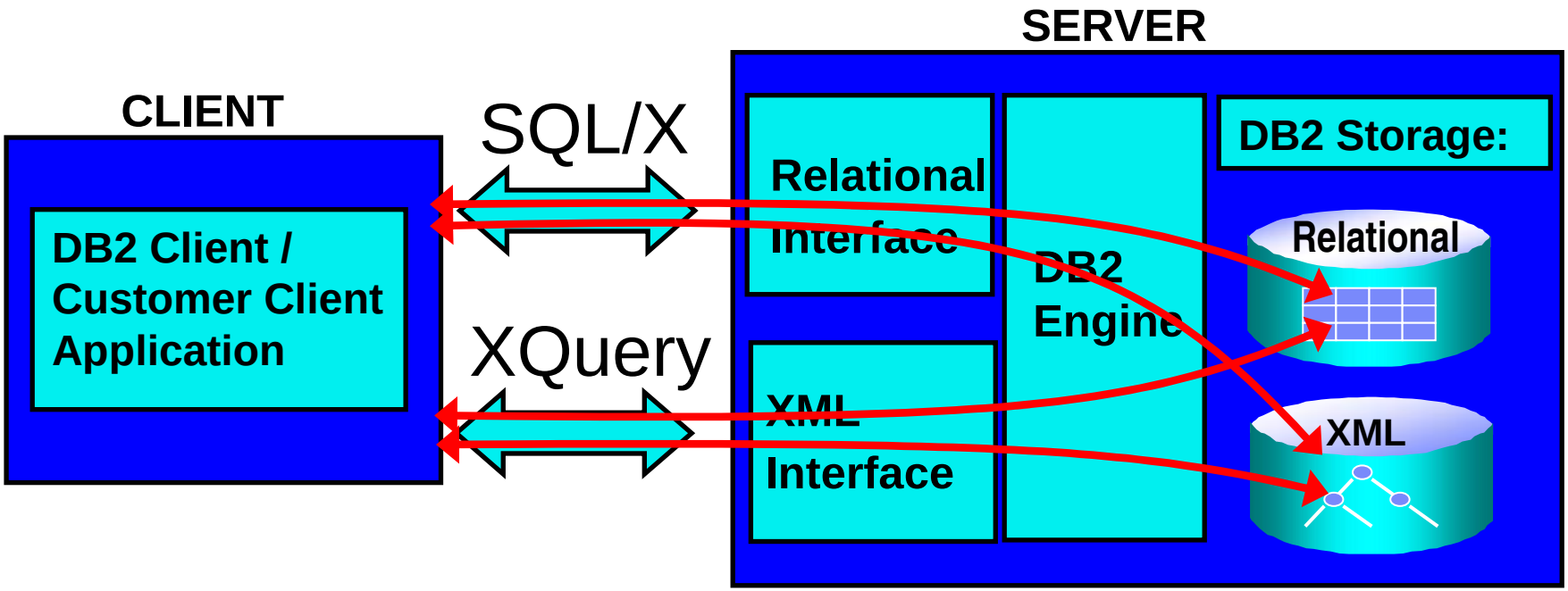
- Why XML? Why XML in Databases?
- Existing XML storage options and problems
- **Native XML in DB2: Performance & Flexibility**
  - **Overview**
  - **Native XML Storage**
  - **XML Indexes**
  - **XQuery, and the Integration with SQL**
- Summary

Native XML in DB2: Key Themes

- Standards compliant + driving the standards
  - XML, XQuery, SQL/XML, XML Schema…
- Flexibility, because that is what XML is all about..
  - Zero, one, or many XML schemas per XML column
- Native (hierarchical) Storage & Sophisticated XML Indexes
  - New “pivot join” to evaluate many predicates concurrently
- Integrated in DB2
  - Leveraging scalability, reliability, performance, availability…
- Integrated with application APIs:
  - JDBC, ODBC, .NET, embedded SQL, CLI,…
- Integrated with SQL
  - Access relational data and XML data in same statement

Integration of XML & Relational Capabilities

- **Native XML data type** (server & client side)
  - (not Varchar, not CLOB, not object-relational !)
- XML Capabilities in all DB2 components
- Applications use XML, or relational data, or both !

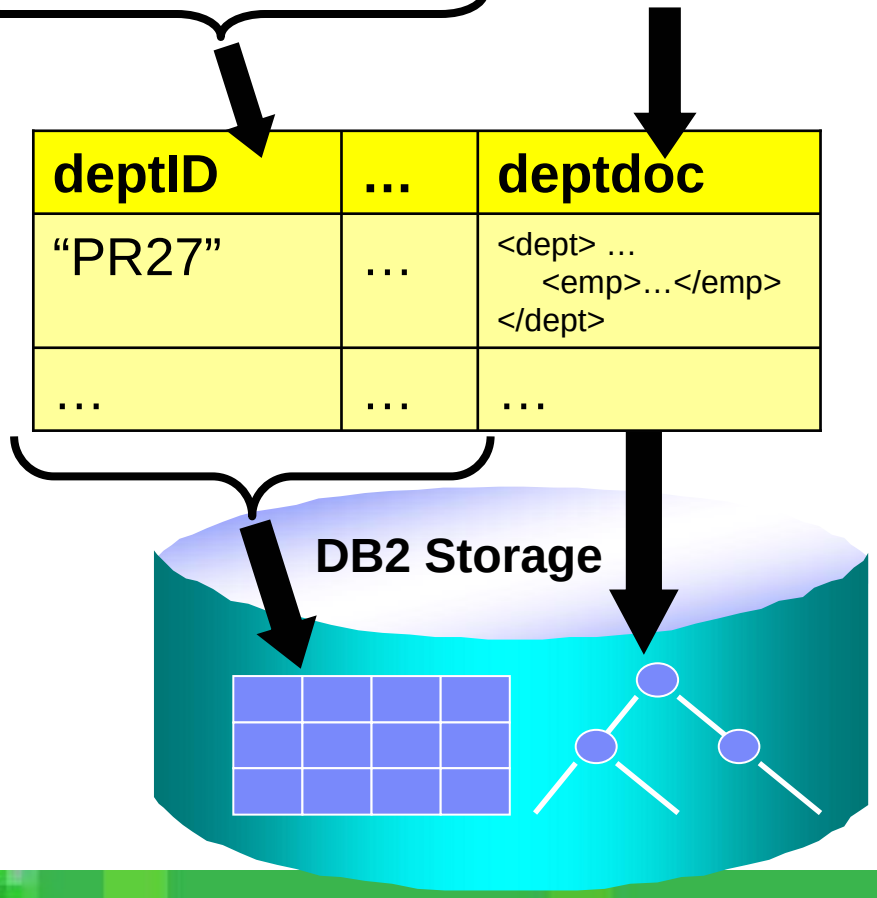


Native XML Storage

- DB2 stores XML in **parsed hierarchical** format (similar to the DOM representation of the XML infoset)

create table dept (deptID char(8),…, deptdoc xml);

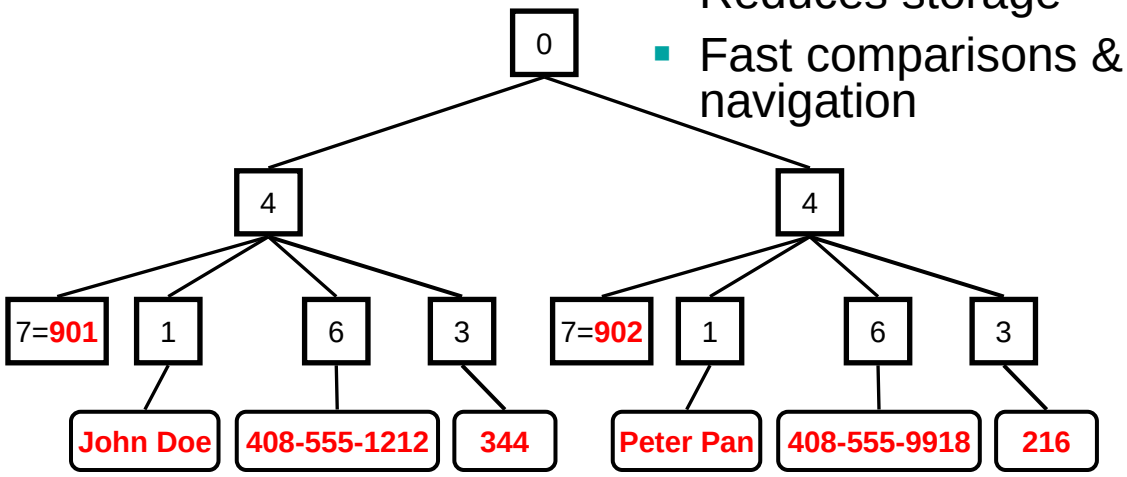
- Relational columns are stored in relational format (tables)
- XML columns are stored **natively** in the XQuery Data Model (i.e. “as trees”)



Efficient Document Tree Storage

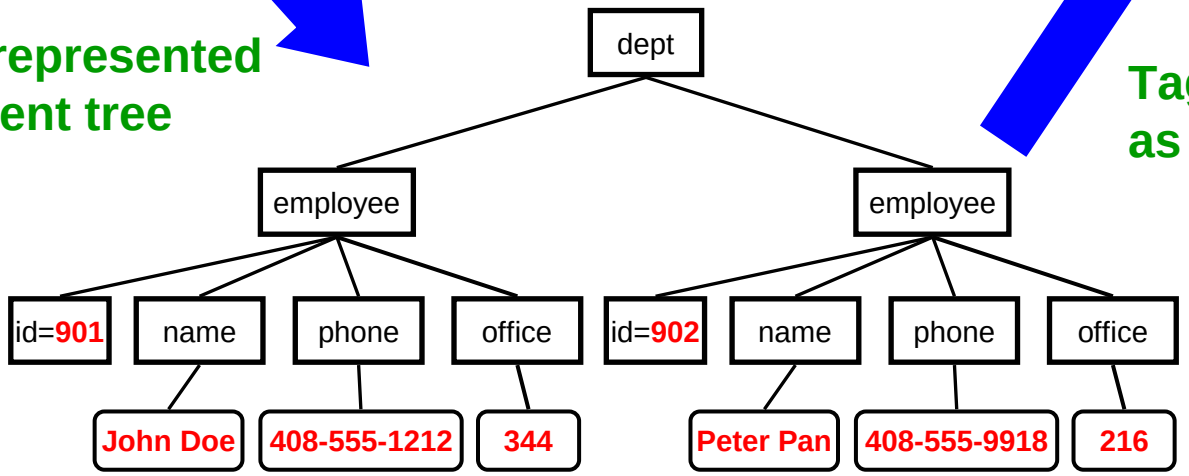
```
<dept>
   <employee id=901>
        <name>John Doe</name>
        <phone>408 555 1212</phone>
        <office>344</office>
    </employee>
    <employee id=902>
        <name>Peter Pan</name>
        <phone>408 555 9918</phone>
        <office>216</office>
    </employee>
</dept>
```

- Reduces storage
- Fast comparisons & navigation



XML text represented as document tree

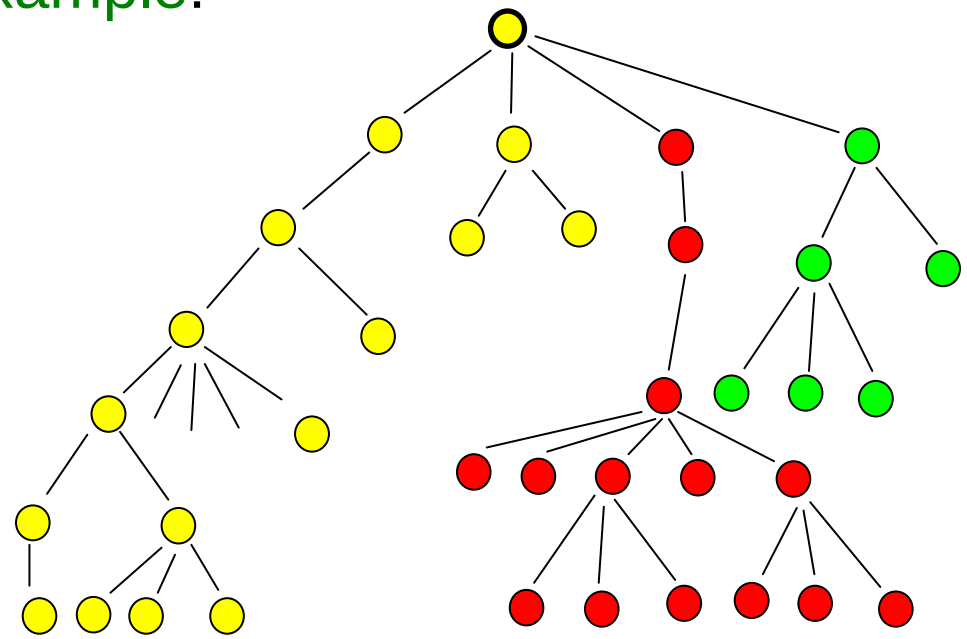
Tag names encoded as unique integers



XML Node Storage Layout

- Node hierarchy of an XML document stored on DB2 pages
- Documents that don't fit on 1 page: split into regions/pages
- Docs < 1 page: 1 region, multiple docs/regions per page

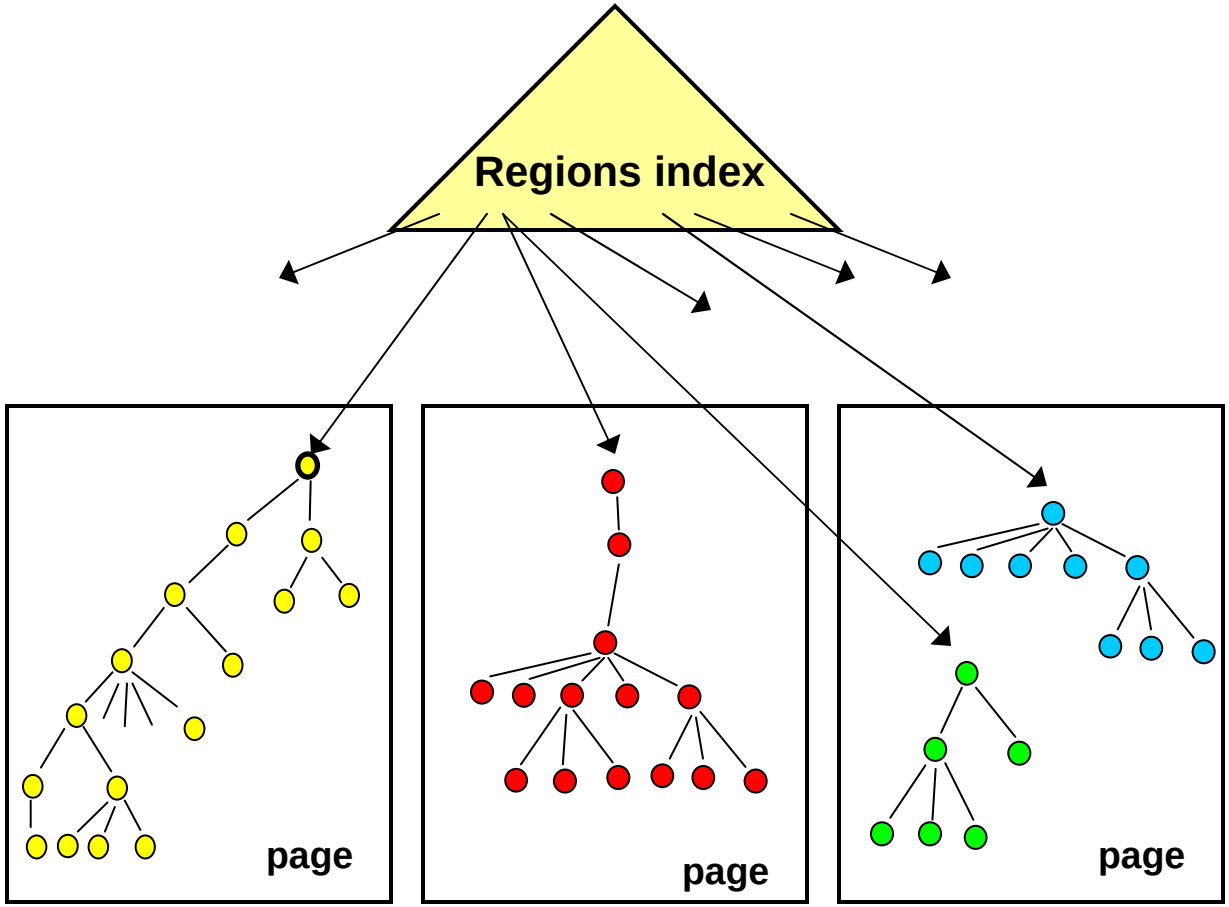
Example:



Document split into 3 regions, stored on 3 pages

XML Storage: “Regions Index”

- not user defined, default component of XML storage layer
- efficient sub-document level access and partial document retrieval



- maps nodeIDs to regions & pages
- allows to fetch required regions instead of full documents
- allows intelligent prefetching

XML Indexes for High Query Performance

- **Define 0, 1 or multiple XML indexes per XML column**
- **Index any elements or attributes, incl. mixed content**
- **Index definition uses an XML pattern to specify which elements/attributes to index (and which not to)**
- **Can index all elements/attributes, but not forced to do so**
- **Can index repeating elements, i.e. which occur more than once per document**
- **0 , 1 or multiple index entries per document**
- **New XML-specific join and query evaluation methods, evaluate multiple predicates concurrently with minimal index I/O**

xmlpattern = XPath without predicates, only child axis (/) and descendent-or-self axis (//)

XML Indexing: Examples

```
<dept bldg=101>
   <employee id=901>
        <name>John Doe</name>
        <phone>408 555 1212</phone>
        <office>344</office>
    </employee>
    <employee id=902>
        <name>Peter Pan</name>
        <phone>408 555 9918</phone>
        <office>216</office>
    </employee>
</dept>
```

```
create table dept(deptID char(8) primary key,  deptdoc xml);
```

```
create index idx1 on dept(deptdoc) generate key
using xmlpattern '/dept/@bldg' as sql double;

create unique index idx2 on dept(deptdoc) generate key
using xmlpattern '/dept/employee/@id' as sql double;

create index idx3 on dept(deptdoc) generate key
using xmlpattern '/dept/employee/name' as sql varchar(35);
```

```
…xmlpattern '//name' as sql varchar(35);          (Index on ALL “name” elements)
…xmlpattern '//@*' as sql double;                 (Index on ALL numeric attributes)
…xmlpattern '//text()‘ as sql varchar(hashed);    (Index on ALL text nodes, hash code)
…xmlpattern ‘/dept//name' as sql varchar(35);
```

```
…xmlpattern '/dept/employee/*/text()' as sql varchar(128);  (All text nodes under employee)
```

```
…xmlpattern 'declare namespace m="http://www.myself.com/";  /m:dept/m:employee/m:name‘
                                                              as sql varchar(45);
```

Querying XML Data in DB2

The following options are supported:

- XQuery/XPath as a stand-alone language
- SQL embedded in XQuery
- XQuery/XPath embedded in SQL/XML
- Plain SQL for full-document retrieval

Example: XQuery as a stand-alone Language

```
create table dept(deptID char(8) primary key,  deptdoc xml);
```

```
XQUERY for  $d in db2-fn:xmlcolumn('dept.deptdoc')/dept
let $emp := $d//employee/name
where $d/@bldg = > 95
order by $d/@bldg
return   <EmpList>
                  {$d/@bldg, $emp}
          </EmpList>
```

- **F**OR: iterates through a sequence, binds variable to items
- **L**ET: binds a variable to a whole sequence of items
- **W**HERE: eliminates items of the iteration
- **O**RDER: reorders items of the iteration
- **R**ETURN: constructs query results

```
<dept bldg=101>
   <employee id=901>
        <name>John Doe</name>
        <phone>408 555 1212</phone>
        <office>344</office>
    </employee>
    <employee id=902>
        <name>Peter Pan</name>
        <phone>408 555 9918</phone>
        <office>216</office>
    </employee>
</dept>
```

Example: SQL embedded in XQuery

```
create table dept(deptID char(8) primary key,  deptdoc xml);
```

- **Identify XML data by a SELECT statement**
- **Leverage predicates/indexes on relational columns**

```
for $d in db2-fn:sqlquery('select deptdoc from dept                    (single document)
                           where deptID = “PR27” ')…

for $d in db2-fn:sqlquery('select deptdoc from dept                    (some documents)
                           where deptID LIKE “PR%” ')…

for $d in db2-fn:sqlquery('select dept.deptdoc from dept, unit         (some documents)
                           where dept.deptID=unit.ID
                           and  unit.headcount > 200’)…..
```

Example: XQuery embedded in SQL/XML

```
create table dept(deptID char(8) primary key,  deptdoc xml);


select deptID,
        xmlquery('for $i in $d/dept
                  let $j := $i//name
                  return $j'   passing deptdoc as "d")
from dept
where  deptID LIKE "PR%"
  and  xmlexists('$d/dept[@bdlg = 101]' passing deptdoc as "d")
```

Other Features in DB2 native XML

- XML Schema Repository
- Schema validation
- Full SQL/XML support
- XML Import/Export
- XML Type in Stored Procedures
- API Extensions (JDBC, CLI, .NET, etc.)
- Visual XQuery Builder
- Annotated schema shredding
- …and more

Summary

- CLOB and shredded XML storage restrict performance and flexibility
- Solution: new **native** XML support in DB2
- High Performance through
  - Hierarchical & parsed XML representation at all layers
  - Path-specific XML Indexing
  - New XML join and query methods
- Flexibility through:
  - Integration of SQL and XQuery
  - Schemas are optional, per document, not per column
  - Zero, one, or many XML schemas per XML column

Questions?

mnicola@us.ibm.com

IBM developerWorks – XML & DB2

- Technical resources for XML and DB2 including training, technical library, product information downloads, support, forums, blogs, and more
- The developerWorks XML zone contains literally hundreds of articles, tutorials, and tips to help a developer make the most of XML-related applications

ibm.com/developerworks/xml

- developerWorks DB2 Training and Certification Center offers a wide variety of resources to help you get where you need to go as a database developer

ibm.com/developerworks/db2

Events

IBM DM Technical Conference – September 12, 2005

http://www-304.ibm.com/jct03001c/services/learning/ites.wss?pageType=page&c=a0000713

XML 2005 Conference

http://2005.xmlconference.org/

IBM Rational Software Development Platform webcasts

- Library of 2005 webcast series with over 30 titles available for replay on demand: ibm.com/developerworks/offers/lp/wc/

IBM developerWorks Live! Technical Briefings

- Live, worldwide events available at no cost
- 11 titles covering PPM, the IBM Rational SDP, SOA, Linux, WebSphere Integration & Infrastructure, IBM Workplace, and more! ibm.com/developerWorks/offers/techbriefings

Resources

IBM developerWorks Tutorials and Training

- Tutorials and training ibm.com/developerworks/training

Downloads

- Easy access to IBM trial software: ibm.com/developerWorks/downloads

Also available: **2005 developerWorks Software Evaluation Kit**

- Over 14GB of the latest trial software, both development and testing tools as well as middleware on DVD and available to you at no charge! ibm.com/developerWorks/offers/sek

To learn more about DB2 Viper XML technology, including technical whitepapers, please visit:
www.ibm.com/software/data/db2/udb/viper

Question & Answer Session

- Be sure to click on "Refresh Q&A" button often to see the new questions and answers
- No need to submit your questions more than once – Your question will not be posted to this page until it is answered

For more information or to submit additional questions after the Q&A session, please email Matthias at:
mnicola@us.ibm.com

Thank you for participating!