
Universe on XML data sources

Marc Daniau – Product group

marc.daniau@sap.com



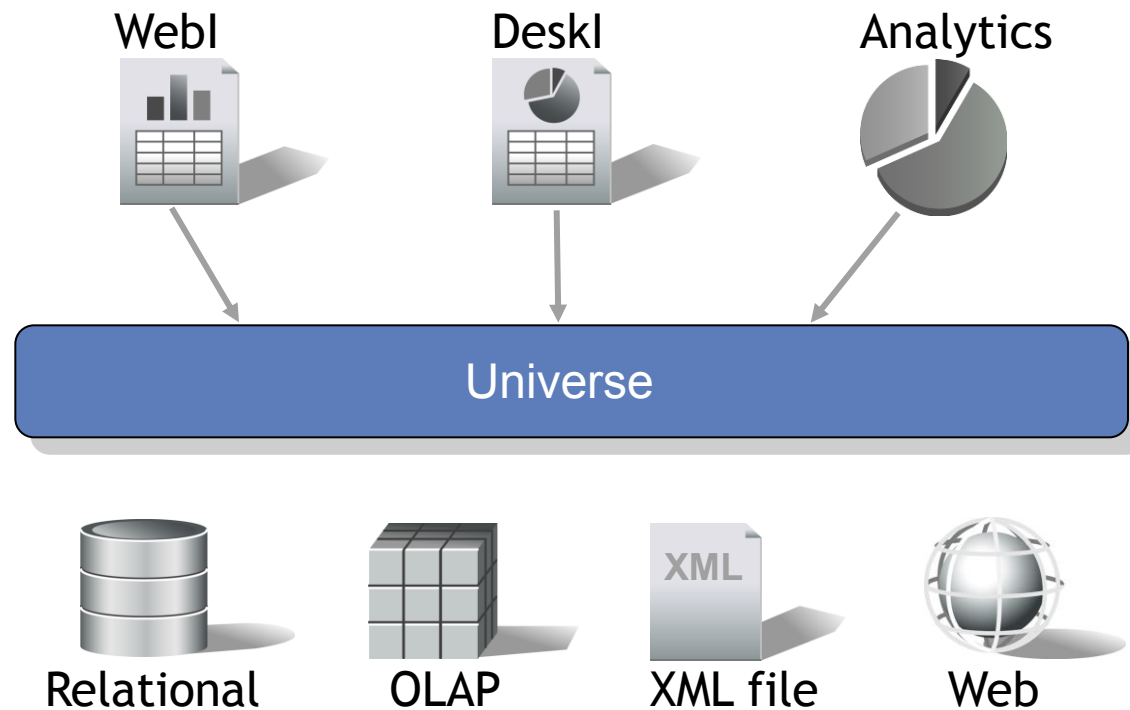
Different kinds of XML data

XML data can be found in a

- ▶ XML column
- ▶ XML file
- ▶ RSS Feed
- ▶ Web service
- ▶ PMML data mining model
- ▶ XBRL document

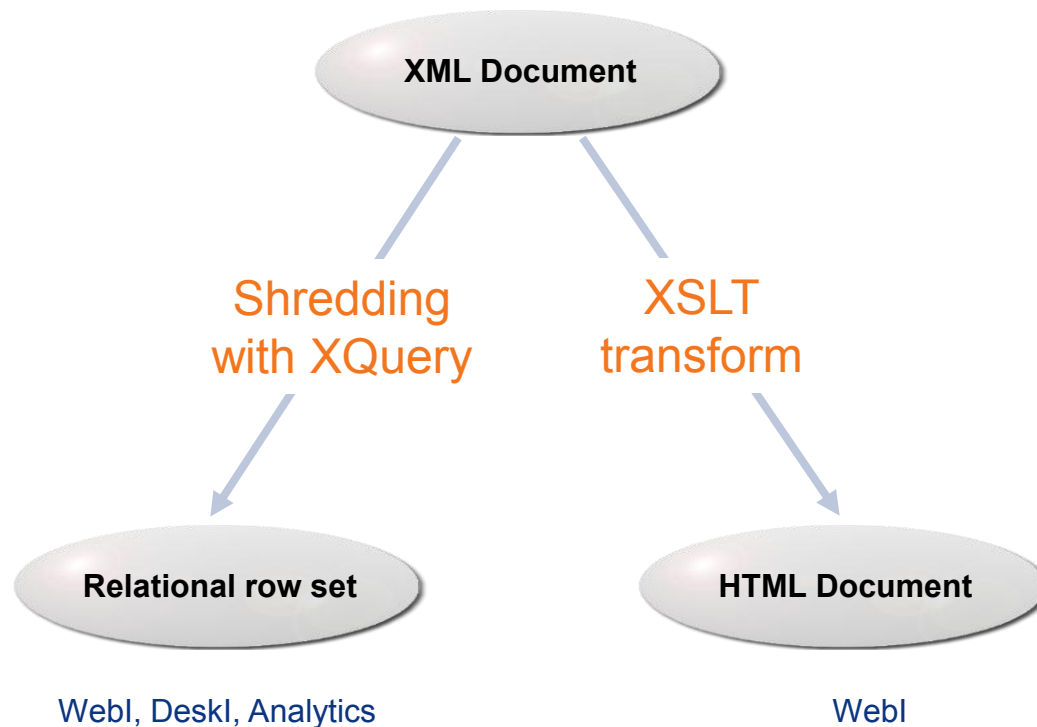
The Business Objects Universe

Querying various data sources and serving multiple clients



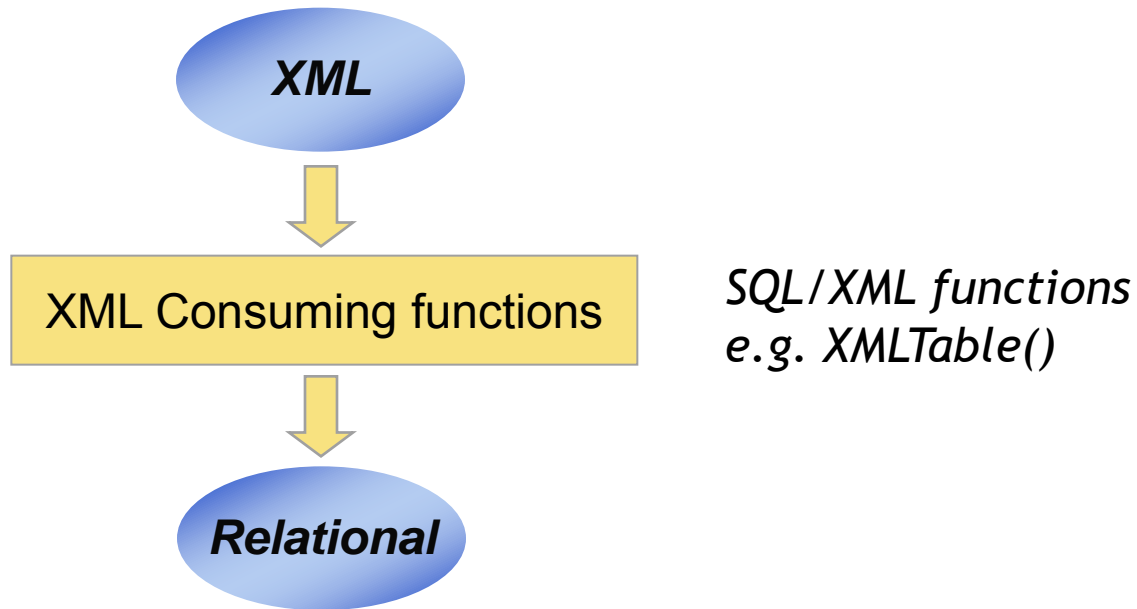
Two ways for consuming XML data

We will present two ways to query XML through a Universe



- ▶ **From XML to relational**
- ▶ **From XML to HTML**
- ▶ **From relational to HTML**
- ▶ **Mixing relational with XML**

From XML to relational



The AdventureWorks demo database

The XML column Store.Demographics

The screenshot displays a SQL Server Enterprise Manager window with a query window and a results window. The query window shows the following SQL statement:

```
SELECT Name, Demographics
FROM Sales.Store
```

The results window shows a table with two columns: Name and Demographics. The data is as follows:

Name	Demographics
1 A Bike Store	<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey"><AnnualSales>300000</AnnualSales></StoreSurvey>
2 Progressive Sports	<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey"><AnnualSales>800000</AnnualSales></StoreSurvey>
3 Advanced Bike Components	<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey"><AnnualSales>1500000</AnnualSales></StoreSurvey>
4 Modular Cycle Systems	<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey"><AnnualSales>300000</AnnualSales></StoreSurvey>
5 Metropolitan Sports Supply	<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey"><AnnualSales>800000</AnnualSales></StoreSurvey>
6 Arctic Exercise Company	<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey"><AnnualSales>1500000</AnnualSales></StoreSurvey>

The Demographics column contains XML data for each store. A callout box points to the XML content for the first store, "A Bike Store". The XML content is as follows:

```
<StoreSurvey xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey">
  <AnnualSales>300000</AnnualSales>
  <AnnualRevenue>30000</AnnualRevenue>
  <BankName>International Bank</BankName>
  <BusinessType>BM</BusinessType>
  <YearOpened>1970</YearOpened>
  <Specialty>Road</Specialty>
  <SquareFeet>7000</SquareFeet>
  <Brands>AW</Brands>
  <Internet>T1</Internet>
  <NumberEmployees>2</NumberEmployees>
</StoreSurvey>
```

An orange arrow points from the first row of the results table to the XML content in the Demographics column. A yellow callout box with a red border points to the XML content and contains the text: "XML content for a single store".

Shredding XML with XQuery against the Microsoft demo database

We define one business object for each XML element that we need to extract

The screenshot shows the SQL Server Enterprise Manager interface. On the left, a tree view displays the 'Sales' database structure, with 'Store' expanded to show 'Bank'. In the center, a schema diagram shows three tables: 'Sales.SalesTerritory' (10 rows), 'Sales.SalesPerson' (17 rows), and 'Sales.Store' (701 rows). Lines indicate relationships: 'SalesTerritory' has a one-to-many relationship with 'SalesPerson', and 'SalesPerson' has a one-to-many relationship with 'Sales.Store'. On the right, a yellow speech bubble contains the text: 'A Universe on top of the AdventureWorks demo database'. Below the diagram, another yellow speech bubble contains the text: 'The SQL definition of the « Bank » object'. At the top, an XQuery statement is shown in a text box: `Store.Demographics.value('declare default element namespace "http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSur`

```
store.Demographics.value('declare default element namespace  
"http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/StoreSurvey";  
/StoreSurvey[1]/BankName', 'varchar(80)')
```


Shredding XML with XQuery against the Microsoft demo database

The demographics XML content decomposed

Australia

Store	Specialty	Annual Sales	Square Feet	Opening Year	Number of Employees	Bank
Bike Part Wholesalers	Touring	800,000	21,000	1999	15	International Security
Budget Toy Store	Mountain	3,000,000	71,000	1998	52	International Bank
Cross-town Parts Shop	Road	800,000	23,000	1990	19	Guardian Bank
Cycle Parts and Accessories	Touring	800,000	20,000	1985	10	International Bank
Eastside Cycle Shop	Touring	800,000	21,000	1988	11	Guardian Bank
Eastward Bike Accessories	Touring	1,500,000	39,000	1974	39	Reserve Security
Expert Cycle Store	Touring	1,500,000	38,000	1995	37	Primary International
Fast Bike Works	Touring	1,000,000	25,000	1971	28	Primary International
First Supplies	Touring	1,000,000	26,000	1996	22	United Security
Fitness Bike Accessories	Touring	1,500,000	39,000	1986	47	Reserve Security
Fitness Discount Store	Touring	800,000	19,000	1993	11	Primary International

Querying an XML file through an IBM DB2 Universe



The definition of the derived table « MENU »

```
SELECT T.NAME, T.DESC, T.PRICE, T.CALORIES
FROM
(
  SELECT
    xmlparse(document
      db2xml.extractCLOB(
        db2xml.xmlfile('D:\public_data\menu.xml'), '//')
      preserve whitespace) as XDOC
    FROM sysibm.sysdummy1
  ) as X,
  xmltable(xmlnamespaces(default 'http://bobjsample.org'),
    '$d/breakfast/food' passing X.XDOC as "d" columns
    NAME          VARCHAR(80) path 'name',
    DESC          VARCHAR(80) path 'description',
    PRICE         VARCHAR(35) path 'price',
    CALORIES      INTEGER      path 'calories'
  ) AS T
```

Extracting the content of the XML file

Shredding the XML document with XQuery

MENU	
NAME	C
DESC	C
PRICE	C
CALORIES	N

Querying an XML file through an IBM DB2 Universe



The XML file and the BOBJ document built on top

```
<?xml version="1.0" encoding="UTF-8"?>
<breakfast xmlns='http://bobjsample.org'>
  <food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>Belgian Waffles with maple syrup</description>
    <calories>650</calories>
  </food>
  <food>
    <name>French Toast</name>
    <price>$4.50</price>
    <description>Thick slices of sourdough bread</description>
    <calories>600</calories>
  </food>
  <food>
    <name>Homestyle Breakfast</name>
    <price>$6.95</price>
    <description>Eggs, bacon, toast and hash browns</description>
    <calories>950</calories>
  </food>
</breakfast>
```

Breakfast Item	Price	Calories
Belgian Waffles	\$5.95	650
French Toast	\$4.50	600
Homestyle Breakfast	\$6.95	950

Consuming Web services

Getting the weather forecast



An IBM DB2 Universe against a SOAP based web service

```
select T.DAY_SEQ, T.DAY_NAME, T.MIN_TEMP_C, T.MAX_TEMP_C
from
(
  SELECT xmlparse(document DB2XML.SOAPHHTTPV(
    'http://www.websvicex.net/WeatherForecast.asmx',
    'http://www.websvicex.net/GetweatherByPlaceName',
    '<GetweatherByPlaceName xmlns="http://www.websvicex.net"><PlaceName>'
    || @Prompt('Enter Place','C',,mono,free) || '</PlaceName></GetweatherByPlaceName>'
  ) preserve whitespace) as XDOC
  FROM sysibm.sysdummy1
) as X,
XMLTABLE(XMLNAMESPACES(DEFAULT 'http://www.websvicex.net'),
  'for $w at $i in
  $d/GetweatherByPlaceNameResponse/GetweatherByPlaceNameResult/Details/WeatherData
return <forecast>
  <pos>{$i}</pos>
  <fullday>{$w/Day}</fullday>
  <tempmax>{$w/MaxTemperatureC}</tempmax>
  <tempmin>{$w/MinTemperatureC}</tempmin>
</forecast>'
  passing X.XDOC as "d" columns
  DAY_NAME      VARCHAR(80)  path 'fullday/Day',
  DAY_SEQ       INTEGER      path 'pos',
  MAX_TEMP_C    FLOAT        path 'tempmax/MaxTemperatureC',
  MIN_TEMP_C    FLOAT        path 'tempmin/MinTemperatureC'
) AS T
```

The SOAP requestor function to get web data

Shredding the XML document with XQuery

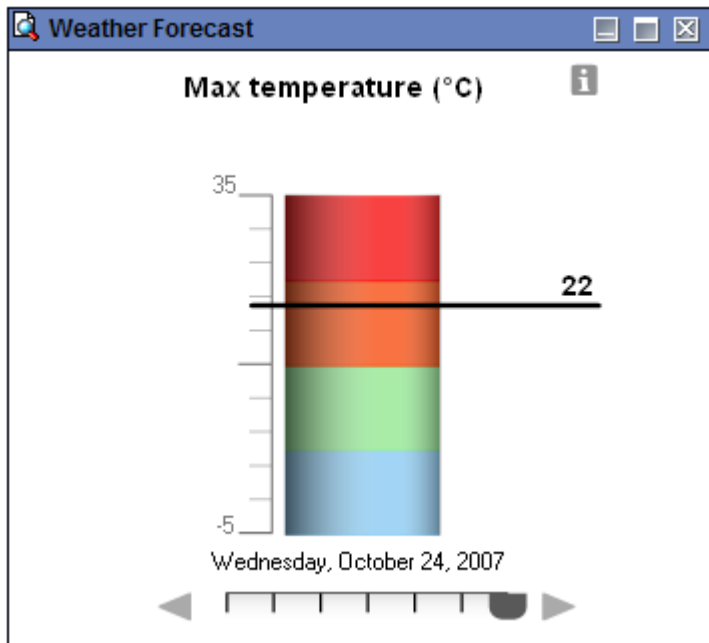
WEATHER	
DAY_SEQ	N
DAY_NAME	C
MIN_TEMP_C	N
MAX_TEMP_C	N

Consuming Web services

Getting the weather forecast



A dashboard analytic on top of the IBM DB2 Universe



Please fill the following prompt(s):

Enter Place

OK Cancel

The figure is a light blue dialog box with a blue border. It contains the text "Please fill the following prompt(s):" at the top. Below this is a label "Enter Place" followed by a text input field containing the text "San Francisco". At the bottom of the dialog box are two buttons: "OK" and "Cancel".

An Oracle PMML data mining model

Output from `dbms_data_mining.get_model_details_xml()`

```
- <PMML version="2.1">
  <Header copyright="Copyright (c) 2004, Oracle Corporation. All rights reserved." />
  - <DataDictionary numberOfFields="9">
    <DataField name="AFFINITY_CARD" optype="categorical" />
    <DataField name="AGE" optype="continuous" />
    <DataField name="BOOKKEEPING_APPLICATION" optype="continuous" />
    <DataField name="CUST_MARITAL_STATUS" optype="categorical" />
    <DataField name="EDUCATION" optype="categorical" />
    <DataField name="HOUSEHOLD_SIZE" optype="categorical" />
    <DataField name="OCCUPATION" optype="categorical" />
    <DataField name="YRS_RESIDENCE" optype="continuous" />
    <DataField name="Y_BOX_GAMES" optype="continuous" />
  </DataDictionary>
  - <TreeModel modelName="DT_SH_CLAS_SAMPLE" functionName="classification"
  + <Extension name="buildSettings">
  + <MiningSchema>
  - <Node id="0" score="0" recordCount="1500">
    <True />
    <ScoreDistribution value="0" recordCount="1120" />
    <ScoreDistribution value="1" recordCount="380" />
  + <Node id="1" score="0" recordCount="712">
  + <Node id="2" score="0" recordCount="788">
    </Node>
  </TreeModel>
</PMML>
```

The attributes of the decision tree model

The hierarchy of nodes

Turning the PMML tree into relational

The definition of the derived table « DT_DYNL »

XQuery flavor

```

SELECT
Z.NODE, Z.PARENT, Z.LEVELN, Z.NSCORE, Z.NPREDICATE, Z.NSURROGATE, Z.RECORDS,
SYS_CONNECT_BY_PATH(NODE, '/') as PATH
FROM
(SELECT
  lpad(to_char(X.node_id),2,0) as NODE, lpad(to_char(X.parent_node_id),2,0) as PARENT,
  X.score as NSCORE, X.record_count as RECORDS, X.node_level as LEVELN,
  to_char(dbms_xmlgen.convert(X.predicate.getclobval(), 1)) as NPREDICATE,
  to_char(dbms_xmlgen.convert(X.surrogate.getclobval(), 1)) AS NSURROGATE
FROM
XMLTABLE('
for $n in /PMML/TreeModel//Node
let $pn := /PMML/TreeModel//Node[Node/@id=$n/@id]
return
<Tag id="{ $n/@id}" parent_id="{ $pn/@id}" score="{ $n/@score}"
record_count="{ $n/@recordCount}" depth="{count($n/ancestor::*)-2}">
  <Predicate>
  {
    if ($n/simplePredicate instance of element(simplePredicate)) then
      concat(
        string($n/simplePredicate/@field),string(' '),
        if ($n/simplePredicate/@operator = "greaterThan") then
... removed part ...
  }
</Tag>
' PASSING dbms_data_mining.get_model_details_xml(@Prompt('Mining
Model','c','Mining Models\Model Name',mono,free))
COLUMNS
"NODE_ID"          NUMBER PATH '/Tag/@id',
"PARENT_NODE_ID"  NUMBER PATH '/Tag/@parent_id',
"SCORE"           NUMBER PATH '/Tag/@score',
"RECORD_COUNT"   NUMBER PATH '/Tag/@record_count',
"NODE_LEVEL"     NUMBER PATH '/Tag/@depth',
"PREDICATE"      XMLType PATH '/Tag/Predicate/text()',
"SURROGATE"      XMLType PATH '/Tag/surrogate/text()'
) X
) Z
WHERE LEVEL > 1 AND Z.LEVELN = LEVEL-1
CONNECT BY PRIOR NODE = PARENT

```

The XMLTable()
function

Calling the function
get_model_details_xml()

The Oracle hierarchical
operator « connect by »

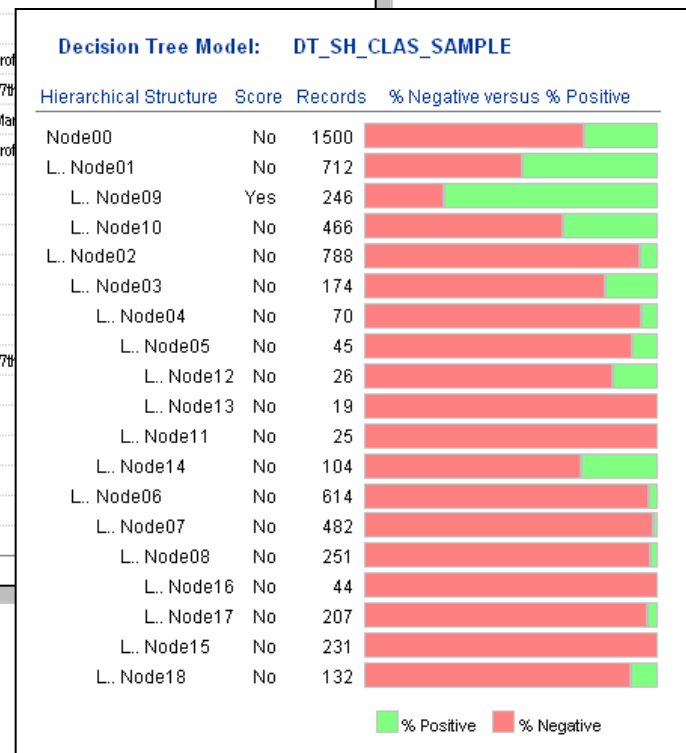
DT_DYNL	
NODE	C
PARENT	C
LEVELN	N
NSCORE	N
NPREDICATE	C
NSURROGATE	C
RECORDS	N
PATH	C

The PMML model rendered in Desk

The decision tree visualized in BOBJ documents

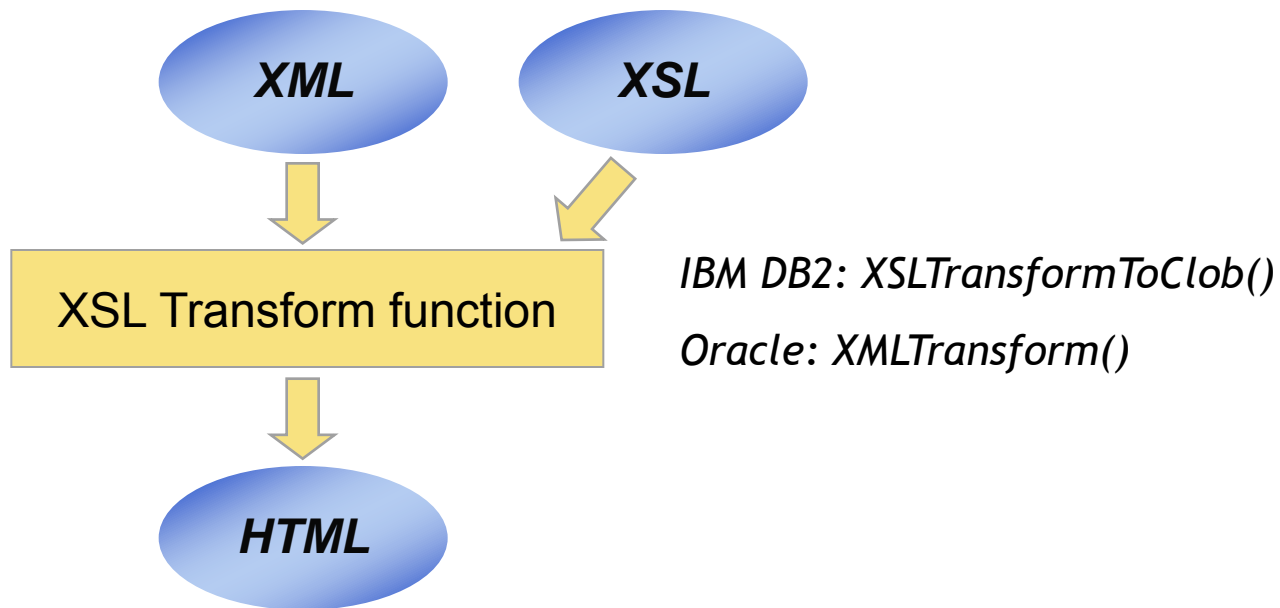
Decision Tree Model: DT_SH_CLAS_SAMPLE

Hierarchical Structure	Score	Negative answers	Positive answers	Percent Negative	Percent Positive	Predicate
Node00	No	1120	380	74.7 %	25.3 %	
L.. Node01	No	382	330	53.7 %	46.3 %	CUST_MARITAL_STATUS isn't "Married"
L.. Node09	Yes	67	179	27.2 %	72.8 %	EDUCATION isn't "Assoc-A" "Bach." "Masters" "PhD" "Prof"
L.. Node10	No	315	151	67.6 %	32.4 %	EDUCATION isn't "10th" "11th" "12th" "1st-4th" "5th-6th" "7th"
L.. Node02	No	738	50	93.7 %	6.3 %	CUST_MARITAL_STATUS isn't "Divorc." "Mabsent" "Mar"
L.. Node03	No	143	31	82.2 %	17.8 %	EDUCATION isn't "Assoc-A" "Bach." "Masters" "PhD" "Prof"
L.. Node04	No	66	4	94.3 %	5.7 %	YRS_RESIDENCE <= 3.5
L.. Node05	No	41	4	91.1 %	8.9 %	AGE > 25.5
L.. Node12	No	22	4	84.6 %	15.4 %	AGE <= 28.5
L.. Node13	No	19	0	100.0 %	0.0 %	AGE > 28.5
L.. Node11	No	25	0	100.0 %	0.0 %	AGE <= 25.5
L.. Node14	No	77	27	74.0 %	26.0 %	YRS_RESIDENCE > 3.5
L.. Node06	No	595	19	96.9 %	3.1 %	EDUCATION isn't "10th" "11th" "12th" "1st-4th" "5th-6th" "7th"
L.. Node07	No	475	7	98.5 %	1.5 %	YRS_RESIDENCE <= 4.5
L.. Node08	No	244	7	97.2 %	2.8 %	AGE > 26.5
L.. Node16	No	44	0	100.0 %	0.0 %	BOOKKEEPING_APPLICATION <= .5
L.. Node17	No	200	7	96.6 %	3.4 %	BOOKKEEPING_APPLICATION > .5
L.. Node15	No	231	0	100.0 %	0.0 %	AGE <= 26.5
L.. Node18	No	120	12	90.9 %	9.1 %	YRS_RESIDENCE > 4.5



- ▶ From XML to relational
- ▶ **From XML to HTML**
- ▶ From relational to HTML
- ▶ Mixing relational with XML

From XML to HTML



The Oracle OE demo database

The warehouses.warehouse_spec XML column

```
SQL> select warehouse_spec from warehouses where warehouse_id=1;
```

```
WAREHOUSE_SPEC
```

```
-----  
<?xml version="1.0"?>  
<Warehouse>  
<Building>Owned</Building>  
<Area>25000</Area>  
<Docks>2</Docks>  
<DockType>Rear load</DockType>  
<WaterAccess>Y</WaterAccess>  
<RailAccess>N</RailAccess>  
<Parking>Street</Parking>  
<UClearance>10 ft</UClearance>  
</Warehouse>
```

Transforming XML into HTML against the Oracle demo database

We define an object on the warehouse_spec XML column
We include the xsl stylesheet as part of the object definition

The screenshot displays the Oracle SQL Developer interface for defining an object. The main window shows the 'Definition' tab for an object named 'Warehouse spec HTML' of type 'Long text'. The 'Select' field contains the following SQL statement:

```
XMLSERIALIZE(CONTENT  
XMLTRANSFORM(WAREHOUSES.WAREHOUSE_SPEC,  
XMLType{
```

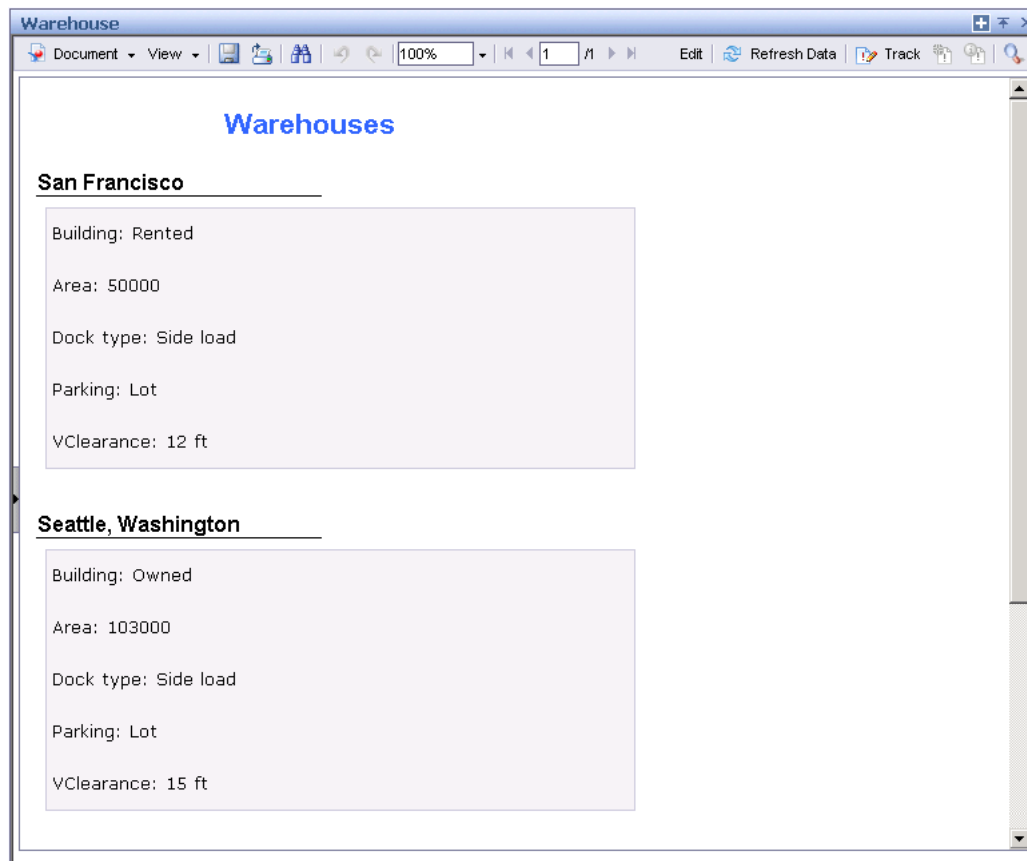
An 'Edit Select Statement of 'Warehouse spec HTML'' dialog box is open, showing the full SQL statement with an XSL stylesheet:

```
XMLSERIALIZE(CONTENT XMLTRANSFORM(WAREHOUSES.WAREHOUSE_SPEC,  
XMLType{  
'<?xml version="1.0" encoding="utf-8"?>  
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">  
<xsl:output method="html" encoding="utf-8" indent="yes" />  
<xsl:template match="/">  
<html>
```

A yellow callout bubble labeled 'Object format' points to a checkbox labeled 'Read As HTML', which is checked in the 'Advanced' tab of the object definition.

Transforming XML into HTML against the Oracle demo database

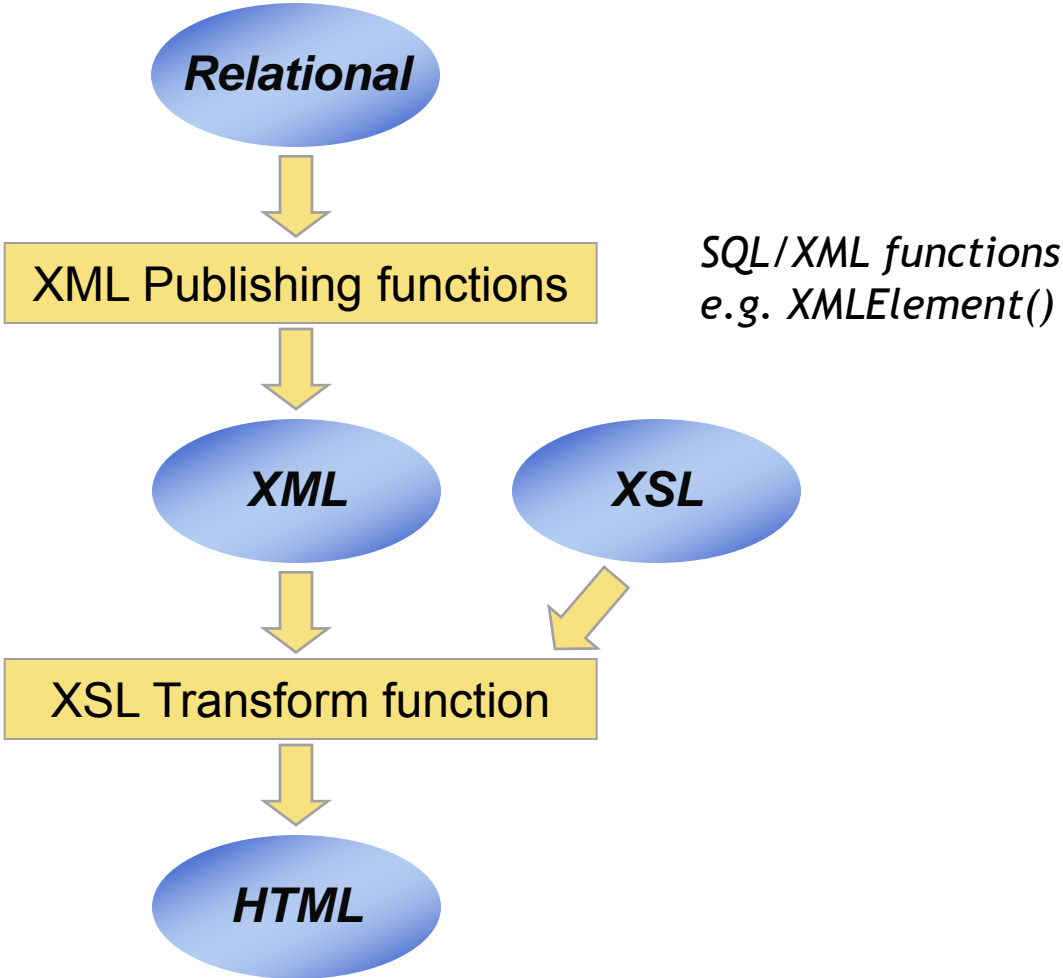
The warehouse spec rendered as HTML in WebI



Topics

- ▶ From XML to relational
- ▶ From XML to HTML
- ▶ **From relational to HTML**
- ▶ Mixing relational with XML

From relational to HTML



Generating an XML document from the DB2 sample database

```
CREATE VIEW ACTIVITY
AS
with
PRJR as (
SELECT
  PROJECT.PROJNAME as PROJECT,
  PRJ_OWNER.FIRSTNME || ' ' || PRJ_OWNER.LASTNAME as MANAGER,
  PROJECT.PRSTDATE as START_DATE,
  sum((EMPPROJACT.EMENDATE - EMPPROJACT.EMSTDATE) * EMPPROJACT.EMPTIME) as DAYS_SPENT,
  RANK() OVER (ORDER BY sum((EMPPROJACT.EMENDATE - EMPPROJACT.EMSTDATE) * EMPPROJACT.EMPTIME) DESC) as PRANK
FROM
  EMPPROJACT,
  EMPLOYEE PRJ_OWNER,
  PROJECT
WHERE
  EMPPROJACT.PROJNO=PROJECT.PROJNO
  AND PROJECT.RESPEMP=PRJ_OWNER.EMPNO
GROUP BY
  PROJECT.PROJNAME,
  PRJ_OWNER.FIRSTNME || ' ' || PRJ_OWNER.LASTNAME,
  PROJECT.PRSTDATE
)
SELECT 'Projects' as TOPIC, XMLDOCUMENT(XMLElement(NAME "activity", XMLAgg( XMLELEMENT(NAME "project",
XMLForest(PROJECT as "pname", MANAGER as "manager", START_DATE as "begin",
DAYS_SPENT as "days", PRANK as "rank"))))) as XDOC
FROM
  PRJR
;
```

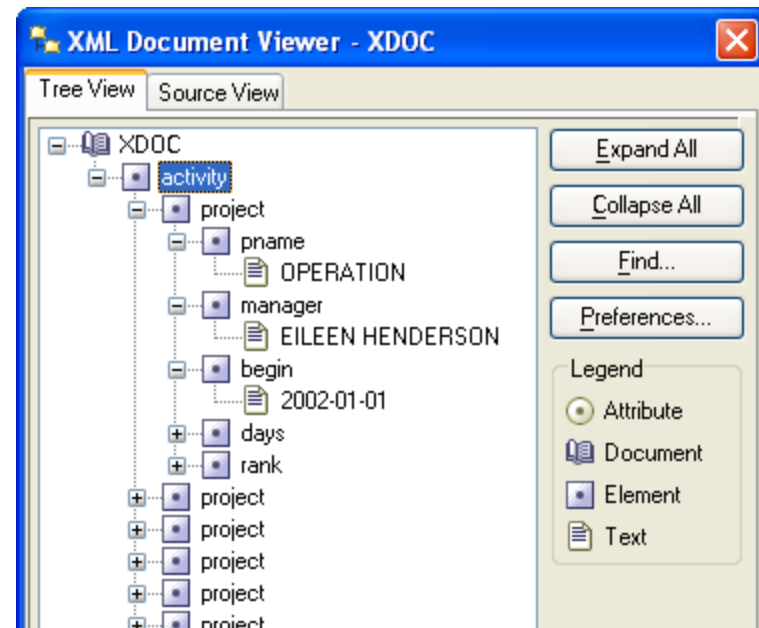
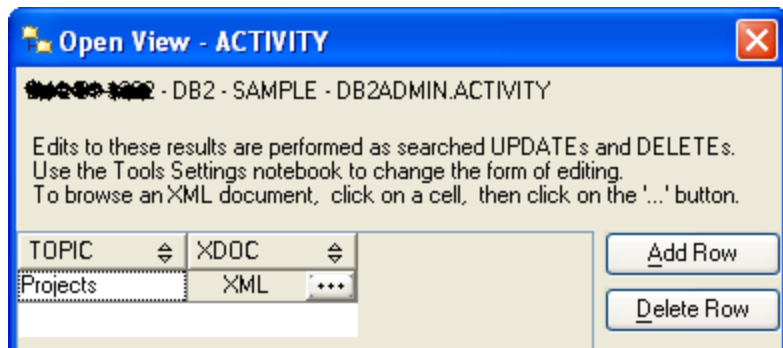
Building a view

Aggregating and ranking activity data by projects

Generating XML

Generating an XML document from the DB2 sample database

The 'activity' view returns an XML document as illustrated here in the DB2 Control Center



XSL templates

We create a table for storing XSL templates

```
create table XTEMPLATE
(
  TOPIC varchar(80),
  ID     integer,
  NAME  varchar(80),
  XSL   CLOB(4k)
)
;
```

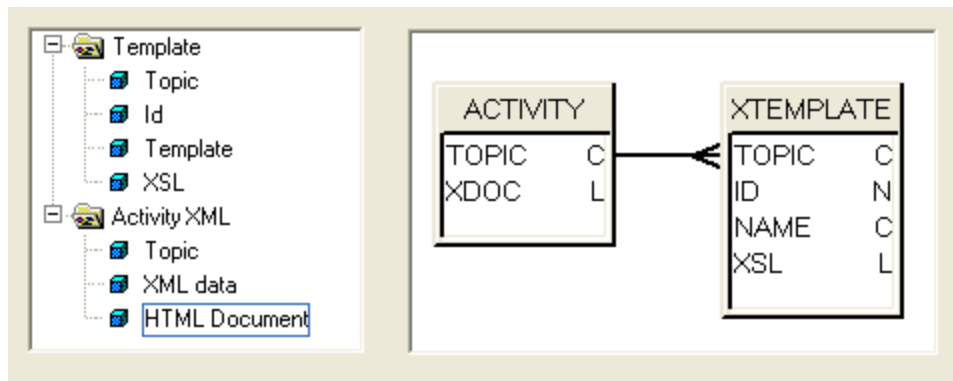
XSL templates

Filling the 'xtemplate' table

```
INSERT INTO XTEMPLATE VALUES ( 'Projects', 1, 'Alphabetical List', DB2XML.XMLCLOB(
'<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="html" encoding="utf-8" indent="yes" />
  <xsl:template match="/">
    <html>
      <style>
        .txt { font-family: Verdana, sans-serif; font-size: xx-small; vertical-align: center; text-align:left;
padding: .5em; background-color: beige;}
        .num { font-family: Verdana, sans-serif; font-size: xx-small; vertical-align: center; text-align:right;
padding: .5em; background-color: beige;}
        .hdr { font-family: Verdana, sans-serif; font-size: smaller;vertical-align: center; text-align:left;
padding: .5em; background-color: #9bd49d;}
        .ctr { font-family: Verdana, sans-serif; font-size: xx-small; vertical-align: center; text-align:center;
padding: .5em; background-color: beige;}
      </style>
      <body>
        <table>
          <tr>
            <th class="hdr">Project Name</th>
            <th class="hdr">Start Date</th>
            <th class="hdr">Manager</th>
            <th class="hdr">Days Spent</th>
          </tr>
          <xsl:for-each select="activity/project">
            <xsl:sort select="name" data-type="text" order="ascending"/>
          <tr>
            <td Class="txt"><em><xsl:value-of select="pname"/></em></td>
            <td Class="ctr"><xsl:value-of select="begin"/></td>
          ...
```

Turning XML into HTML

A Universe against the 'activity' view and the 'xtemplate' table



The screenshot shows the definition of a 'HTML Document' type. The 'Name' field is 'HTML Document' and the 'Type' is 'Long text'. The 'Description' field is empty. The 'Select' field contains the SQL query: 'DB2XML.XSLTtransformToClob(XMLSERIALIZE(ACTIVITY.XDOC AS CLOB(1M)), XTEMPLATE.XSL, 0)'. The interface includes tabs for 'Definition', 'Properties', 'Advanced', and 'Keys'.

Turning XML into HTML

Viewing the HTML document in Web



The screenshot shows a web browser window titled "Activity". The browser's address bar is empty, and the menu bar includes "Document", "View", "Save", "Find", "Undo", "Redo", "Zoom" (set to 100%), and "Refresh Data". The main content area displays a table with the following data:

Project Name	Start Date	Manager	Days Spent
OPERATION	2002-01-01	EILEEN HENDERSON	50,500
W L ROBOT DESIGN	2002-01-01	BRUCE ADAMSON	23,431
SYSTEMS SUPPORT	2002-01-01	THEODORE SPENSER	17,675
W L PROD CONT PROGS	2002-02-15	ELIZABETH PIANKA	11,331
W L PROGRAM DESIGN	2002-01-01	JENNIFER LUTZ	11,330
WELD LINE AUTOMATION	2002-01-01	CHRISTINE HAAS	10,700
APPLICATIONS SUPPORT	2002-01-01	WING LEE	10,100
DB/DC SUPPORT	2002-01-01	JASON GOUNOT	10,100
GENERAL ADMIN SYSTEMS	2002-01-01	EVA PULASKI	10,100
SCP SYSTEMS SUPPORT	2002-01-01	RAMLAL MEHTA	10,100
W L PROGRAMMING	2002-01-01	IRVING STERN	10,100
USER EDUCATION	2002-01-01	SALLY KWAN	5,750
OPERATION SUPPORT	2002-01-01	JOHN GEYER	2,525
ACCOUNT PROGRAMMING	2002-01-01	MARIA PEREZ	2,469
PAYROLL PROGRAMMING	2002-01-01	JAMES JEFFERSON	2,038
PERSONNEL PROGRAMMING	2002-01-01	DANIEL SMITH	1,181
QUERY SERVICES	2002-01-01	SALLY KWAN	950
WELD LINE PLANNING	2002-01-01	MICHAEL THOMPSON	814
ADMIN SERVICES	2002-01-01	CHRISTINE HAAS	200

Turning XML into HTML

The screenshot shows a software interface with a table of project data and a 'Prompts' dialog box. The table has three columns: 'Top 5 Projects', 'Days Spent', and 'Rank'. The 'Prompts' dialog box is open, showing a list of template names and a 'Run Query' button.

Top 5 Projects	Days Spent	Rank
<i>OPERATION</i>	50,500	1
<i>W L ROBOT DESIGN</i>	23,431	2
<i>SYSTEMS SUPPORT</i>	17,675	3
<i>W L PROD CONT PROGS</i>	11,331	4
<i>W L PROGRAM DESIGN</i>	11,330	5

Prompts
Reply to prompts before running the query.

Template name : Top 5 Projects

Run Query
Cancel

Refresh Values

Alphabetical List
Projects by Manager
Top 5 Projects

Template name
>> Top 5 Projects
<<

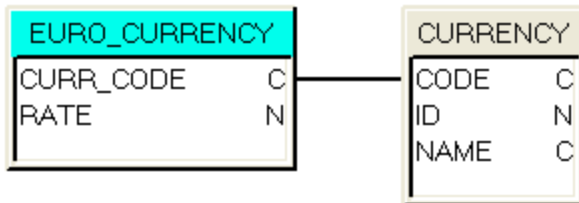
This WebI report offers three different layouts prompted to the end-user at refresh time

- ▶ From XML to relational
- ▶ From XML to HTML
- ▶ From relational to HTML
- ▶ **Mixing relational with XML**

Combining relational data with RSS feeds



In an Oracle Universe, we define a derived table on top of a RSS feed
We then join it to a relational table that contains the labels



SQL definition of the derived table

```
select substr(TXT,9,3) as CURR_CODE,  
to_number(replace(substr(TXT,14,8),',','')) as RATE  
from XMLTABLE('/rss/channel/item' passing  
HTTPURITYPE('http://currencysource.com/RSS/EUR.xml').getxml()  
columns  
  TXT varchar2(100) path '/item/title'  
)
```


Combining relational data with RSS feeds



An heterogeneous query joining together a relational table and RSS feed data

Name	Currency Code	Exchange Rate
Argentine Peso	ARS	4.055063
Australian Dollar	AUD	1.586423
Bahrain Dinar	BHD	0.505268
Botswana Pula	BWP	8.386356
Brazilian Real	BRL	2.604100
Brunei Dollar	BND	2.067317
Canadian Dollar	CAD	1.437496
Chilean Peso	CLP	708.973400
Chinese Yuan	CNY	10.241670
Colombian Peso	COP	2,635.310000
Cyprus Pound	CYP	0.583304
Czech Koruna	CZK	28.661120
Danish Krone	DKK	7.432293
Euro	EUR	1.000000
Hungarian Forint	HUF	247.545100
Icelandic Krona	ISK	84.679480
Indian Rupee	INR	55.050430

© 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP BusinessObjects Explorer, StreamWork, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.

Sybase and Adaptive Server, iAnywhere, Sybase 365, SQL Anywhere, and other Sybase products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Sybase, Inc. Sybase is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.