Extended Markup Language (XML)

- What is the "position" of my data?
- Line numbers, nodes,
- What the heck is this lab doing for me in the "REAL" world?

Exchanging data with structure…. Storing in a native XML data type - 249

1. sbs - 57 data type - nothing else

462 books - heap 419

2. 422 - xml indexes
3. 4 types - 425
    a. Primary
    b. Secondary - PATH
    c. Secondary - PROPERTY
    d. Secondary - VALUE
4. xml_deadlock_report event – 451/592

461 –

1. CASE 260
2. XQUERY – 221
3. 3 BASIC OPS
    a. Returning Results As XML with FOR XML
    b. Querying XML Data with XQuery
    c. Using the XML Data Type
4. Describe XML documents - 232
5. Convert relational data to XML.
6. Shred XML to tables
7. XML CDATA – 223
8. Prolog
9. XML fragments
10. XML document (can only have) with single root node – 223
11. Node – what am I vs dbase/table? Both have columns (attributes)etc…. boht get queried..
12. Element-centric - 224
13. Element/attribute/values
14. position/area – 224
15. data exchange – is text 225
16. order by – 228
17. Order of columns in SELECT – 228
18. XQUERY –
19. XPATH Expressions – 240
20. Absolute and Relative PATH
21. Aggregate functions – 239
22. Navigation – 240
23. Attribute :: - (double colon) – 241
24. Atomic values

25. Sequences
26. True exit of the expression – 241
27. Colon delimiters and navigation
28. If..then..else – 242
29. FLOWR – for loop - 239/243 - for, let, where, order by, and return
30. Return DDL triggers in XML – 250
31. Execution plan information
32. Five methods - 250

HOW TO CONVERT TABLE DATA TO XML AND XML TO TABLE

HOW TO CONVERT TABLE DATA TO XML AND VICEVERSA.

The below code explains how to convert the data in a table to xml form and then convert the xml back into table data.

Creating sample table with data:

```
CREATE TABLE tmpEmployee(ID INT,NAME VARCHAR(100))
GO

INSERT INTO tmpEmployee
SELECT 1,'Devi'
union
SELECT 2,'Prasad'
GO

SELECT * FROM tmpEmployee
GO
```

Below code converts data and schema of table tmpEmployee into XML, Then it uses the XML to convert it back into the table:

```
/*Below code converts Table SCHEMA & Data to XML*/

DECLARE @TableData XML,@TableSchema XML

SELECT @TableSchema = (
    select  column_name,
            data_type,
```

```
            case(is_nullable)
                when 'YES' then 'true'
                else 'false'
            end as is_nullable,
                        CHARACTER_MAXIMUM_LENGTH as Charlen
    from information_schema.columns [column]
    where table_name = 'tmpEmployee'
    for xml auto,root('Table')
        )

SELECT @TableData = (
SELECT *
FROM tmpEmployee Row
FOR XML AUTO, BINARY BASE64,root('TableData')
)

SELECT @TableSchema,@TableData




/*Below code converts XML to Table*/

if object_id('tempdb..#XMLColumns') is not null
drop table #XMLColumns

SELECT x.value('@column_name', 'sysname') AS column_name
,x.value('@data_type', 'sysname') AS data_type
,x.value('@is_nullable', 'VARCHAR(20)') AS is_nullable
,x.value('@Charlen', 'VARCHAR(20)') AS Charlen
into #XMLColumns
FROM @TableSchema.nodes('/Table/column') TempXML (x)

select * from #XMLColumns

DECLARE @SQL nVARCHAR(MAX) = 'SELECT '
```

```
SELECT @SQL = @SQL + '
x.value(''@'+column_name+''', '''+data_type+case when Charlen is null then '' else
'('+Charlen+')' end + ''''+') AS ['+column_name+'],'
from #XMLColumns

SET @SQL = LEFT(@SQL, LEN(@SQL) - 1)

SELECT @SQL = @SQL + ' FROM @TableData.nodes(''/TableData/Row'') TempXML (x)'

EXEC sp_executeSQl @SQL,N'@TableData xml',@TableData=@TableData
```

Output:

tmpEmployee:

```
ID          NAME
----------- ------------------------------------------------------------------------------
-------------------------
1           Devi
2           Prasad

(2 row(s) affected)
```

@TableSchema:

```
<Table>
  <column column_name="ID" data_type="int" is_nullable="true" />
  <column column_name="NAME" data_type="varchar" is_nullable="true" Charlen="100" />
</Table>
```

@TableData:

```
<TableData>
  <Row ID="1" NAME="Devi" />
  <Row ID="2" NAME="Prasad" />
</TableData>
```

Table generated from XML:

```
ID          NAME
----------- ------------------------------------------------------------------------------
-------------------------
```

```
1           Devi
2           Prasad


(2 row(s) affected)
```