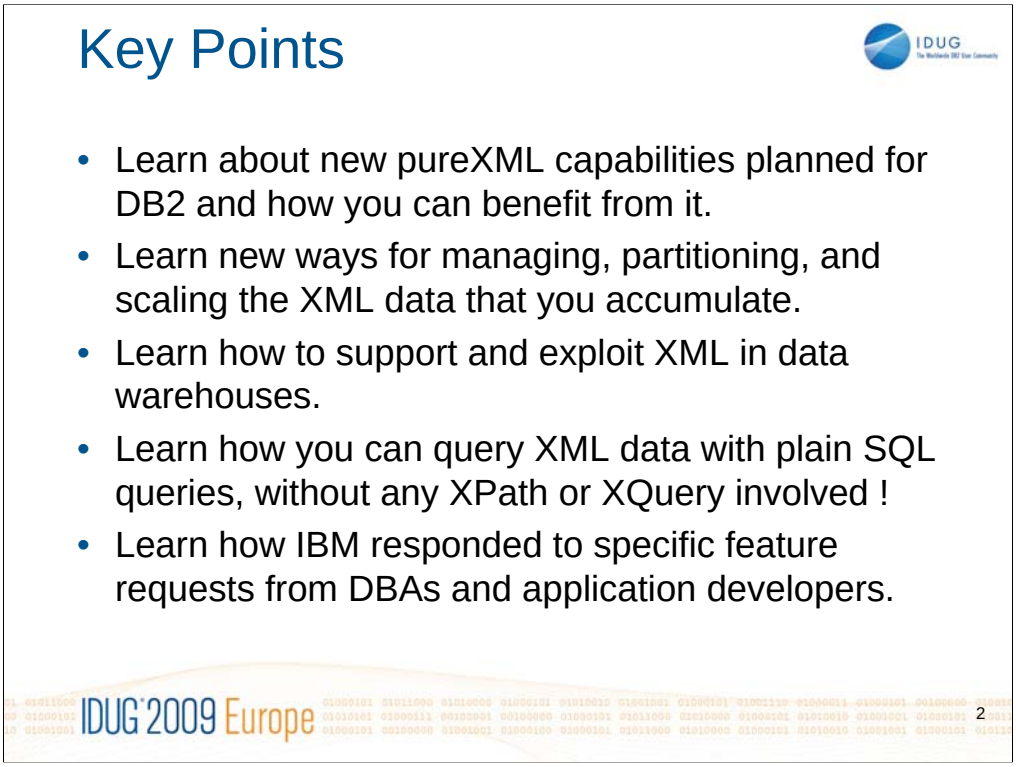
Abstract:

This presentation provides a technical overview of new DB2 pureXML capabilities, which enable easier management, scaling, and warehousing of XML data in DB2. This session walks you through the new XML functionality and internal architectural enhancements in DB2 9.7. Learn about XML in partitioned databases and partitioned or clustered tables. Understand the new compression support for all your XML data and XML indexes. Find out how application development can get easier with XML support in UDFs and new support for relational views over XML data. New admin functions and utility features complete the picture for the DBA.

Speaker:

Matthias Nicola is a senior software engineer at IBM's Silicon Valley Lab, in San Jose, CA, USA. He focuses on all aspects of XML in DB2. Matthias works closely with the DB2 pureXML development teams as well as with customers and business partners to help them design, optimize and implement XML solutions. Previously Matthias worked on data warehouse performance with Informix Software. (www.matthiasnicola.de)

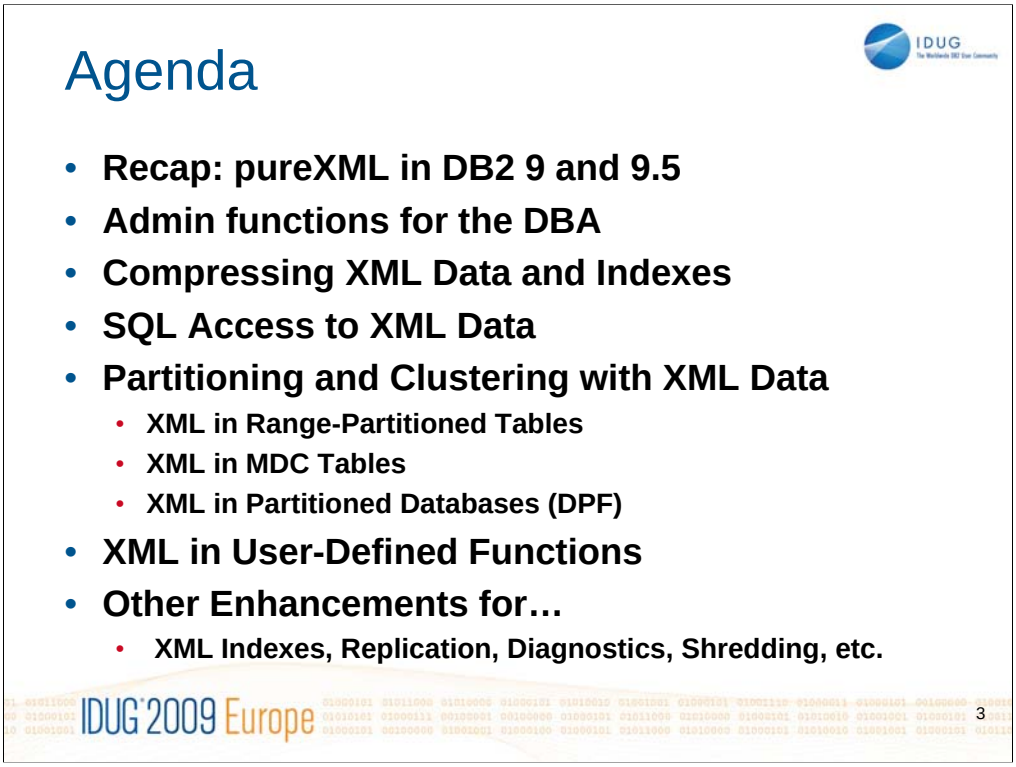


Learn about new pureXML capabilities planned for DB2 and how you can benefit from it.
Learn new ways for managing, partitioning, and scaling the XML data that you accumulate.

Learn how to support and exploit XML in data warehouses.

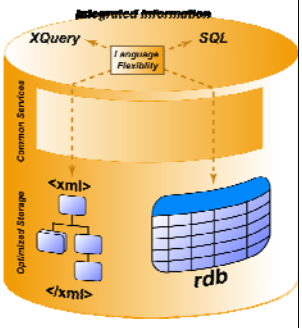
Learn how you can query XML data with plain SQL queries, without any XPath or XQuery involved !

Learn how IBM responds to specific feature requests from DBAs and application developers.



Let's first recap the basics of pureXML in DB2 9.1 and DB2 9.5, and then go through the new XML features in 9.7, listed in this agenda.

Recap: pureXML in DB2 9.1 and 9.5




```
create table customer (cid integer, info XML)

insert into customer (cid, info) values (?,?)

select cid, info from customer

select xmlquery('$INFO/customer/name')
from customer
where cid > 1234 and
xmlexists('$INFO/customer/addr[zip = 95123]')

xquery for $i in db2-fn:xmlcolumn("CUSTOMER.INFO")/customer
where $i/addr/zip = 95123
return <myresult>{$i/name}</myresult>
```


For more details, see:

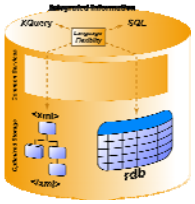
http://www.ibmpressbooks.com/cookbook

http://www.vldb2005.org/program/paper/thu/p1164-nicola.pdf

http://www.ibm.com/developerworks/db2/library/techarticle/dm-0606nicola/

Recap: pureXML in DB2 9.1 and 9.5

```
create index idx1 on customer(info) generate key using
xmlpattern '/customer/addr/zip' as sql varchar(5)

update customer
set info = ?
where ….

update customer
set info = xmlquery(‘copy $new := $INFO
        modify do replace value of $new/customer/addr/zip
                with 95141
        return  $new')
where …;
```



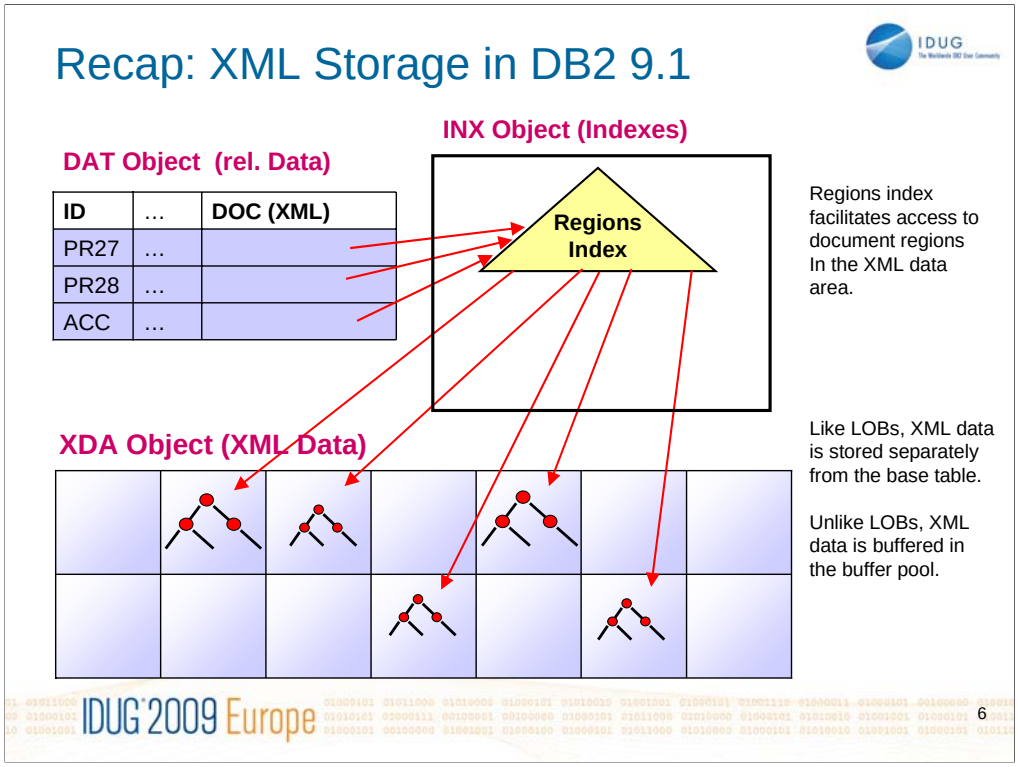
Plus: XML Schema Support, Utilities, Shredding, XSLT, etc.

For more details see:

http://www.ibm.com/developerworks/db2/library/techarticle/dm-0710nicola/

http://www.ibm.com/developerworks/db2/library/techarticle/dm-0708nicola/

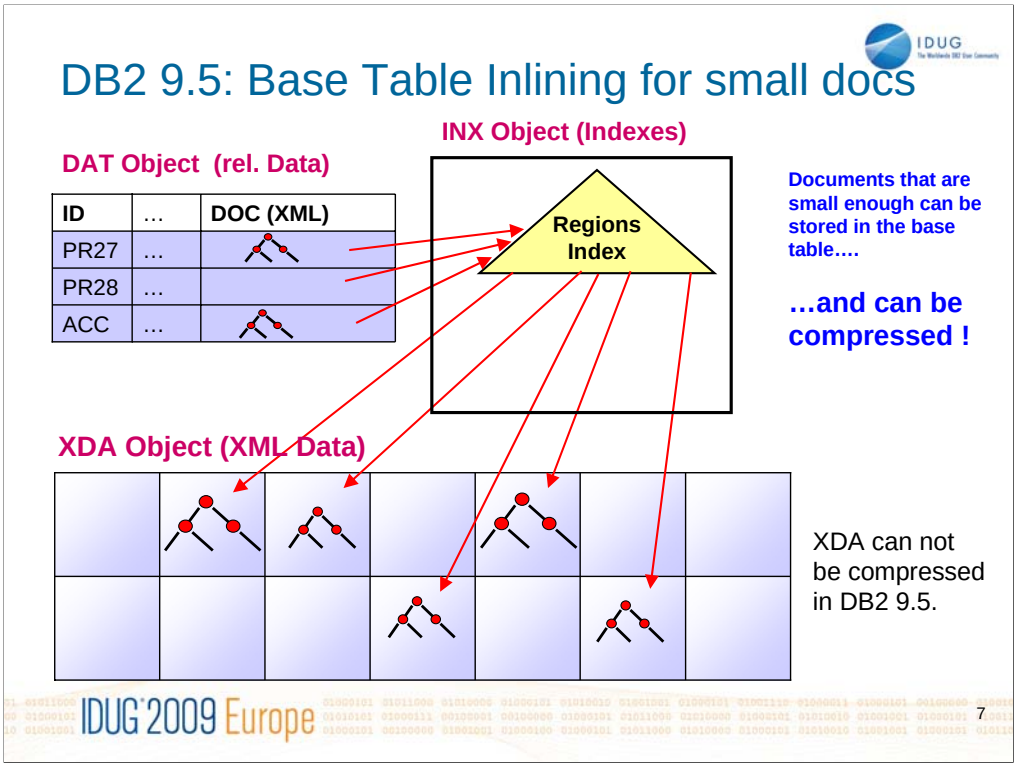
http://www.ibm.com/developerworks/db2/library/techarticle/dm-0709nicola/



If an XML document is too large to fit on a single page in a table space, DB2 splits the document into multiple regions which are then stored on multiple pages. This is transparent to your application and allows DB2 to handle XML documents up to the bind-in limit of 2GB per document.

The DAT object holds the base table rows. The descriptor in the XML column uses a logical reference to the XML data. The reference needs to be resolved into a physical address using the XML Region index. Both the XML Region Index and the XML Path Index are contained in the INX object. The XML Region Index points into the XML pages located in the XDA object.

The information on this slide applies to Db2 9 GA. DB2 9.5 offer several improvements and optimizations of the XML storage architecture, such as inlining and compression.

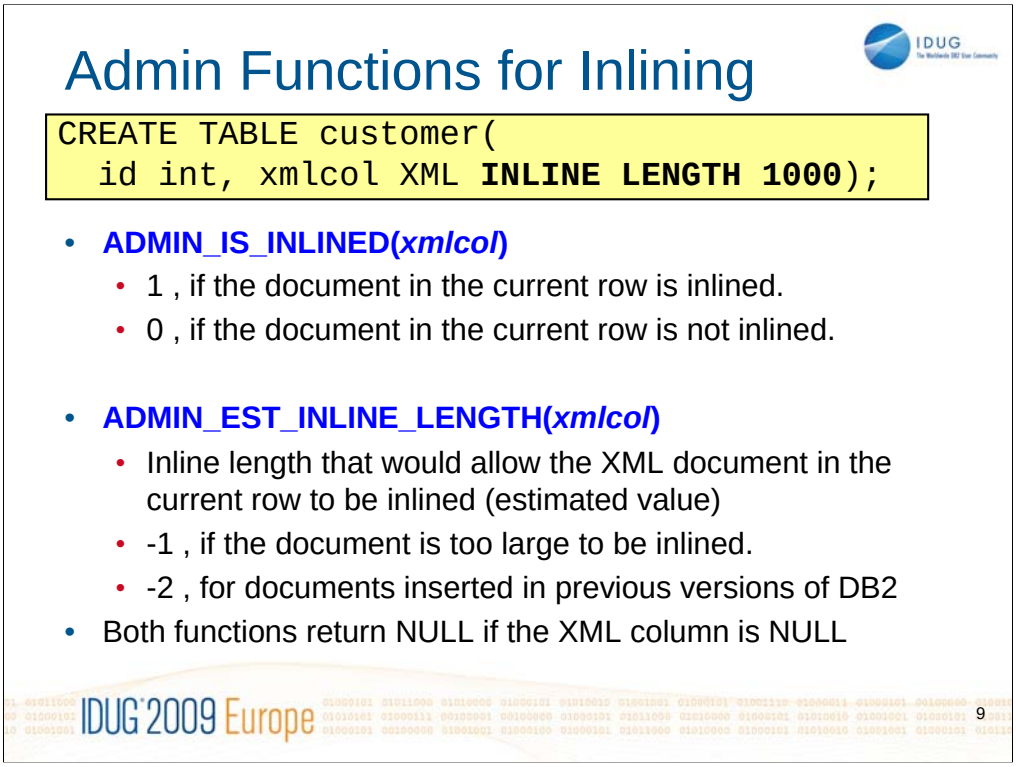


Here is how you inline and compress XML documents in DB2 9.5:

create database departments pagesize 16 K;
create table dept (deptID char(8),...,deptdoc XML inline length 10240);
alter table dept compress yes;

Agenda

- Recap: pureXML in DB2 9 and 9.5
- **Admin functions for the DBA**
- Compressing XML Data and Indexes
- SQL Access to XML Data
- Partitioning and Clustering with XML Data
  - XML in Range-Partitioned Tables
  - XML in MDC Tables
  - XML in Partitioned Databases (DPF)
- XML in User-Defined Functions
- Other Enhancements for…
  - XML Indexes, Replication, Diagnostics, Shredding, etc.

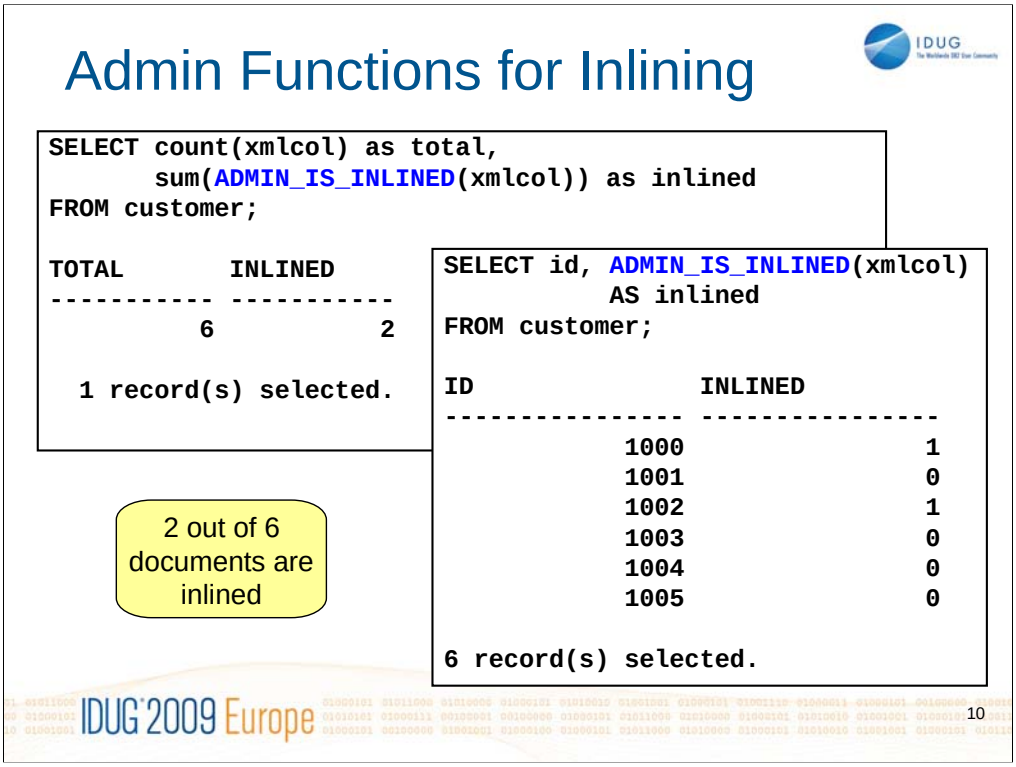


Common questions from many DBAs:

•How do I know whether a certain document is inlined or not?

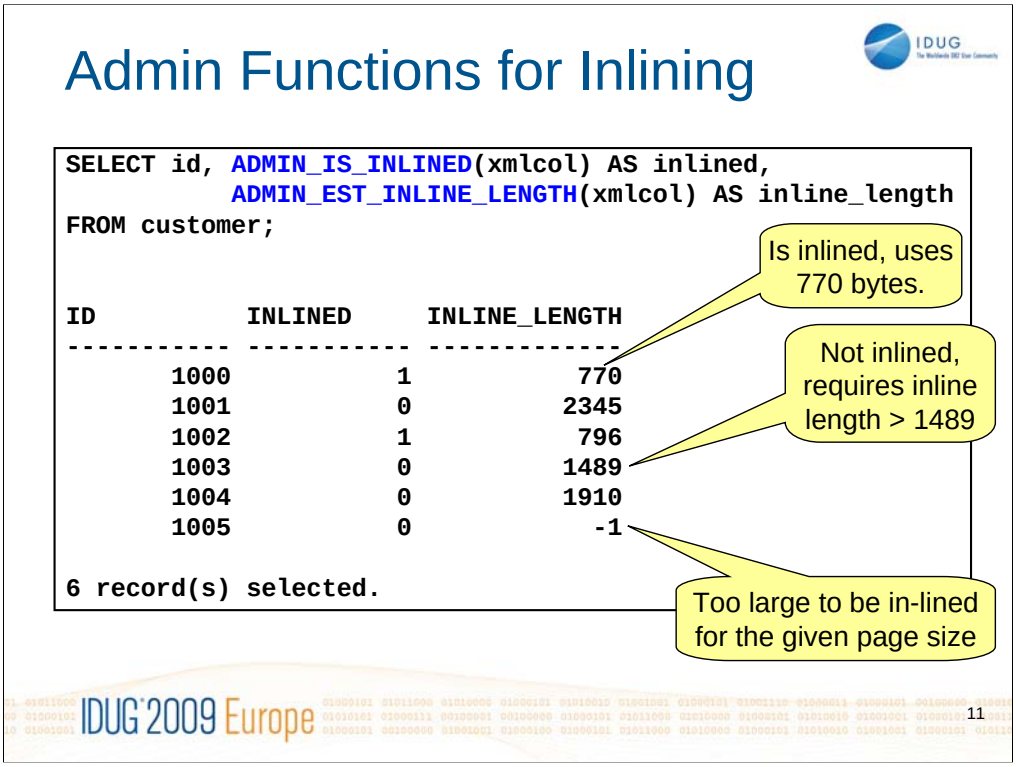
•How do I know which inline length to use so that most of my documents will be inlined.

These new functions help you answer these questions. These functions are in direct response to requests from DBAs.



The function ADMIN_IS_INLINED takes an XML column name as input and returns 1 if the document is inlined, otherwise. 0.

Of course you can wrap functions such as SUM or AVG around the function ADMIN_IS_INLINED in order to obtain aggregated information.



In this example, the query result shows that all documents can be inlined if you set the inline length larger than 2345 bytes.

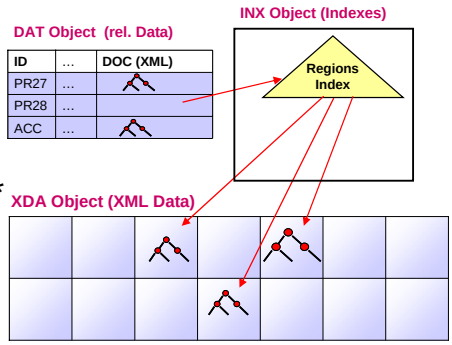
Agenda

- Recap: pureXML in DB2 9 and 9.5
- Admin functions for the DBA
- **Compressing XML Data and Indexes**
- SQL Access to XML Data
- Partitioning and Clustering with XML Data
  - XML in Range-Partitioned Tables
  - XML in MDC Tables
  - XML in Partitioned Databases (DPF)
- XML in User-Defined Functions
- Other Enhancements for…
  - XML Indexes, Replication, Diagnostics, Shredding, etc.

XML Data & Index Compression

```
CREATE TABLE customer(id int, xmlcol XML)
COMPRESS YES;
```

- Compression of DAT Object
- Compression of XDA Object*
- Separate compression dictionaries for DAT and XDA*
- Compression of any user-defined index*
- No compression of regions index or MDC block indexes

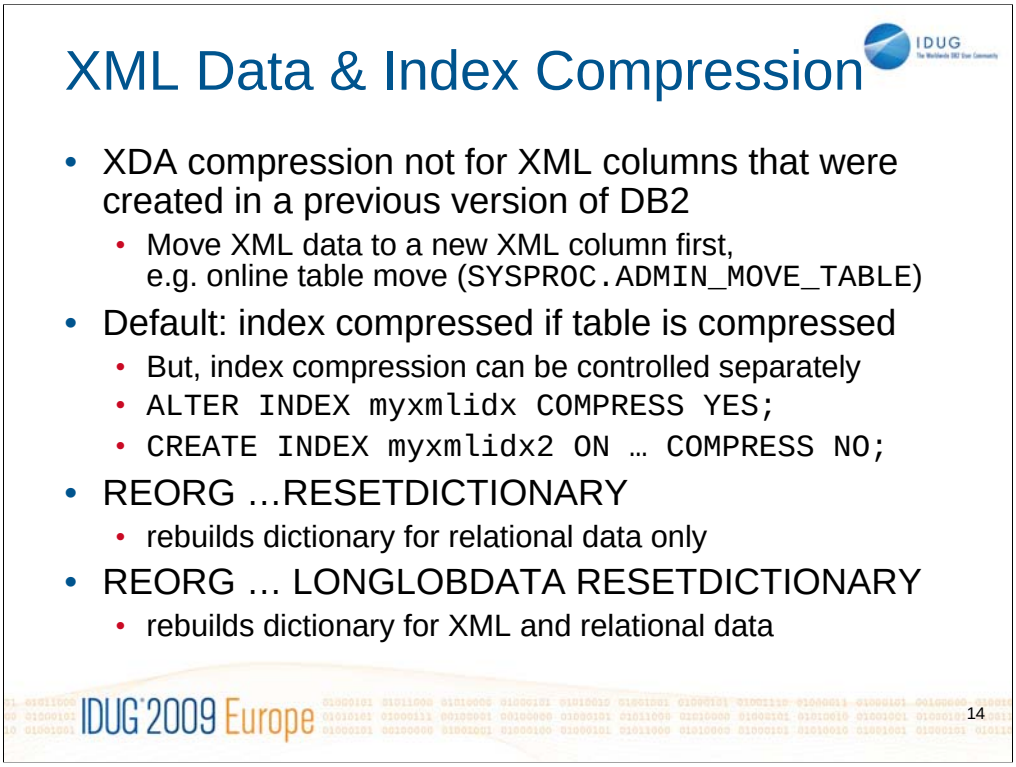


*see next slide for details

New in DB2 9.7 is:

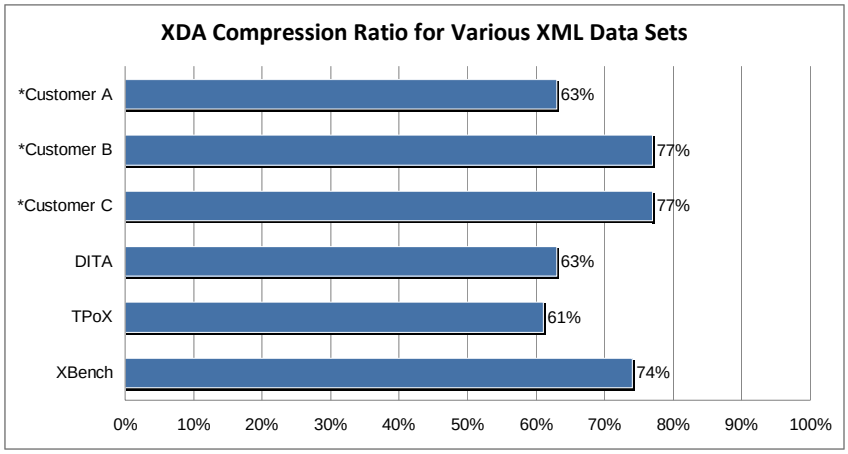
-the compression of the XDA object

-the compression of any user-defined indexes



The compression of the XDA object is only possible for XML columns that were created in DB2 9.7. If you migrate tables with XML columns from DB2 9.1 or 9.5 to DB2 9.7, and if you want to use XDA compression, you must recreate the table and move the XML documents from the old to the new table.

XDA Compression saves 60% to 80% of the storage space



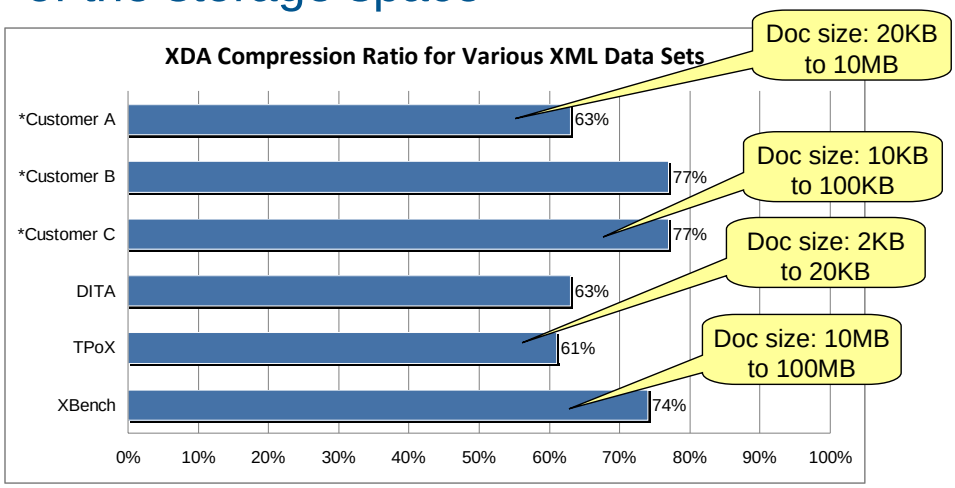
* data sets from DB2 customers

We tested XDA compression with data sets from existing DB2 customers as well as with XML benchmark data.

DITA is an XML document format for content, esp. for authoring, producing, and delivering technical information such as in product manuals.

(see: http://en.wikipedia.org/wiki/Darwin_Information_Typing_Architecture)

XDA Compression saves 60% to 80% of the storage space

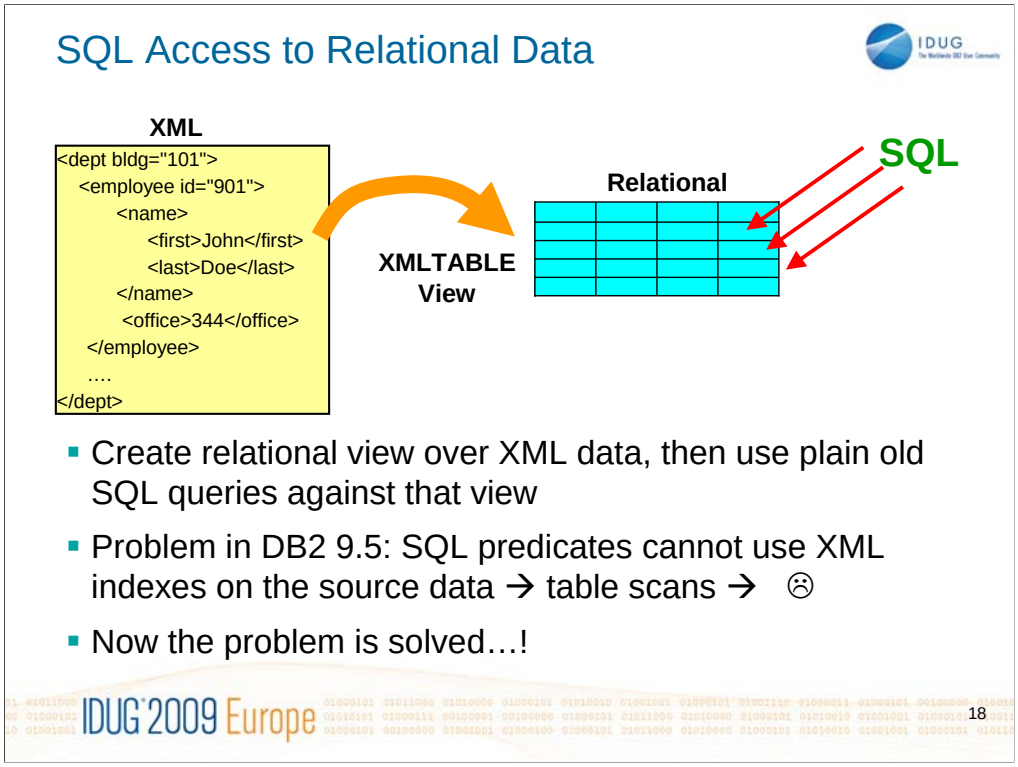


* data sets from DB2 customers

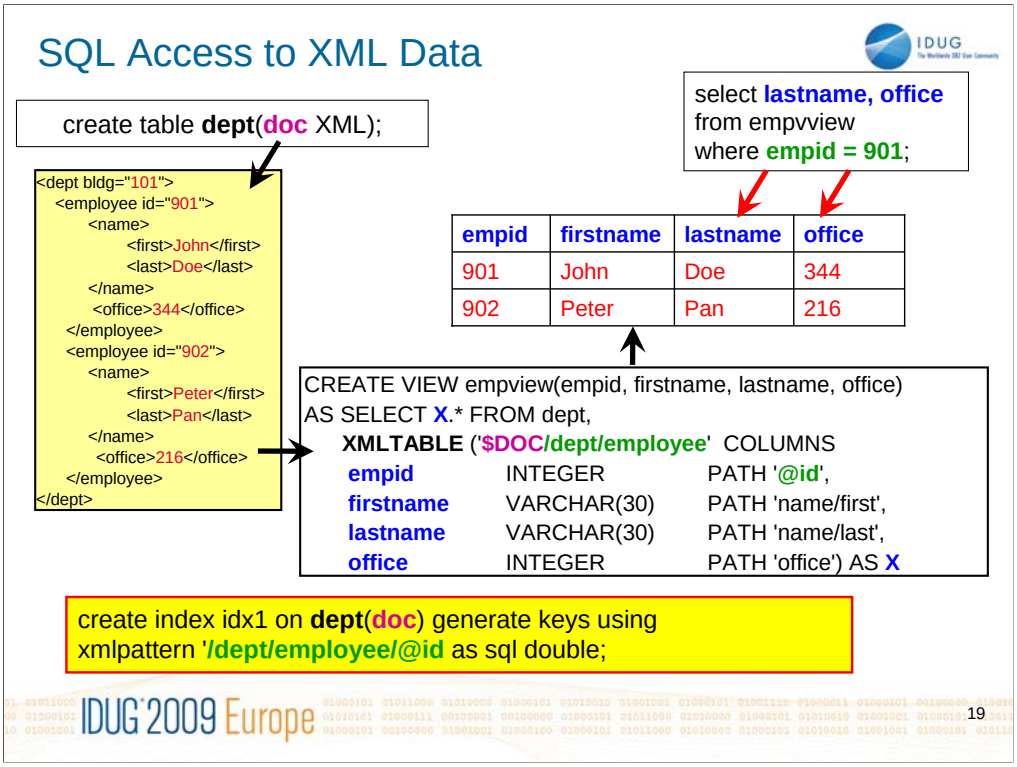
Note that the compression ratio does not depend on your average XML document size !

Agenda

- Recap: pureXML in DB2 9 and 9.5
- Admin functions for the DBA
- Compressing XML Data and Indexes
- **SQL Access to XML Data**
- Partitioning and Clustering with XML Data
  - XML in Range-Partitioned Tables
  - XML in MDC Tables
  - XML in Partitioned Databases (DPF)
- XML in User-Defined Functions
- Other Enhancements for…
  - XML Indexes, Replication, Diagnostics, Shredding, etc.



It can be helpful to use a view to expose XML data in relational format, e.g. to allow legacy SQL application access to the XML data.



The XMLTABLE function is used in the *FROM* clause of the *SELECT* statement together with the table dept that it operates on. The XMLTABLE function is implicitly joined with the table dept and applied to each of its rows.

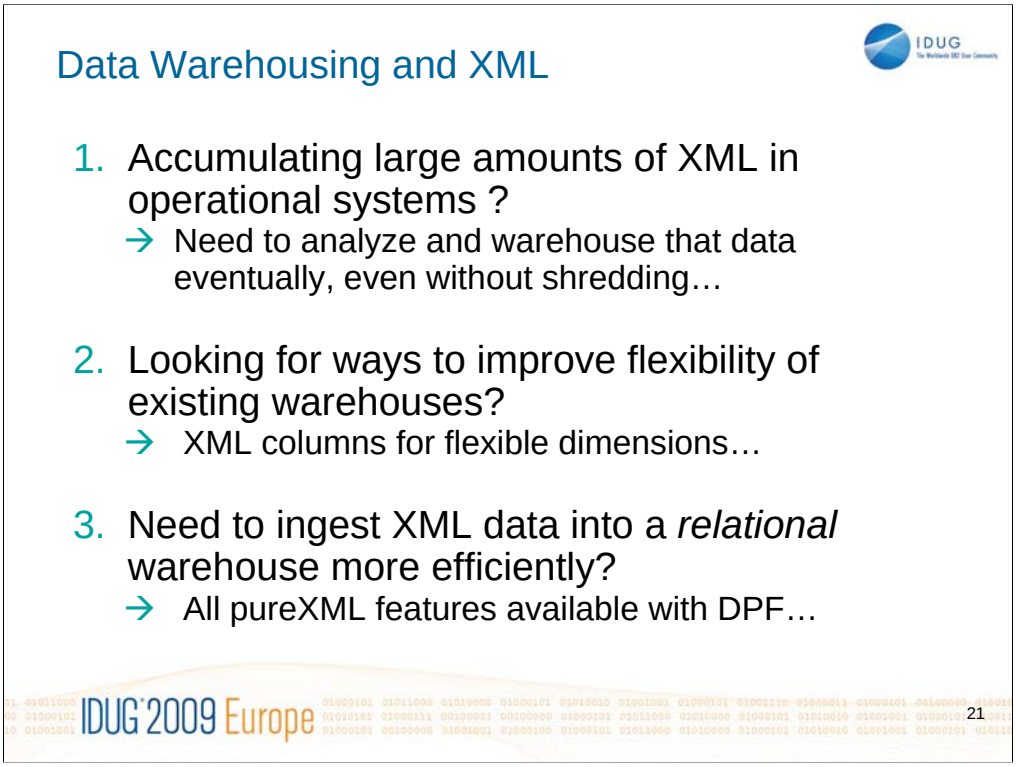
The XMLTABLE function contains a *row-generating* XQuery expression and, in the COLUMNS clause, one or multiple *column-generating* expressions. The row-generating expression is the XPath $DOC/dept/employee.

The row-generating expression is applied to each XML document in the XML column and produces one or multiple employee elements (sub-trees) per document. The output of the XMLTABLE function contains one row for each employee element. Hence, the output produced by the row-generating XQuery expression determines the cardinality of the result set of the SELECT statement.

The COLUMNS clause is used to transform XML data into relational data. Each of the entries in this clause defines a column with a column name and a SQL data type. Here, the returned rows have 4 columns named *empID*, *firstname* and *lastname* and *office* of data types integer and varchar(30). The values for each column are extracted from the employee elements and cast to the SQL data types. For example, the path *name/first* is applied to each employee element to obtain the value for the column *firstname*. The row-generating expression provides the context for the column-generating expressions. In other words, you can typically append the column-generating expressions to the row-generating expression to get an intuitive idea of what a given XMLTABLE function returns in its columns.

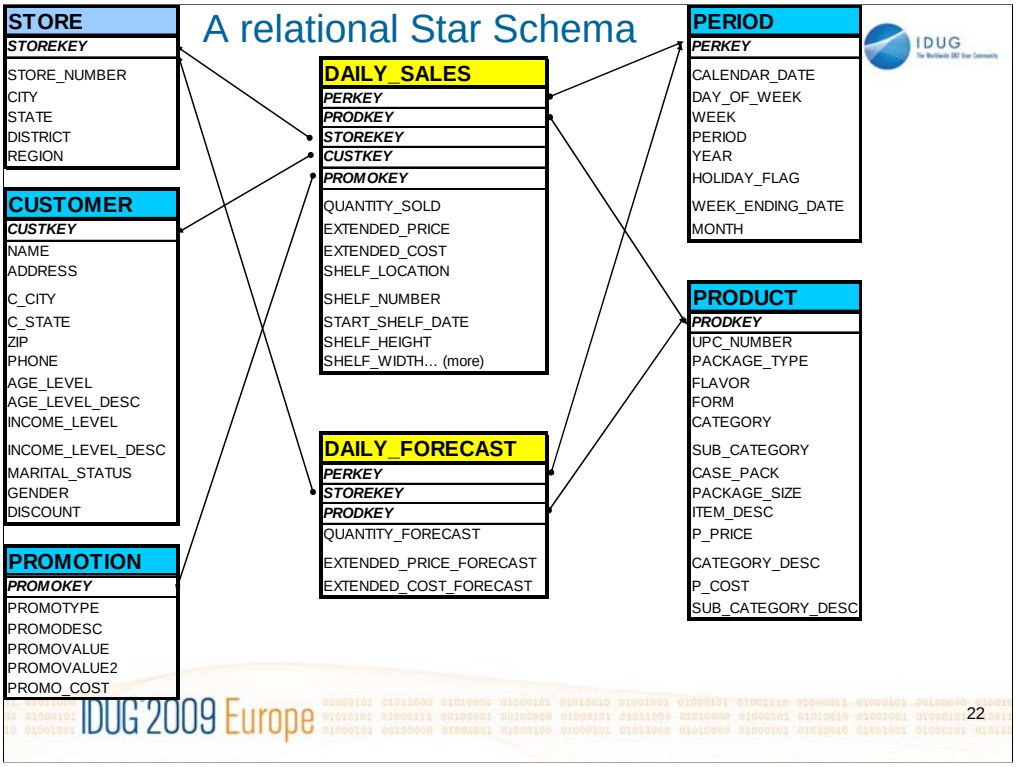
Agenda

- Recap: pureXML in DB2 9 and 9.5
- Admin functions for the DBA
- Compressing XML Data and Indexes
- SQL Access to XML Data
- Partitioning and Clustering with XML Data
  - XML in Range-Partitioned Tables
  - XML in MDC Tables
  - XML in Partitioned Databases (DPF)
- XML in User-Defined Functions
- Other Enhancements for…
  - XML Indexes, Replication, Diagnostics, Shredding, etc.



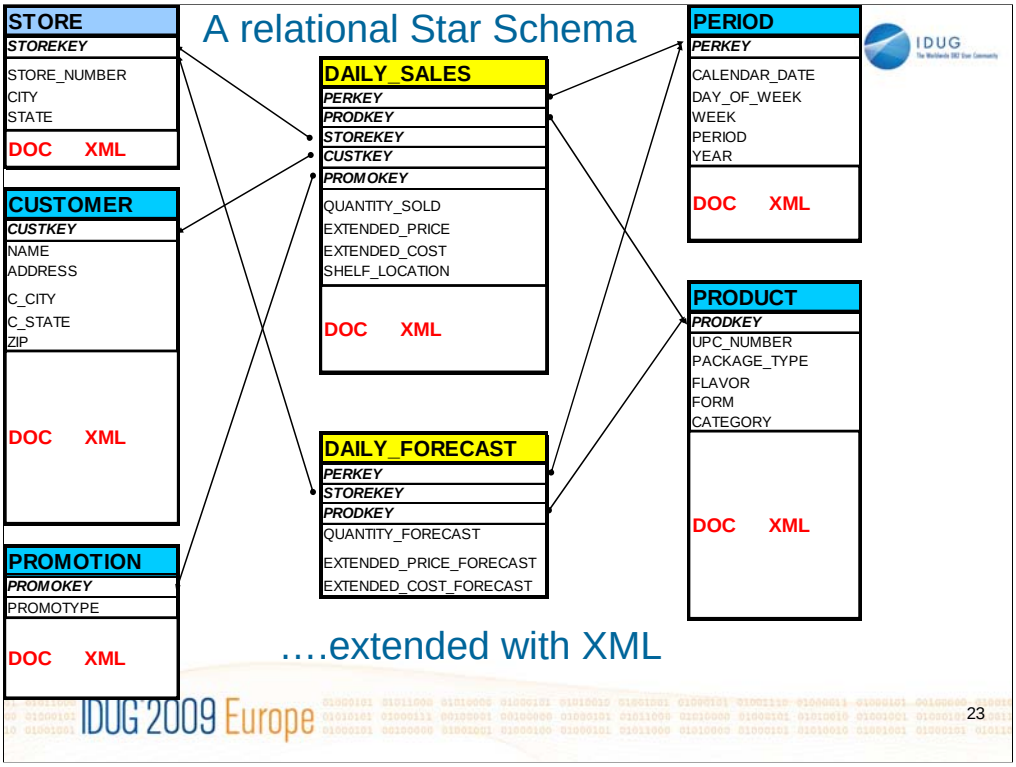
Three reason why you may want to think about data warehousing and XML.

Imagine a dimension table PRODUCT that contains columns for a variety of product characteristics along which you want to slice and dice your sales data in the fact table. As new products are being introduced, new product attributes become interesting for analytics. However, it may not be feasible to add several columns to the dimension table for every new product that gets introduced. Let's look at how XML can help….



Traditional dimension tables in a warehouse

- rigid, hard to change, relational columns only
- hampers change and limits business agility
- can't afford to add columns to dimension tables for every sparsely used attribute of a product or customer etc.



XML Columns allow you to add attributes to products, customers, etc. without having to add columns to your database schema.

XML in DPF, RP, MDC Tables

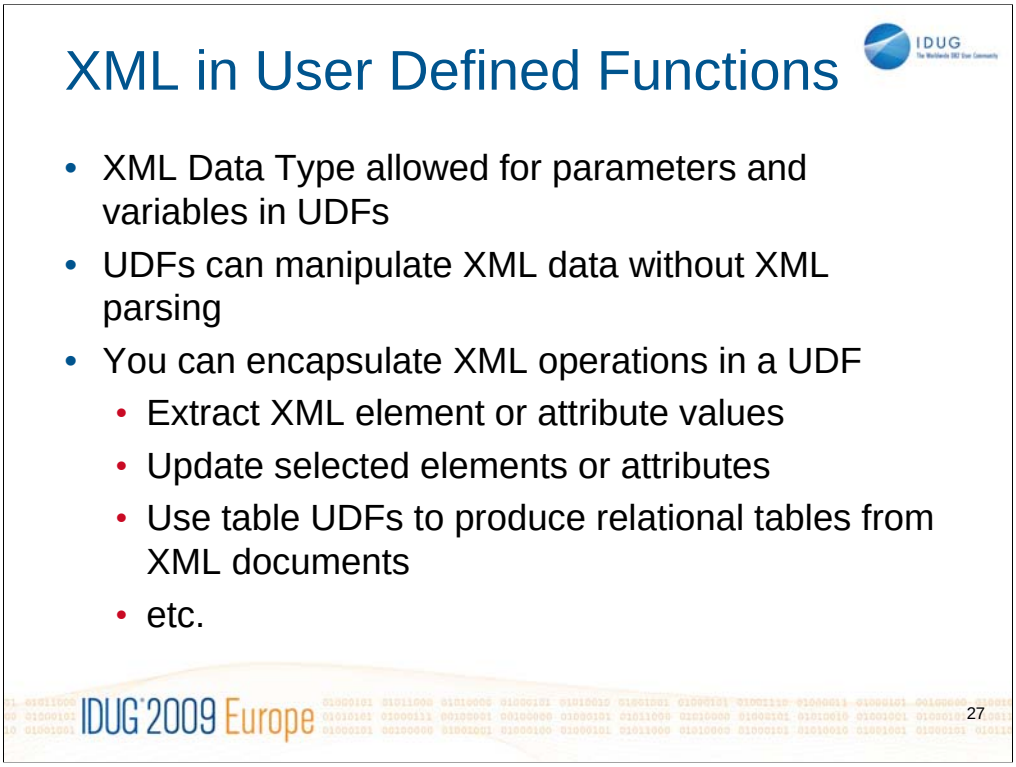
- All of the following have to be relational columns:
  - DPF distribution key
  - Range partitioning key
  - MDC clustering columns
- XML column is payload in DPF, RP, MDC table
- Cannot distribute, partition, or organize by XML values
- Can extract XML values into relational columns, then use those to distribute, partition, or organize

Indexes in MDC and RP Tables

- Both MDC “block” indexes and XML indexes can be used in the same query
  - Index AND-ing plans of block indexes and XML indexes !
- Range partitioned tables:
  - Relational indexes can be local (partitioned) or global (non-partitioned) indexes
  - XML Regions Index is always a local index
  - User-defined XML Indexes are (for now) global indexes

Agenda

- Recap: pureXML in DB2 9 and 9.5
- Admin functions for the DBA
- Compressing XML Data and Indexes
- SQL Access to XML Data
- Partitioning and Clustering with XML Data
  - XML in Range-Partitioned Tables
  - XML in MDC Tables
  - XML in Partitioned Databases (DPF)
- **XML in User-Defined Functions**
- Other Enhancements for…
  - XML Indexes, Replication, Diagnostics, Shredding, etc.



DB2 9.1 and 9.5 already support the XML data type in stored procedures:

http://www.ibm.com/developerworks/data/library/techarticle/dm-0701oliva/

But, the XML data type was so far not supported in UDFs. This support has now been added in DB2 9.7.

Scalar UDF with XML

```
CREATE FUNCTION getname(doc XML)
RETURNS VARCHAR(25)
BEGIN ATOMIC
  RETURN XMLCAST(XMLQUERY('$d/customerinfo/name'
                          PASSING doc AS "d")
              AS VARCHAR(25));
END#
```

```
SELECT getname(info) AS name
FROM customer
WHERE cid = 1002 #

NAME
-------------------------
Jim Noodle

 1 record(s) selected.
```


Hide XML Extraction in a UDF !

Table UDF with XML

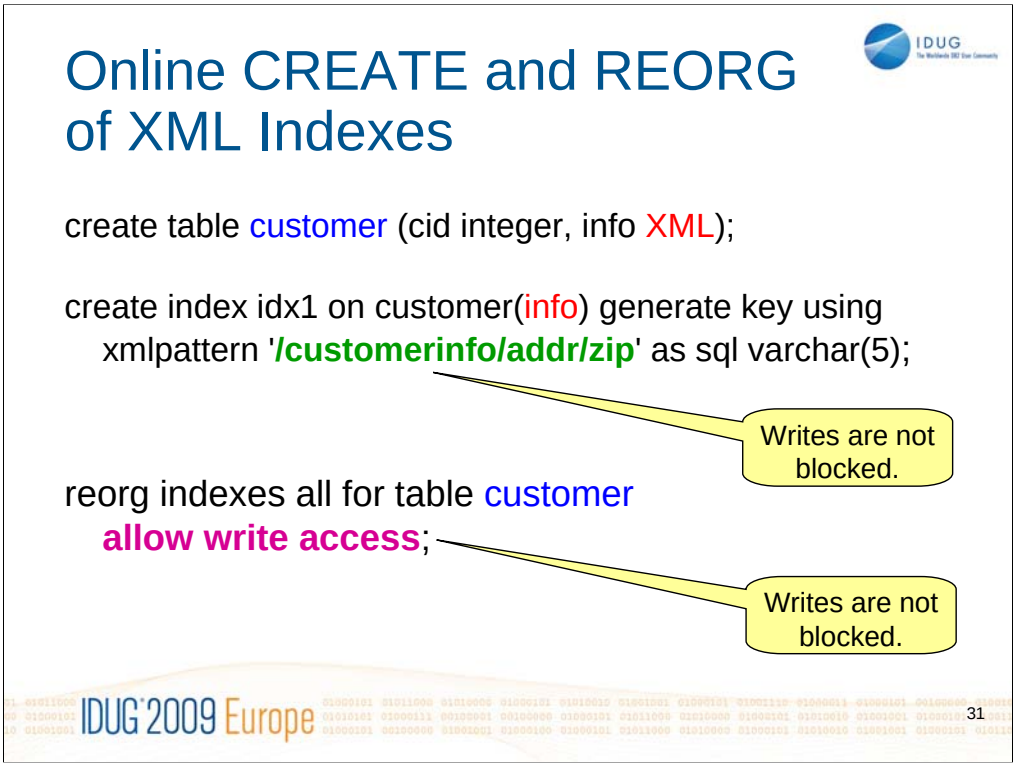
```
CREATE FUNCTION getphone(doc XML)
RETURNS TABLE(type VARCHAR(10), number VARCHAR(20))
BEGIN ATOMIC
  RETURN
    SELECT type, number
    FROM XMLTABLE('$d/customerinfo/phone' PASSING doc AS "d"
           COLUMNS
              type   VARCHAR(10)  PATH '@type',
              number VARCHAR(20)  PATH '.') ;
END #
```

```
SELECT cid, p.type, p.number
FROM customer, TABLE(getphone(info)) p
WHERE cid = 1004 #

CID              TYPE       NUMBER
---------------- ---------- --------------------
            1004 work       905-555-4789
            1004 home       416-555-3376

  2 record(s) selected.
```


Hide even more XML Extraction in a UDF.

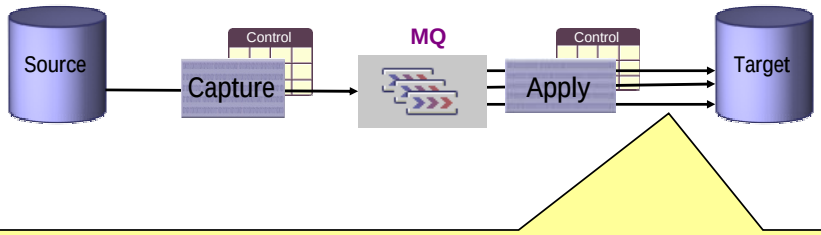
Agenda

- Recap: pureXML in DB2 9 and 9.5
- Admin functions for the DBA
- Compressing XML Data and Indexes
- SQL Access to XML Data
- Partitioning and Clustering with XML Data
  - XML in Range-Partitioned Tables
  - XML in MDC Tables
  - XML in Partitioned Databases (DPF)
- XML in User-Defined Functions
- Other Enhancements for…
  - XML Indexes, Replication, Diagnostics, Shredding, etc.



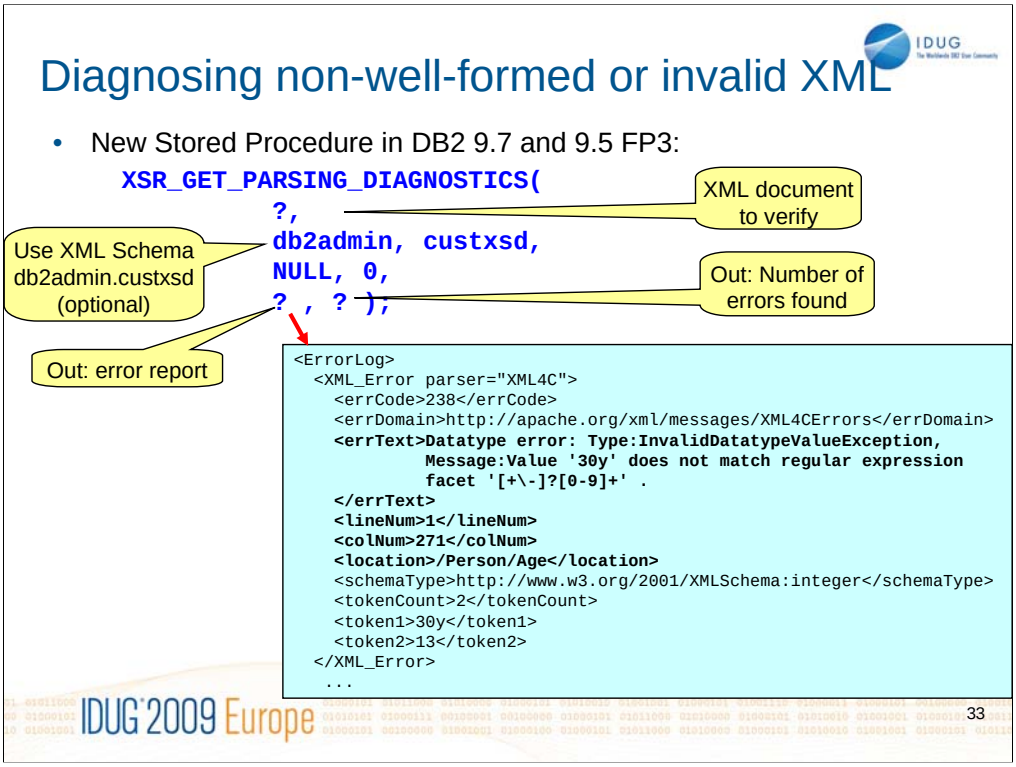
Prior to DB2 9.7, creating or reorg'ing XML indexes almost implied that writes against the table are blocked until the index creation or reorganization is completed. This restriction has now been lifted in DB2 9.7. The "create index" statement is now by default an online operation, as for relational indexes.

Q Replication: Apply with XML Functions



- DB2-to-DB2 replication allows XML functions at "apply"
- Supported functions include:
  - XMLQUERY, XMLVALIDATE, XMLCAST, XMLDOCUMENT, XMLELEMENT, XMLCONCAT,...
- Usage includes:
  - Validate against a different XML Schema at the target
  - Extract document fragments or values at the target (XMLQUERY)
  - Use XQuery Update expression to modify/transform documents

Restrictions: the functions XMLQUERY and XMLVALIDATE are only allowed when replication LUW to LUW or zOS to zOS, but not between zOS and LUW.

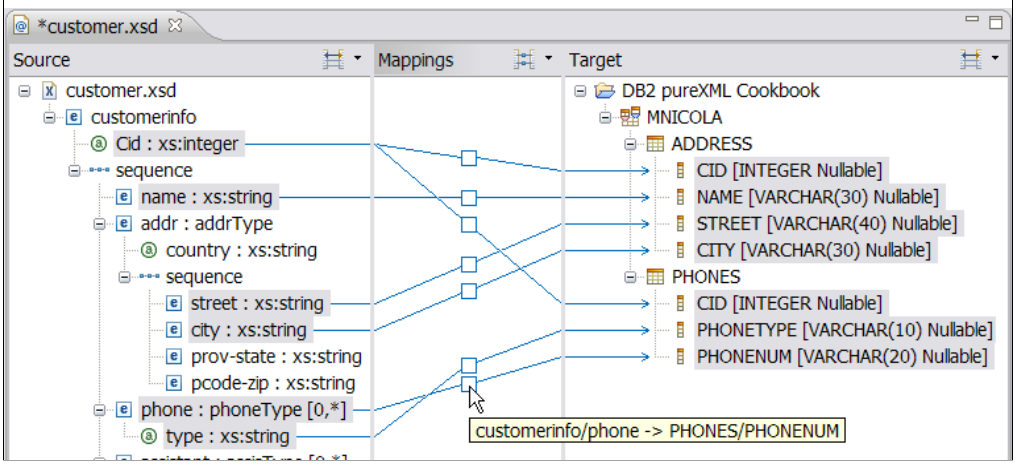


For more details, see:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.xml.doc/doc/r0054517.html

Bulk Decomposition

- Decomposition = Schema-based shredding of XML to relational tables
- Decomp in DB2 9.5: One document at a time, one SP call per document.



The screenshot shows the visual editor for XML-to-relational mappings in IBM Data Studio Developer 2.1. On the left is the XML Schema of the XML data that is to be shredded. On the right are the relational target tables. The target tables "ADDRESS" and "PHONE" belong to the relational schema "MNICOLA". The "DB2 pureXML Cookbook" is simply the name that I chose for the Data Project in IBM Data Studio. It also happens to be the title of a book that I am currently writing (available in July).

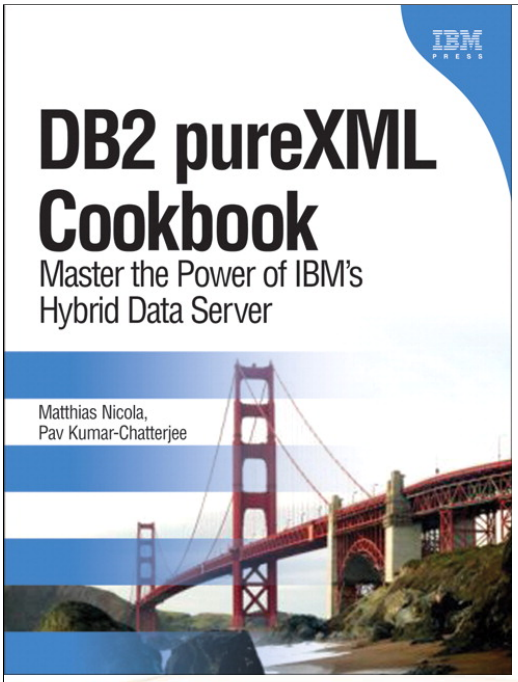
Bulk Decomposition

- New: Shred all document identified by a query
- LOAD first, then use CLP command or SP call to shred:

```
DECOMPOSE XML DOCUMENTS IN
  'SELECT cid, info FROM customer'
  XMLSCHEMA db2admin.customerxsd
  MESSAGES /home/matthias/errorreport.xml ;

CALL XDB_DECOMP_XML_FROM_QUERY (
  'DB2ADMIN', 'CUSTOMERXSD',
  'SELECT cid, info FROM customer ',
  0, 0, 0, NULL, NULL, 1,
  :numInput, :numDecomposed, :errorreportBuf);
```

Summary

- New admin functions to check inlining
- Can Compress all your XML Data and Indexes
- SQL Access to XML Data via XMLTABLE Views
- XML in the Warehouse
  - XML in DPF, MDC, and Range-Partitioned Tables
- XML in User-Defined Functions
- Online CREATE and REORG of XML Indexes
- Customizable Replication
- Improved Diagnostics
- Bulk Shredding



IDUG

- Comprehensive coverage of pureXML in
  DB2 for Linux, UNIX, Windows and
  DB2 for z/OS
- **http://tinyurl.com/pureXML**

Further Reading

- "Enhance business insight and scalability of XML data with new DB2 9.7 pureXML features"
  http://www.ibm.com/developerworks/data/library/techarticle/dm-0904db297purexml/
- "15 best practices for pureXML performance in DB2"
  http://www.ibm.com/developerworks/db2/library/techarticle/dm-0610nicola/
- "pureXML™ in DB2 9: Which way to query your XML Data?"
  http://www.ibm.com/developerworks/db2/library/techarticle/dm-0606nicola/
- "XMLTABLE by Example", Part 1 & 2
  http://www.ibm.com/developerworks/db2/library/techarticle/dm-0708nicola/
  http://www.ibm.com/developerworks/db2/library/techarticle/dm-0709nicola/
- “Update XML in DB2 9.5”:
  http://www.ibm.com/developerworks/db2/library/techarticle/dm-0710nicola/
- DB2 Documentation & Resources:
  http://publib.boulder.ibm.com/infocenter/db2luw/v7r1/index.jsp
  http://www.ibm.com/developerworks/wikis/display/db2xml/Technical+Papers+and+Articles
- Performance of DB2 9 pureXML vs. CLOB and shredded XML storage
  http://www.ibm.com/developerworks/db2/library/techarticle/dm-0612nicola/
- XML Database Benchmark:
  http://tpox.sourceforge.net/ , http://tpox.sourceforge.net/tpoxresults.htm

Speaker:

Matthias Nicola is a senior software engineer at IBM's Silicon Valley Lab, in San Jose, CA, USA. He focuses on all aspects of XML in DB2. Matthias works closely with the DB2 pureXML development teams as well as with customers and business partners to help them design, optimize and implement XML solutions. Previously Matthias worked on data warehouse performance with Informix Software. (www.matthiasnicola.de)