

DB2 for z/OS PureXML Overview and Performance Practices

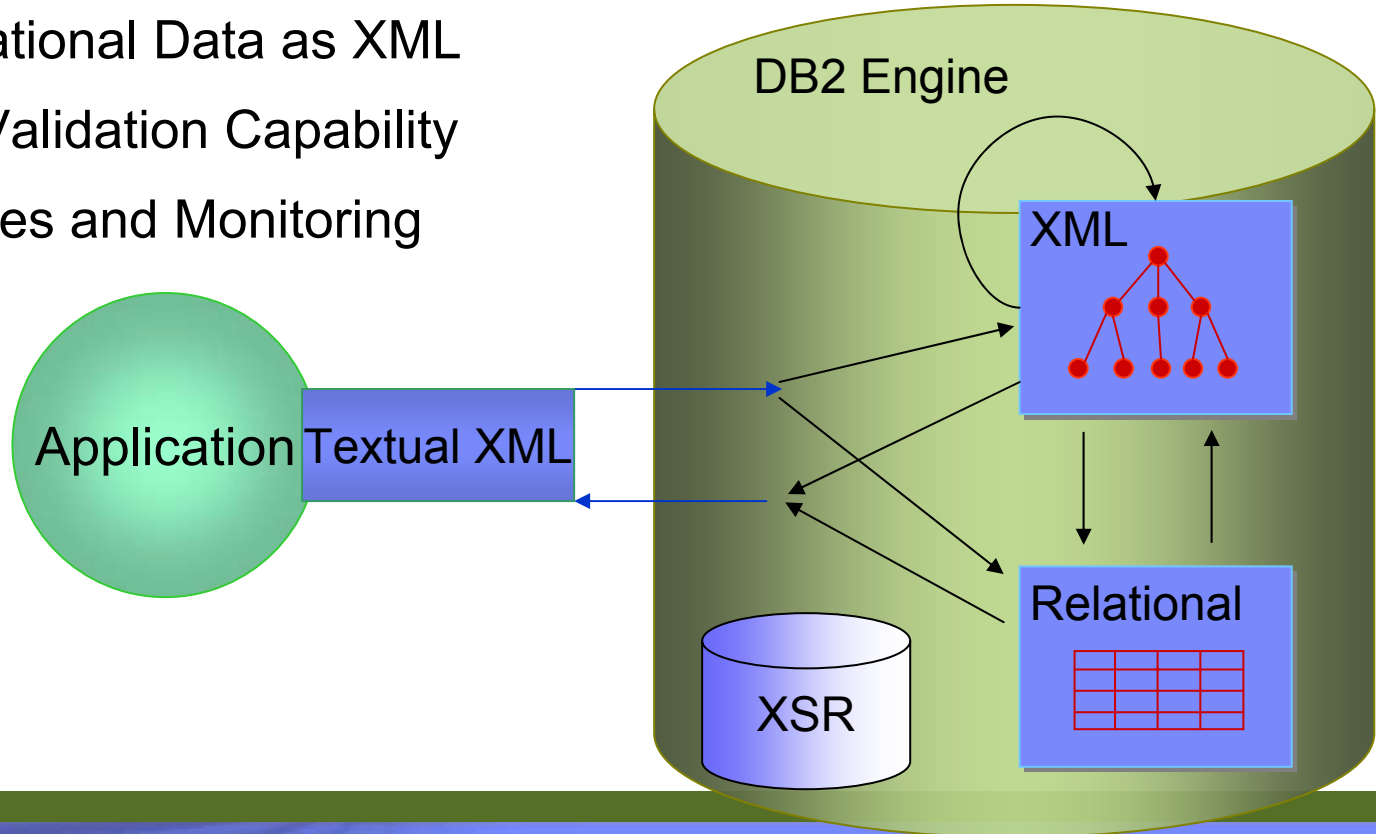
*Akiko Hoshikawa, DB2 for z/OS Performance
Silicon Valley Lab*

Best Performance Practices

- Overview of pureXML support in DB2 9 for z/OS
 - ▶ Quick view of what is XML?
 - ▶ Understand the physical structure of XML support
- Best Practices for XML Performance
 - ▶ Best Practices for Inserting XMLs
 - ▶ Best Practices for XML Queris
 - ▶ Best Practice for Cost Reduction
 - XML workload and zAAP redirection
 - XML workload and zIIP redirection

What is pureXML?

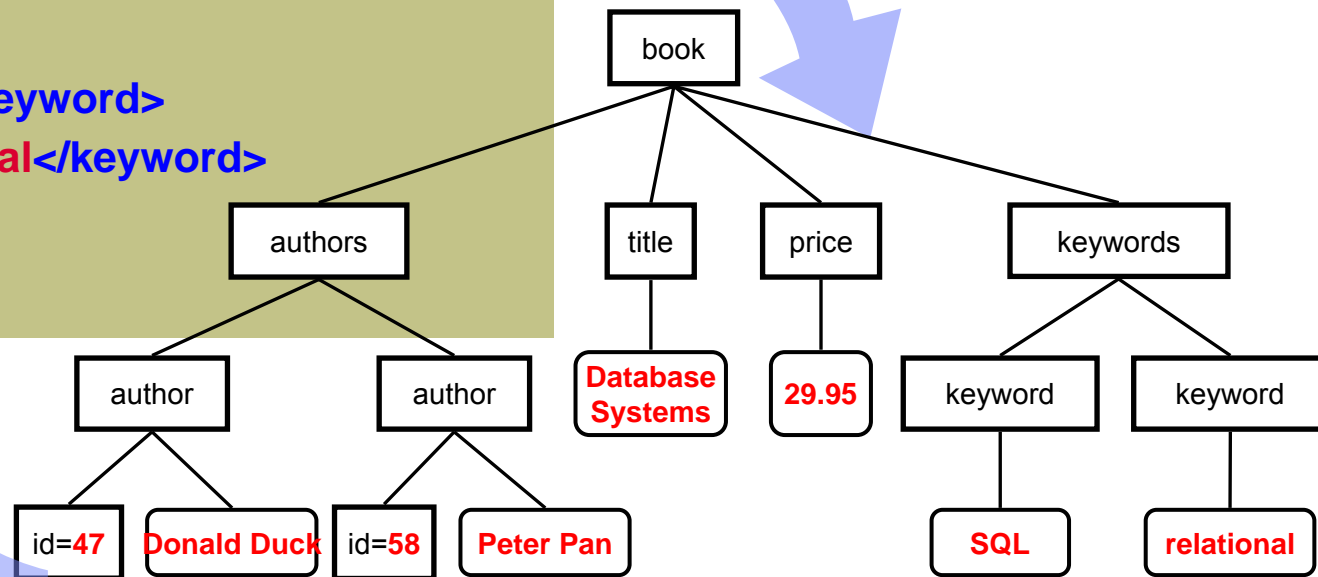
- XML native storage
- Query Capability
- Indexing Inside of XML Documents
- Publishing Relational Data as XML
- XML Schema Validation Capability
- Database Utilities and Monitoring



The XQuery Data Model

```
<book>
  <authors>
    <author id="47">Donald Duck</author>
    <author id="58">Peter Pan</author>
  </authors>
  <title>Database systems</title>
  <price>29.95</price>
  <keywords>
    <keyword>SQL</keyword>
    <keyword>relational</keyword>
  </keywords>
</book>
```

XML Parsing




Serialization

XMLQUERY, XMLEXISTS, XMLTABLE with XPath

- XMLQUERY is best to find a specific part of a specific document.
- XMLEXISTS is used for filtering documents
- XMLTABLE is best to find common parts of potentially many documents or to iterate through the result.


```
SELECT XMLQUERY('/book/price'  
              PASSING book)  
FROM books WHERE  
XMLEXISTS('/book[title="Database systems"]'  
          PASSING book);
```

last	first
Donald	Duck
Peter	Pan



<price>29.95</price>

```
SELECT XT.*  
FROM  
book B,  
XMLTABLE('/book[title="Database  
systems"]/author'  
          PASSING B.book  
          COLUMNS last VARCHAR(50) PATH 'last',  
                   first VARCHAR(50) PATH 'first'  
          ) XT  
ORDER BY XT.last;
```



XML Storage on Mature Infrastructure

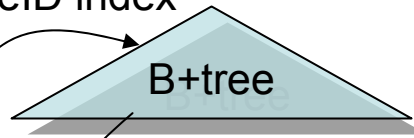
DocID index



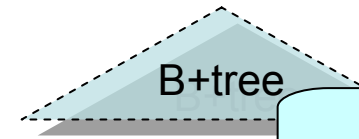
Base Table

DocID	...	XMLCol
1	(DB2_GENERATED_DOCID_FOR_XML)	
2		
3		

NodeID index



XML index (user)



Internal XML Table

DOCID	MIN_NODEID	XMLData
1	02	
2	02	
2		
3	02	

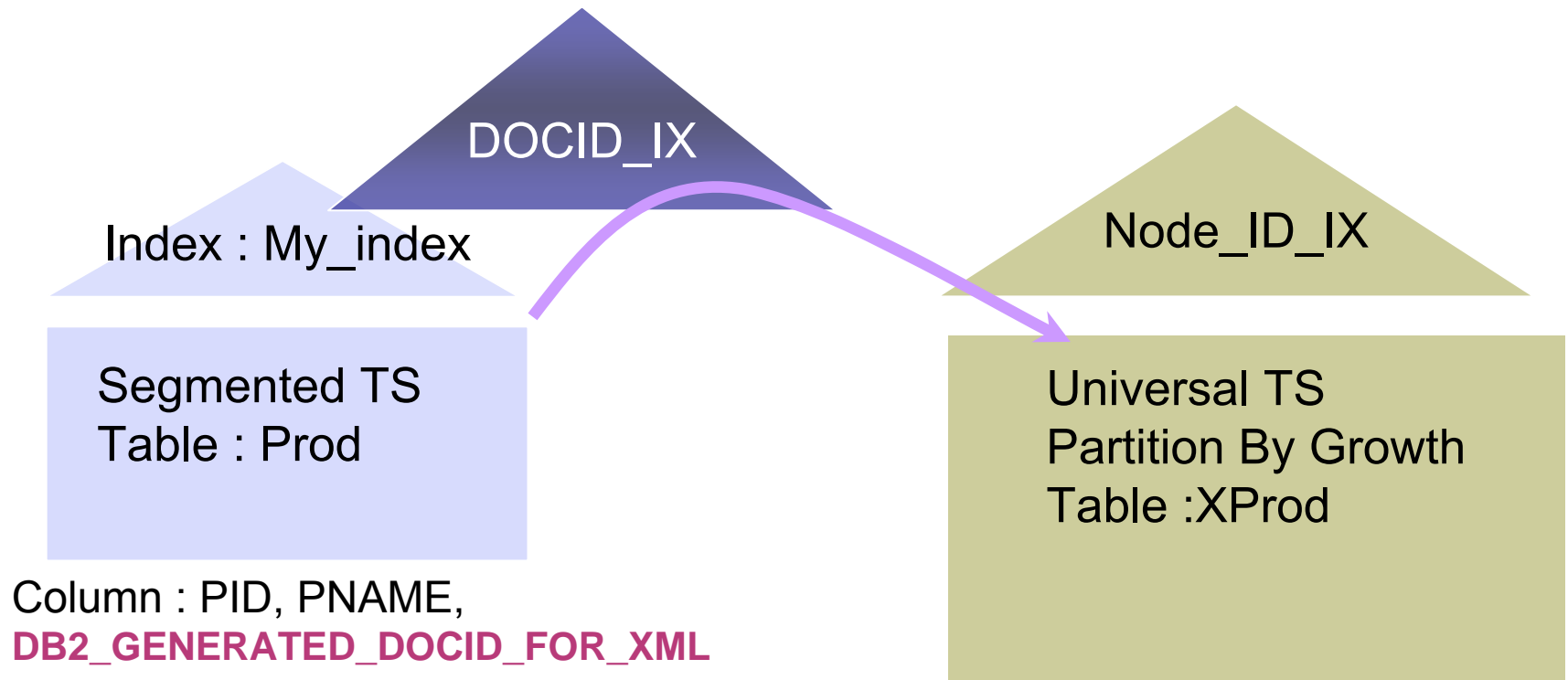
Regular Table space

A table with an XML column has a DocID column, used to link from the base table to the XML table. A DocID index is used for getting to base table rows from XML indexes.

Each XMLData column is a VARBINARY, containing a subtree or a sequence of subtrees, with context path. Rows in XML table are freely movable, linked with a NodeID index.

Create XML column

```
CREATE TABLE PROD(PID INT, PNAME VARCHAR(20), XMLCOL XML);  
CREATE INDEX My_index ON PROD (PID);
```



DBA's Questions on XML table

- What is XML table space? How does it grow?
- What is difference from LOB?
 - ▶ Base : Partitioned or Range Partitioned UTS
 - XML : Range Partition UTS
 - ▶ Base : Segmented or Partitioned By Growth UTS
 - XML : PBG UTS
 - ▶ Page size 16K

A few Things that are New related to XML

1. Implicitly created objects – DBA cannot create explicitly
2. XML indexes: XPath and keys
3. XML Schema registration
4. XMLDATA contains StringIDs in a catalog table SYSIBM.SYSXMLSTRINGS (dictionary)
 - UNLOAD FROMCOPY restricted, DSN1COPY
5. A new XML lock type, ID '35'x in traces
 - IFCID 20, 21, 107, 150, 172, and 196.
6. XML keyword in some utilities

LOB or XML ?

LOB

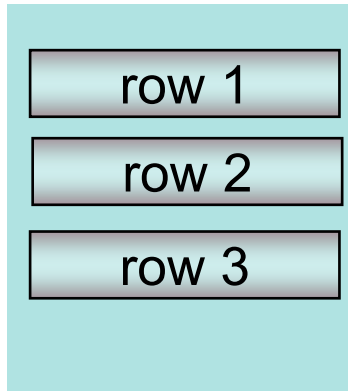
- LOB space
 - ▶ No Compression
- Flow optimization
 - ▶ Plus zIIP DRDA
- Low CPU Cost in Insert/Update
- Low CPU in select whole document
- LOB Lock Avoidance
- No XML Query
- No Index

XML

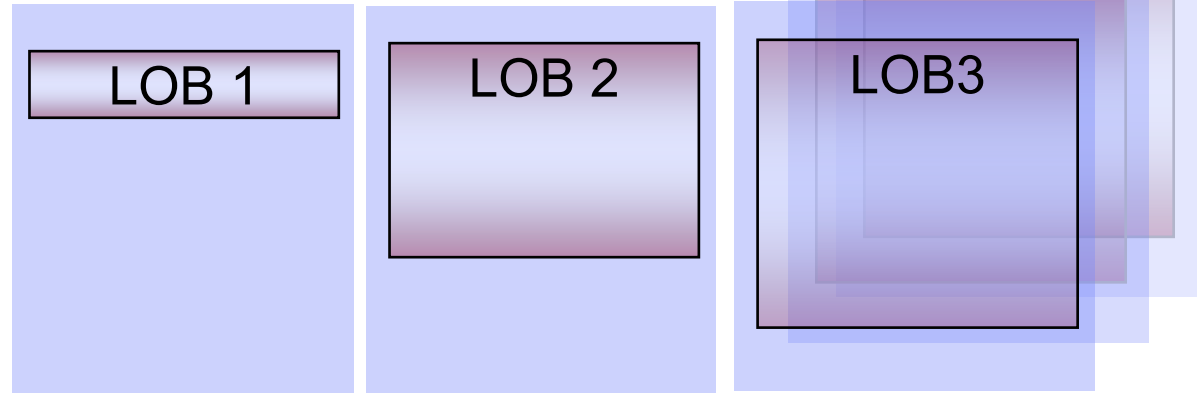
- Universal Table Space
 - ▶ Compression
 - ▶ Append
- Flow optimization + streaming
 - ▶ Plus zIIP DRDA
- High cost in Insert/Update
 - ▶ zIIP/zAAP eligible for parser
- Validation
- XML Index
- XML Query using XPath, SQL/XML
- XML Table

LOB and XML

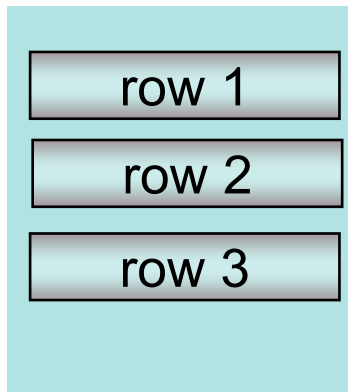
BASE table



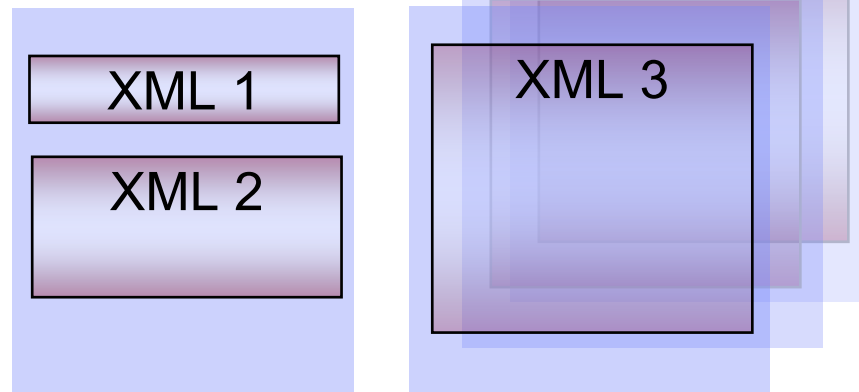
LOB Aux table (4K,8K,16K,32K)



BASE table

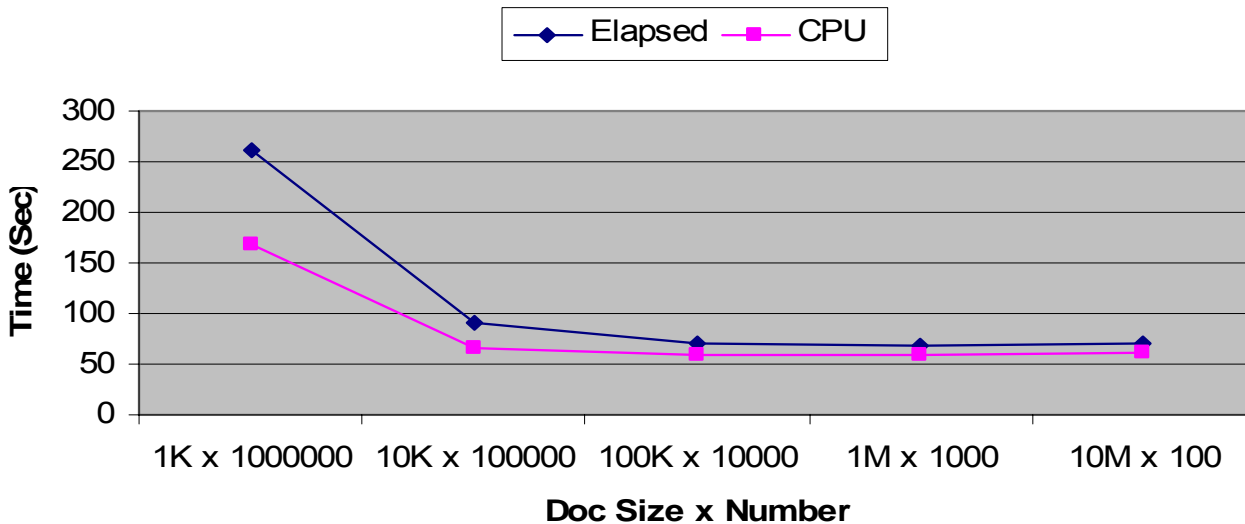


XML table(16K)



Best Practice 1 : Choose “right” XML size

- Pick your XML size with your unit of UPDATE
 - ▶ Liner cost on Insert/Serialization cost
 - ▶ Update is an entire document replacement
 - ▶ Concurrency
 - New XML lock for document level concurrency



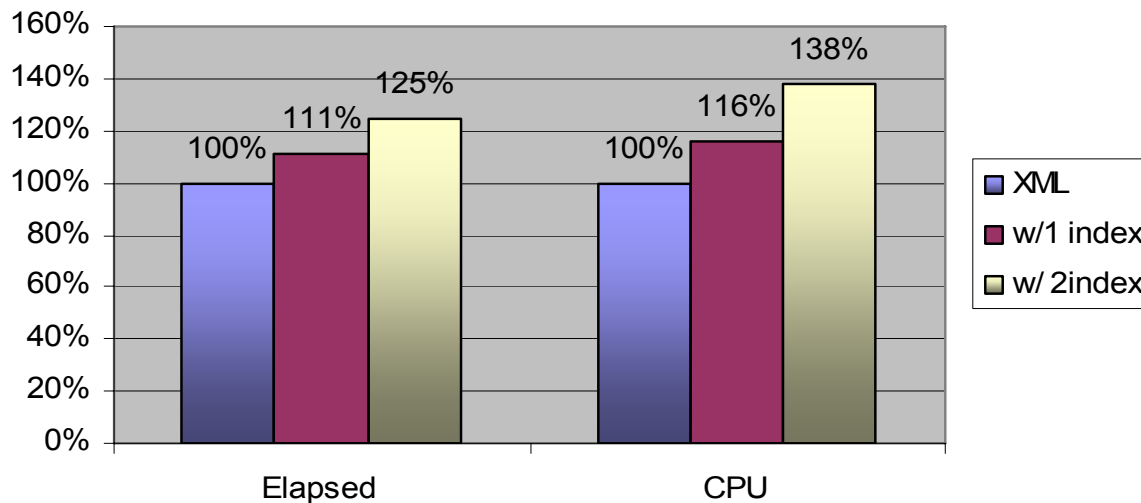
XML Concurrency

- Document level concurrency by XML lock – X'35' lock
 - ▶ Before PK55966 (Dec/07) There were additional s-page locks on XML table
 - ▶ Before PK72604/PK68265(plan 4Q/08) There are additional x-page locks on XML table

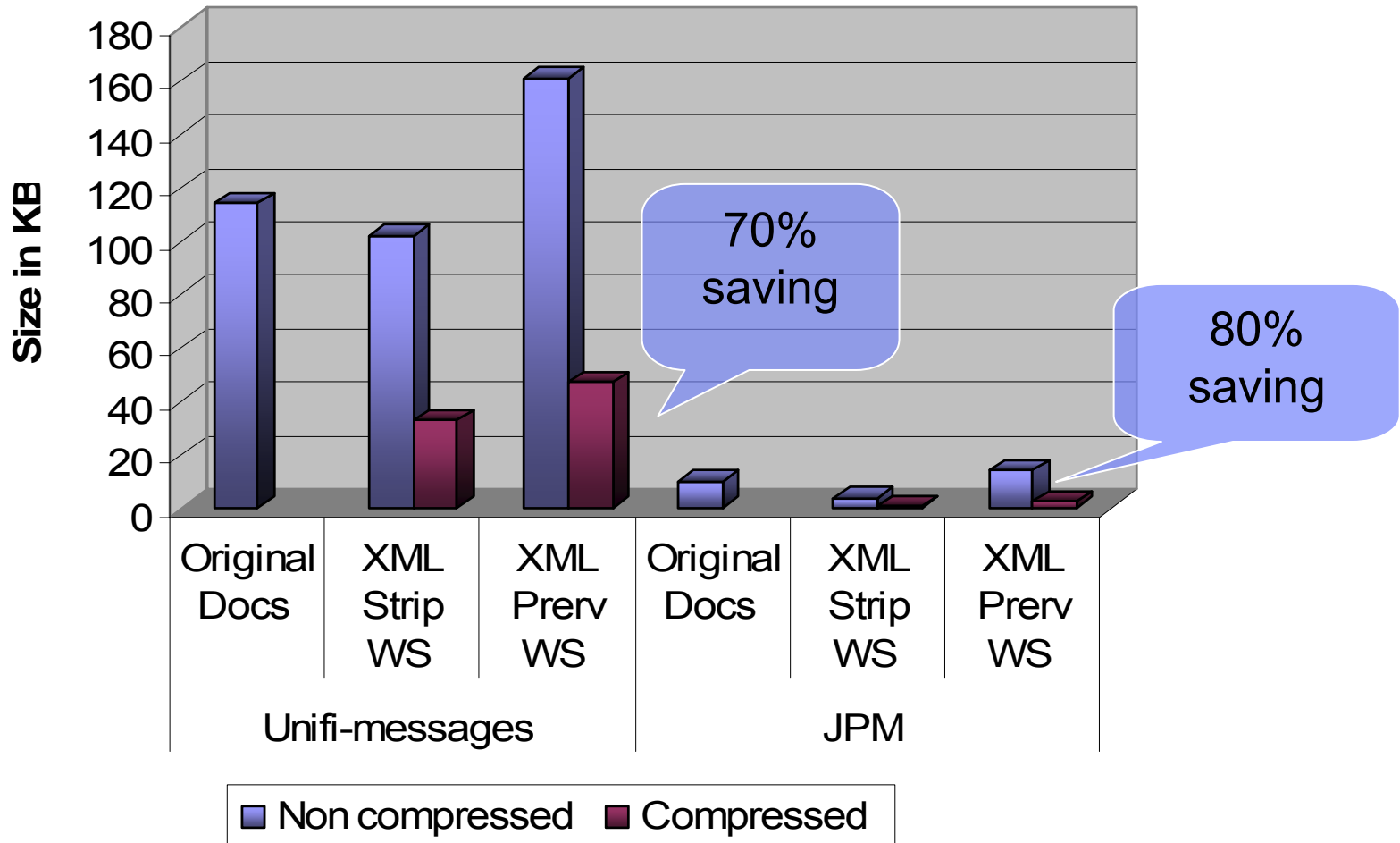
DML	Base Table Lock Size	Base Lock	XML Lock size	XML Lock X'35'	XML table Lock
INSERT	PAGE/ROW	X	Document	X (commit)	Page latch
UPDATE/ DELETE	PAGE/ROW	S->X U->X X	Document	X (commit)	Page latch
SELECT with CS, CD(NO)	PAGE/ROW	S (release at next fetch) or lock avoidance	Document	S (next fetch)	None

Best Practice 2 : Improve Insert Performance

- Insert performance considerations
 - ▶ Logging - hidden log cost
 - ▶ Spread your insert to multiple partitions if possible
 - ▶ Define only the necessary indexes /a/b/c/@d rather than //@d



Best Practice 3: Use Compression for Preserving White Space



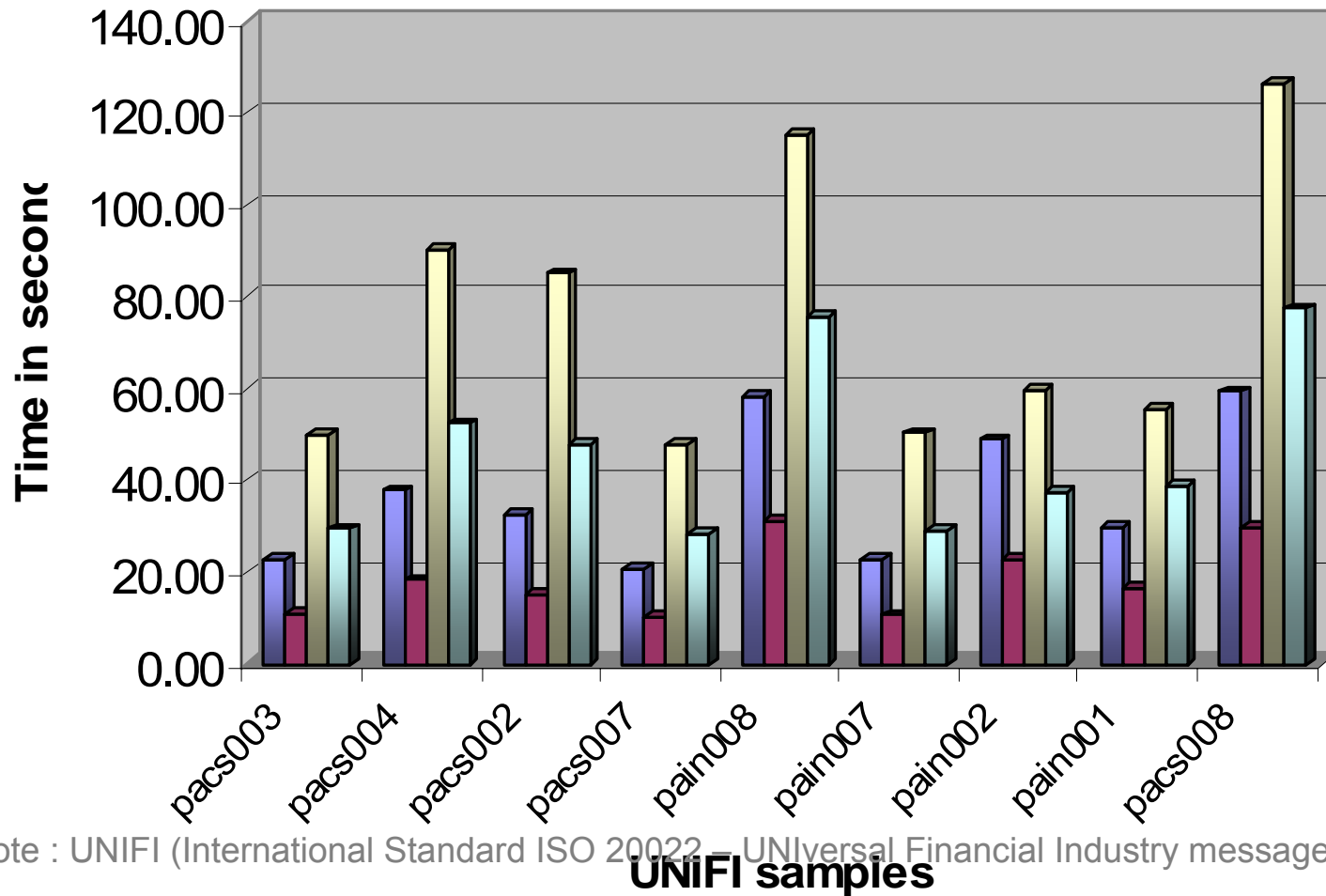
Best Practice 4: Be Aware of Cost of Validation

- Insert using UDF (SYSFUN.DSN_XMLVALIDATE function)
 - ▶ Register XML schema in DB2 XSR
 - ▶ Current limitation doc size 50MB

```
Insert into Prod (pid, pname,xmlcol) values ( 12345, "Minnie Mouse",  
XMLPARSE(DOCUMENT SYSFUN.DSN_XMLVALIDATE (CAST ? AS CLOB),  
'SYSXSR.PRODCT'))
```

- Typically 1.8- 3x CPU increase from non-validation case
 - ▶ Future plan
 - Remove 50MB limitation
 - Support of Validation via Specialty Engines (zIIP/zAAP)

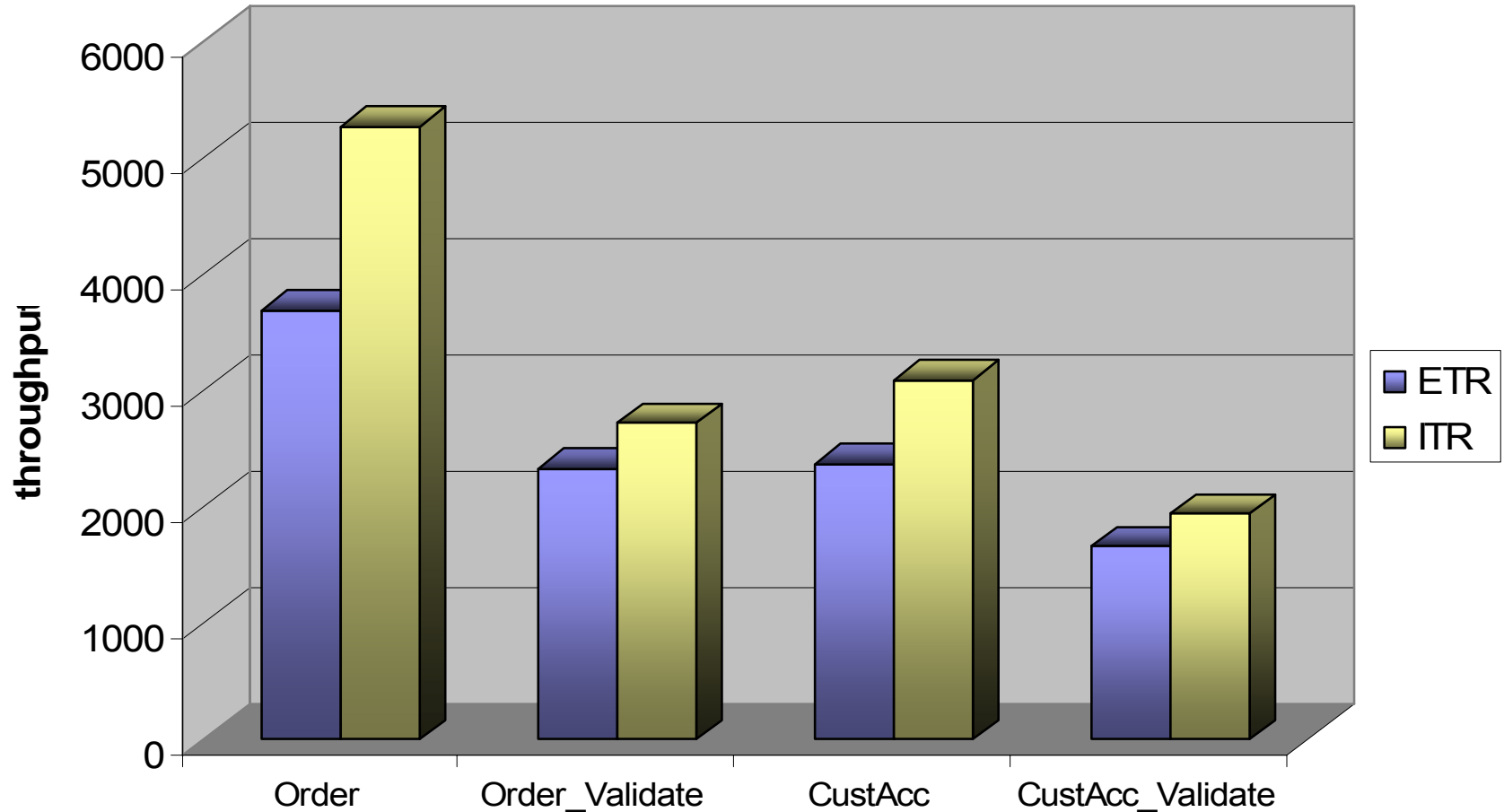
UNIFI Validation 50K insert



Note : UNIFI (International Standard ISO 20022 – UNiversal Financial Industry message scheme)

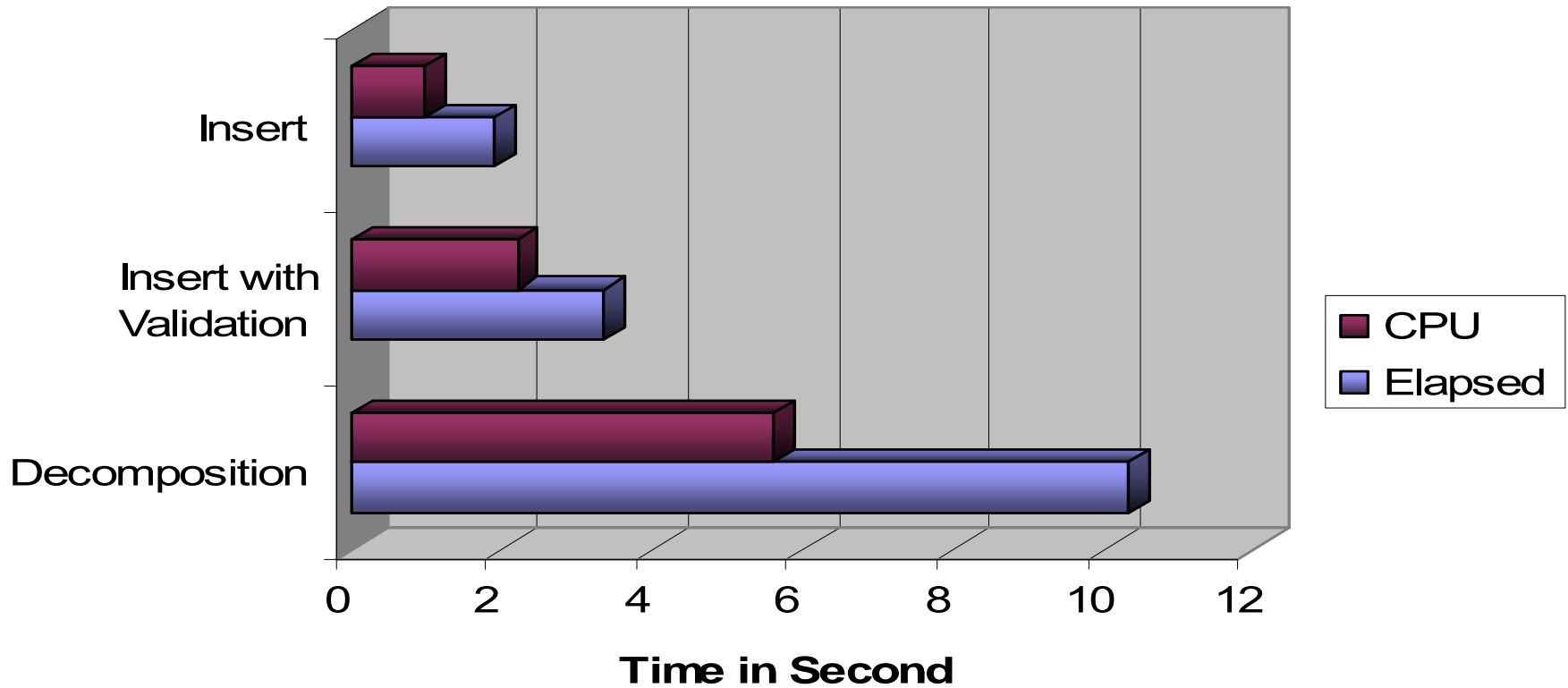
Insert and Validation : Multi threads

TPoX Insert 20 threads



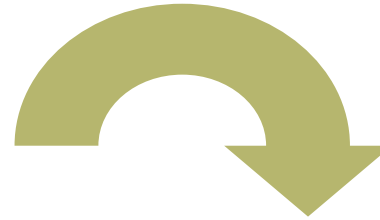
Insert or Shred?

4K doc single thd, 3000x



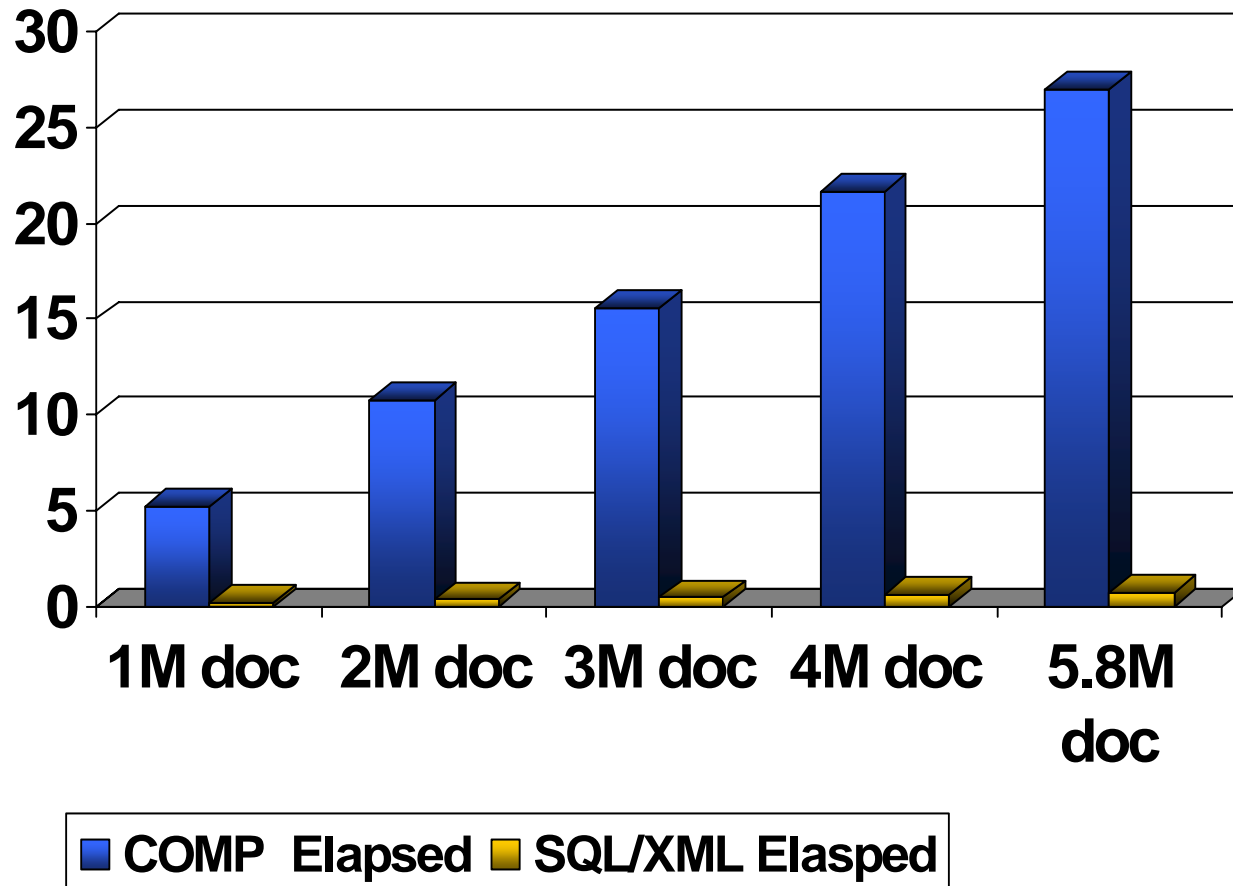
Quick Shredding – XMLTABLE

```
<product>
  <item item_id = 777>
    <name>Magic Pen</name>
    <category> Stationary </category>
    <manufacture >
      <name> "ABC" </name>
    </manufacture>
    <price> 5.00 </price>
    <comment> era
  </item>
</product >
```



```
SELECT T.*
FROM Prod X,
XMLTable ('//item',
          PASSING X.XMLCOL
          COLUMNS
            "Item_ID"          INTEGER      PATH '@item_id',
            "Product Name"    CHAR(20)     PATH 'name',
            "Manufacture"     CHAR(20)     PATH 'manufacture/name',
            "US Price"        DEC(9,2)     PATH 'price',
            "Comment"         CHAR(80)     PATH 'comment'
          WITH ORDINALITY "Seqno") AS T
WHERE X.PID = ? ;
```

Best Practice 5: Use XML Publishing (V8 NFM)



Best Practice 6: Speed up with XML Index

Assuming there are 100,000 rows (100,000 XML doc) in the table

```
<product>
  <item item_id = 777>
    <name>Magic Pen</name>
    <category> Stationary </category>
    <manufacture >
      <name> "ABC" </name>
    </manufacture>
    <price> 5 </price>
    <comment> erasable pen </comment>
  </item>
</product >
```

SELECT .. WHERE **XMLEXISTS**
(**'/product/item[@item_id = 777]'...**);

INDEX on **'/product/item/@item_id'**

Or **'//@item_id'** AS SQL DECFLOAT ;

QNO	A_TYPE	A_NAME	MC	PREFETCH
1	DX	IX1	1	L

Speed up with XML Index

```
<product>
  <item item_id = 777>
    <name>Magic Pen</name>
    <category> Stationary </category>
    <manufacture >
      <name> "ABC" </name>
    </manufacture>
    <price> 5 </price>
    <comment> erasable pen </comment>
  </item>
</product >
```

XMLEXISTS

('/product/item[@item_id = 777]'...)

AND XMLEXISTS

('//manufacture[name="ABC"] '...);

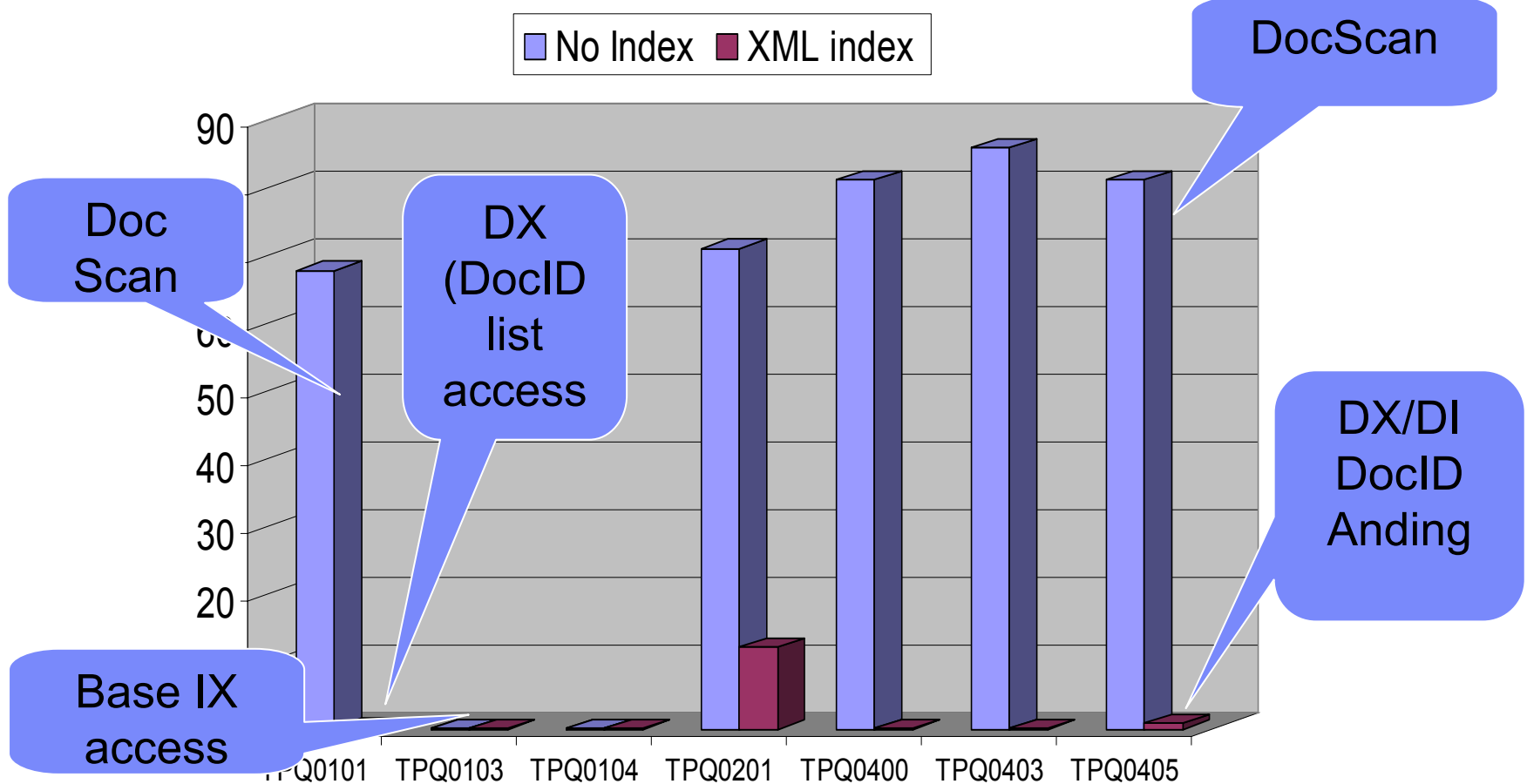
INDEX on **'/product/item/@item_id'** AS DECIMAL(10,2);

INDEX on **'//manufacture/name'** AS SQL VARCHAR(26);

QNO	A_TYPE	A_NAME	MC	INDX_ONLY	PREFETCH
1	M		0	N	L
1	DX	IX1	1	Y	
1	DX	IX2	1	Y	S
1	DI		0	N	

Speed up with XML index

- Having a good XML index is critical for query performance



Best Practice 7 : Code Specific XML path and define “just right” index

```
<product>
  <item item_id = 777>
    <name>Magic Pen</name>
    <category> Stationary </category>
    <manufacture >
      <name>“ABC” </name>
    </manufacture>
    <price> 5 </price>
    <comment> erasable pen </comment>
  </item>
</product >
```

Filtered by

/product/item/manufacture/name

Indexed by

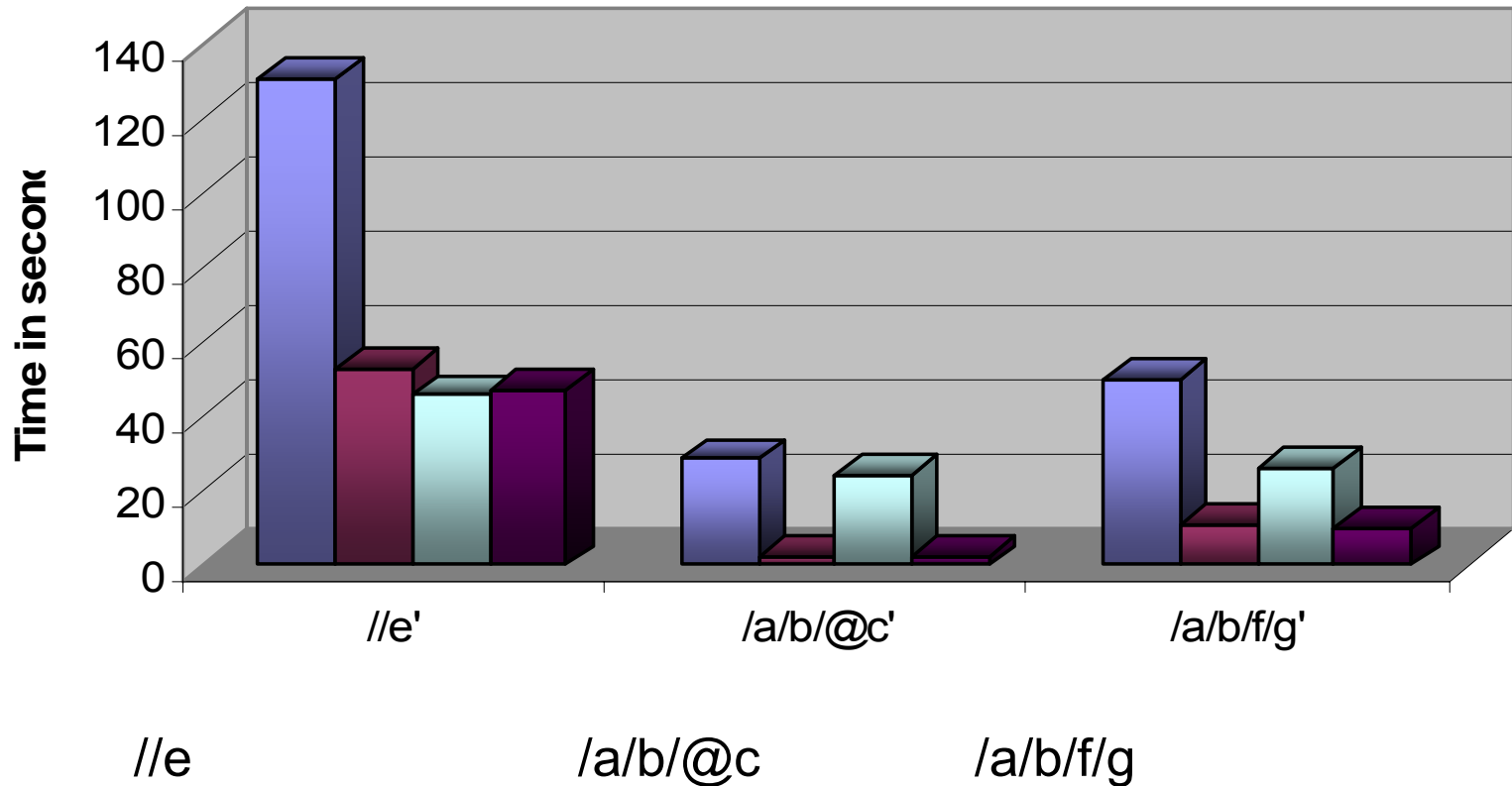
/product/item/manufacture/name

Indexed by

//name

XML Index Create or Rebuild

■ Create Elapsed ■ Create CPU ■ Rebuild Elapsed ■ Rebuild CPU



Best Practice 8 : RUNSTATS on XML Table and Indexes

- RUNSTATS against XML table is “required”

- ▶ LISTDEF

```
LISTDEF LIST01 INCLUDE TABLESPACES DATABASE XMLDB  
RUNSTATS TABLESPACE LIST LIST01 TABLE(ALL) INDEX(ALL)
```

- ▶ RUNSTATS does not scan XML documents.
 - Collect basic information (number of pages, card)

Summary for XML Access and Index Usage

- Indexes are only for XMLEXISTS predicate
- Use fully specified paths than // or * in your queries
- Index type and Namespaces in Index and Query needs to match
- Index and query containment
 - ▶ //d index can be used for XMLEXISTS /a/b/c/d or //d
 - ▶ /a/b/c/d index can be used for /a/b/c/d but not for //d
- Insert overhead
 - ▶ //d index may be more expensive to maintain than /a/b/c/d
- Do not forget RUNSTATS

Best Practice 9 : UNLOAD to HFS

- LOAD/UNLOAD using File Reference Variables
 - ▶ XML > 32KBytes

```
TEMPLATE TCLOBF DSN /u/akiko/xml3/M25/EBCDIC DIR(10) DSNTYPE(HFS)
UNLOAD TABLESPACE XMLDB.TS25M
FROM TABLE USRT014.TB25M
  (XML POSITION(*) VARCHAR CLOBF TCLOBF )
  EBCDIC
```

- Avoid zFS
 - ▶ zFS has known performance issue when single directory contains millions of files.

Best Practice 10 : Monitor XML Performance

- Traditional Performance monitoring tools can be used for XML performance monitoring.
 - ▶ DB2 Accounting, Statistics trace and RMF
 - ▶ Access path check – OSC or plan tables
- Monitor XML storage consumption periodically.
 - ▶ Accounting/Statistics will have XML storage usage section
 - ▶ Zparm XMLVALA, XMLVALS to control XML storage

Best Practice 11 : Stay Current on XML Maintenance

- XML APARs
 - ▶ XMLTABLE function by [PK51571](#), [PK51572](#), [PK51573](#)
 - XMLTABLE and XMLCAST support
 - XML Query various performance and storage optimization
 - ▶ XML LOAD Performance improvement by [PK47594](#)
 - ▶ Additional BIF by [PK55585/PK55831](#)
 - ▶ XML query performance improvement by [PK58914](#), [PK56337](#)
 - ▶ XML access path by [PK57158](#)
 - ▶ XML storage zparm by [PK25747](#), XML thread storage monitor by [PK54837](#)
 - ▶ XML concurrency improvement by [PK52885](#), [PK55966](#), [PK59770](#), [PK68265](#)
 - ▶ XML insert with index performance improvement by [PK66218](#)
- Universal table space – PBG APAR
 - ▶ PGB concurrency improvement by [PK72414](#) [PK72415](#) [PK72417](#) [PK72419](#) [PK72420](#)
- XML System Service - up to 30% CPU reduction in parsing by z/OS 1.9 or above

Best Practice to Reduce Cost of XML : DRDA and XML System Service Redirection

- DB2 zIIP support will apply with XML process
 - ▶ DRDA zIIP
 - ▶ Utility index zIIP (Only base table indexes)
- Non Validation Parser zAAP support
 - ▶ 100% of XML System Service under TCB mode is eligible
 - ▶ z/OS 1.9 above and 1.8/1.7 with APARs
 - ▶ PK50575 for zAAP accounting support
- Non Validatoin Parser zIIP support
 - ▶ 100% of XML System Service under SRB mode is eligible
 - ▶ z/OS 1.10 and 1.9/1.8 with APARs
- Validation Parser zIIP support - planned in V9 maintenance stream

What is Eligible on Which IBM Specialty Engines?

DB2 9 XML workload invoked from . . .	DRDA® via TCPIP	Execution Mode	zAAP eligible	zIIP eligible
CICS®	No	TCB	Yes	No
IMS™	No	TCB	Yes	No
TSO	No	TCB	Yes	No
WebSphere® for z/OS (JCC T2)	No	TCB	Yes	No
Stored Procedure/UDF/Trigger	No	TCB	Yes	No
Native SQL Stored Procedure	No	TCB	Yes	No
Call Attach (CAF)	No	TCB	Yes	No
RRS Attach	No	TCB	Yes	No
Local LOAD utility (data)	No	TCB	Yes	No
Index Build phase of LOAD *	No	SRB	No	Yes
WebSphere for z/OS (JCC T4)	Yes	SRB	No	Yes
WebSphere distributed (JCC T4)	Yes	SRB	No	Yes
Distributed transactions using DB2 Connect™	Yes	SRB	No	Yes
Stored Procedure/UDF/Trigger	Yes	TCB	Yes	No
Native SQL Stored Procedure	Yes	SRB	No	Yes

zIIP/zAAP Monitoring

- RMF WLM Activity Report
- DB2 Accounting : Combined zIIP/zAAP counter (PK50575 5/08)

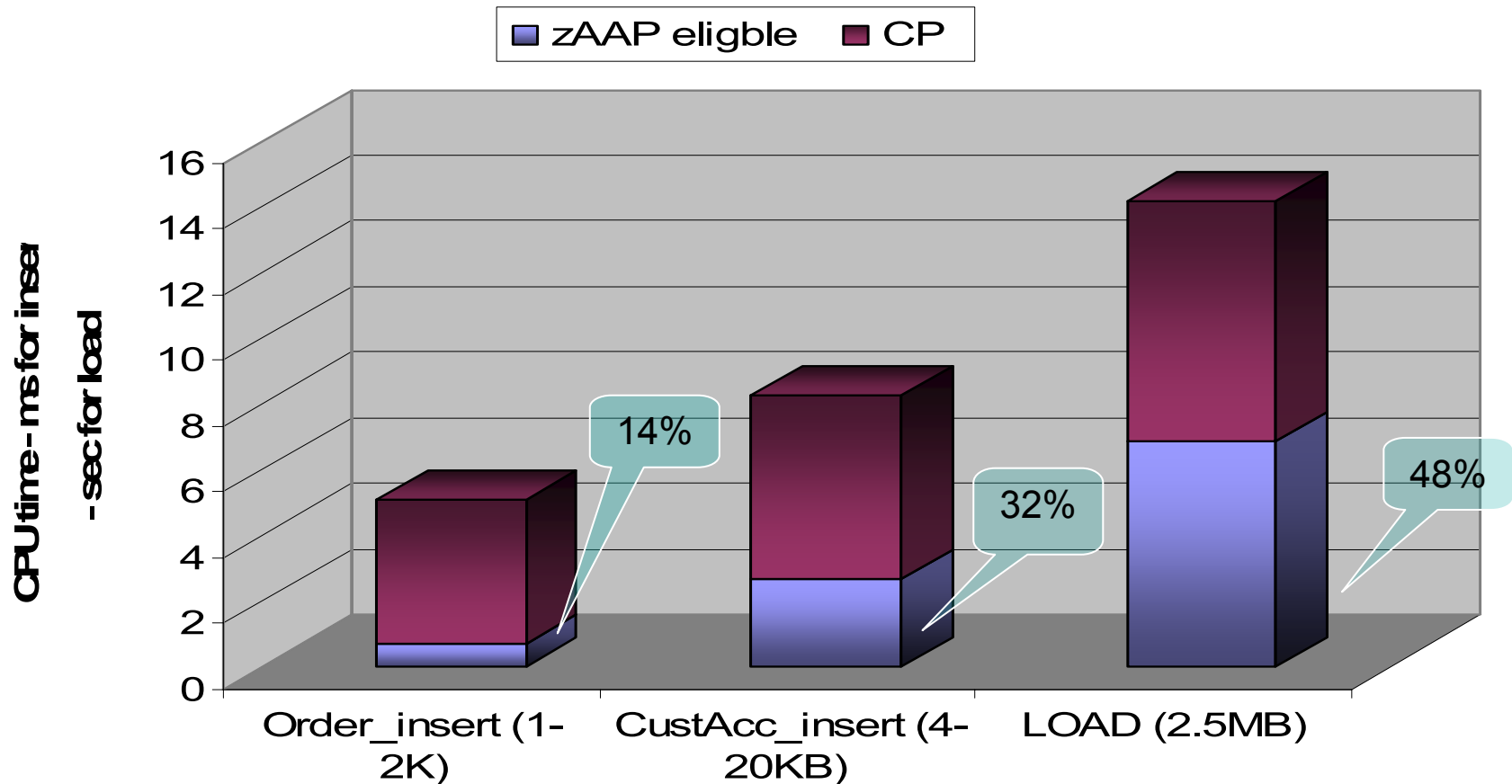
AVERAGE	APPL (CL.1)	DB2 (CL.2)
-----	-----	-----
CP CPU TIME	0.006263	0.005252
AGENT	0.006263	0.005252
NONNESTED	0.001553	0.000560
STORED PROC	0.004660	0.004660
UDF	0.000049	0.000032
TRIGGER	0.000000	0.000000
PAR.TASKS	0.000000	0.000000
SECP CPU	0.001739	N/A
SE CPU TIME	0.003546	0.002518
NONNESTED	0.001580	0.000351
STORED PROC	0.001967	0.001967

Eligible for zIIP but did not run on zIIP (same as IICP)

Time spent in Specialty Engines (zIIP + zAAP)

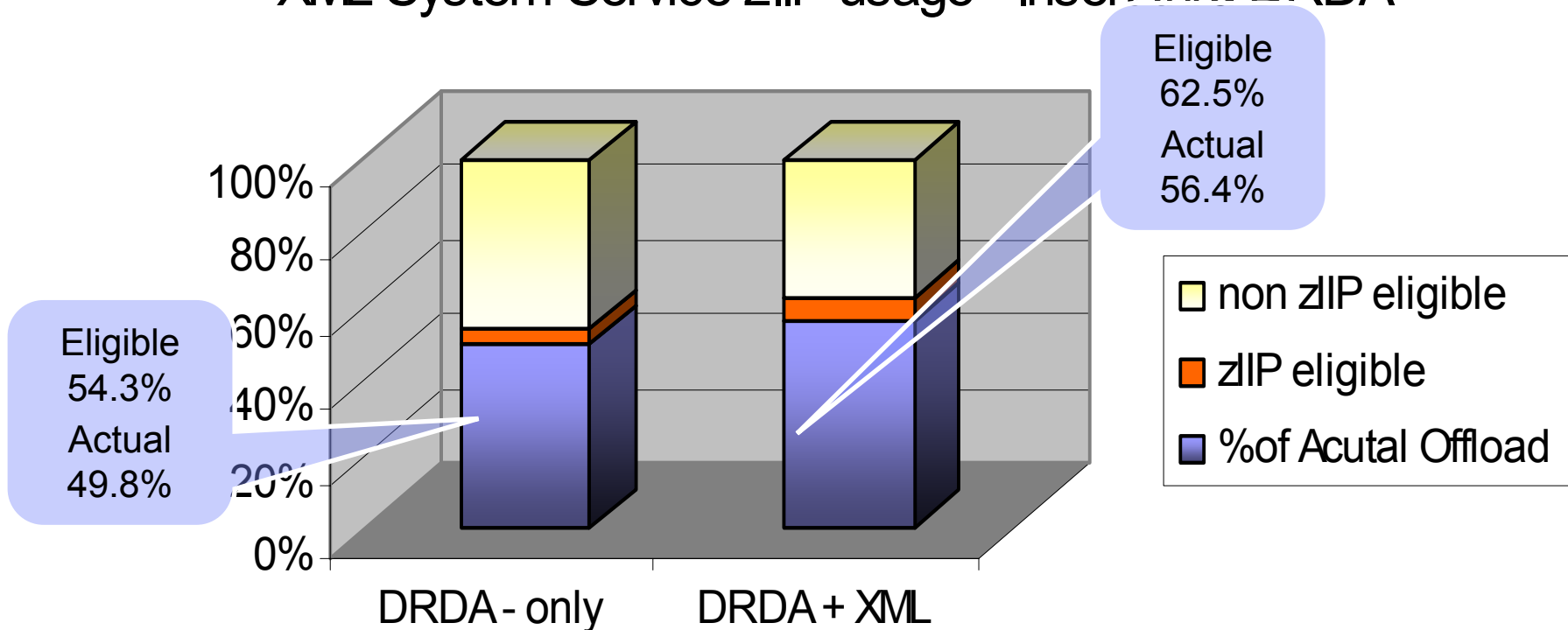
XML System Service Specialty Engine Support

XML System Service in Insert and Load

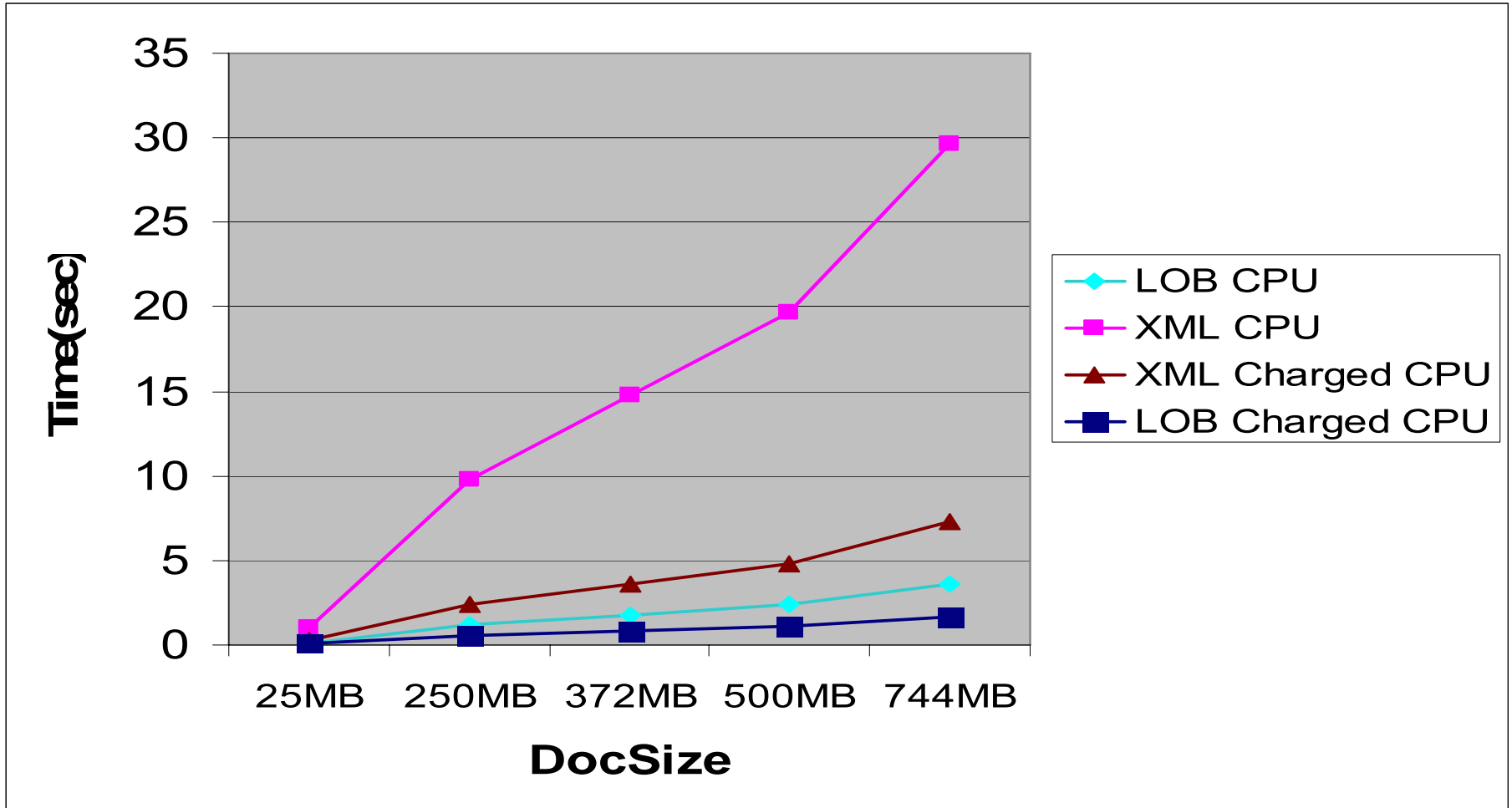


XML System Service Specialty Engine Support

XML System Service zIIP usage- Insert thru DRDA



LOB vs. XML with zIIP



zIIP Tuning Practice

- Avoid zIIP Constraint situation
 - ▶ Add more zIIP processors
 - ▶ ZIIPAWMT (ZAAPAWMT for zAAP) in IEAOPTxx
 - Default 12000 (12 ms)
 - Impact when the specialty engine will ask for help from the general CPs.
 - If response time is growing and you want the general CPs to help with more work, lower to 1000 or 500 (z10)
 - If response time is fine, use default 12000

Best Practices....

1. Choose right XML size
2. Insert Performance
3. Consider Compression
4. Be Aware of Cost of Validation
5. Use XML Publishing Function for composition
6. Speed Up with XML Indexes
7. Code Specific XML path and “right” level path for index
8. RUNSTATS on XML Tables and Indexes
9. Unload to HFS
10. Monitor XML Performance
11. Stay Current on XML maintenance
12. Utilize IBM Specialty Engines for XML applications

Resources

- **DB2 9 and z/OS XML System Services Synergy Update**
 - ▶ <http://www-03.ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP101227>
- **DB2 9 for z/OS Performance Redbook**
 - ▶ <http://www.redbooks.ibm.com/redbooks/pdfs/sg247473.pdf>.
- **DB2 Version 9.1 for z/OS XML Guide, SC18-9858**
- **XML topics in DB2developerworks**
 - ▶ <http://www-128.ibm.com/developerworks/wikis/display/db2xml/Home>

Disclaimer

© Copyright IBM Corporation [current year]. All rights reserved.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

IBM, the IBM logo, ibm.com, DB2, PureXML are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

Thank you!

akiko@us.ibm.com

XML Questions ?

Send them to db2zxml@us.ibm.com