-- Example: Untyped XML Column in Table

CREATE TABLE AdminDocs (

id int primary key,

xDoc Xml not null

)

DROP TABLE AdminDocs

-- Example: Inserting Data into Untyped XML Column

INSERT INTO AdminDocs VALUES (6,

'<catalog>

<product dept="WMN">

<number>557</number>

<name language="en">Fleece Pullover</name>

<colorChoices>navy black</colorChoices>

</product>

<product dept="ACC">

<number>563</number>

<name language="en">Floppy Sun Hat</name>

</product>

<product dept="ACC">

<number>443</number>

<name language="en">Deluxe Travel Bag</name>

</product>

<product dept="MEN">

<number>784</number>

<name language="en">Cotton Dress Shirt</name>

<colorChoices>white gray</colorChoices>

<desc>Our <i>favorite</i> shirt!</desc>

</product>

</catalog>')

INSERT INTO AdminDocs VALUES (2,

'<doc id="123">

<sections>

<section num="1"><title>XML Schema</title></section>

<section num="3"><title>Benefits</title></section>

<section num="4"><title>Features</title></section>

</sections>

</doc>')

INSERT INTO AdminDocs VALUES (3,

'<doc id="1234">

<sections>

<section num="1"><title>XML Schema</title></section>

<section num="3"><title>Benefits1</title><title>Benefits2</title><title>Benefits3</title></section>

<section num="4"><title>Features</title></section>

</sections>

</doc>')

-----------------------------------------------------------------------------------------XPath

select \*

from AdminDocs

-- Example: Using Query() Method

SELECT id, xDoc.query('/catalog')

FROM AdminDocs

SELECT id, xDoc.query('/catalog/product')

FROM AdminDocs

SELECT id, xDoc.query('/catalog/product/@dept')

FROM AdminDocs

SELECT id, xDoc.query('//product[]')

FROM AdminDocs

SELECT id, xDoc.query('/\*/product')

FROM AdminDocs

SELECT id, xDoc.query('/\*/product[@dept="WMN"]')

FROM AdminDocs

SELECT id, xDoc.query('/\*/child::product[attribute::dept="WMN"]')

FROM AdminDocs

SELECT id, xDoc.query('/\*/child::product[@dept="WMN"]')

FROM AdminDocs

SELECT id, xDoc.query('/\*/child::product[attribute::dept="WMN"]')

FROM AdminDocs

SELECT id, xDoc.query('//product[dept="WMN"]')

FROM AdminDocs

SELECT id, xDoc.query('descendant-or-self::product[attribute::dept="WMN"]')

FROM AdminDocs

SELECT id, xDoc.query('//product[number > 500]')

FROM AdminDocs

where id=6

SELECT id, xDoc.query('//product/number[. gt 500]')

FROM AdminDocs

where id=6

SELECT id, xDoc.query('/catalog/product[4]')

FROM AdminDocs

where id=6

SELECT id, xDoc.query('//product[number > 500][@dept="ACC"]')

FROM AdminDocs

where id=6

SELECT id, xDoc.query('//product[number > 500][1]')

FROM AdminDocs

where id=6

-----------------------------------------------------------------------------------------XQueary

SELECT xDoc.query(' for $prod in //product

let $x:=$prod/number

return $x')

FROM AdminDocs

where id=6

SELECT xDoc.query(' for $prod in //product/number

return $prod')

FROM AdminDocs

where id=6

--This Query writing to AdminDocs table. xDoc column has XML document.

--Query function has used. For the Query function I have given by Xquery of Flwor expression.

--It is telling for, variable, in, path expression. Since my XML document has 4 products.

--According to the number of products we get iterations.

--2nd Let, variable is there, for the 1st iteration 1st product is there, 1st product Number is taking to x variable.

--Print x variable content, It means content of the product (Number of the product)

SELECT xDoc.query(' for $prod in //product[@dept="ACC"]

let $x:=$prod/number

where $x=563

return $x')

FROM AdminDocs

where id=6

SELECT xDoc.query(' for $prod in //product

let $x:=$prod/number

where $x>500

return $prod')

FROM AdminDocs

where id=6

SELECT xDoc.query('for $prod in //product

let $x:=$prod/number

where $x>500

return (<Itemshbaba>{$x}</Itemshbaba>)')

FROM AdminDocs

where id=6

SELECT xDoc.query('for $x in //product/number

where $x>500

return (<Itemshbaba>{$x}</Itemshbaba>)')

FROM AdminDocs

where id=6

----------------------------------------------------------------------------------wrong

--SELECT xDoc.query(' for $prod in //product[number > 500]

--let $x:=$prod/number

--where $x>500

--return (<Item>{$x}</Item>)')

--FROM AdminDocs

--where id=6

--SELECT xDoc.query(' for $prod in //product[number > 500]

--let $x:=$prod/number

--where $x>500 AND //product/@dept=ACC

--return (<Item>{$x}</Item>)')

--FROM AdminDocs

--where id=6

----SELECT xDoc.query(' for $prod in //product[number > 500]

----let $x:=$prod[@dept]

----where $x=ACC

----return $prod')

----FROM AdminDocs

----where id=6

------------------------------------------------------------------------------- correct / important

SELECT xDoc.query('for $prod in //product[number<500]

let $x:=$prod/@dept

where $x="ACC"

return (<a> My No Is = {data($prod/number)} </a>)')

FROM AdminDocs

where id=6

---------------------------------------------------------------------------both gives same output

SELECT xDoc.query('for $prod in //product[number<500]

let $x:=$prod/@dept

where $x="ACC"

return $prod')

FROM AdminDocs

where id=6

SELECT xDoc.query('for $prod in //product

let $a:=$prod/number

let $b:=$prod/@dept

where $a<500 and $b="ACC"

return $prod')

FROM AdminDocs

where id=6

--------------------------------------------------------------------------------order by

SELECT xDoc.query('for $prod in //product

order by ($prod/number)[1]

return $prod')

FROM AdminDocs

WHERE id=6

---------------------------------------------------------------------------------same

SELECT xDoc.query('for $prod in //product

let $x:=$prod/number

where $x>500

return (<Item>{data($x)}</Item>)')

FROM AdminDocs

where id=6

SELECT xDoc.query('for $prod in //product

where $prod/number>500

return (<Item>{data($prod/number)}</Item>)')

FROM AdminDocs

where id=6

------------------------------------------------------------------------if else

SELECT xDoc.query('for $prod in //product

let $x:=$prod/number

return if ($x>500)

then <book>{$x}</book>

else <paper>{$x}</paper>')

FROM AdminDocs

where id=6

SELECT xDoc.query('for $prod in //product

let $x:=$prod/number

return if ($x>500)

then <book>{data($x)}</book>

else <paper>{data($x)}</paper>')

FROM AdminDocs

where id=6

----------------------------------------------------------------------------------------------------------Exists

--Select something if anotherthing is exists.

--This perticular documnt is asking to check this perticular path ('/doc[@id = 123]'),

--If any document found, to fitting to this path ('/doc[@id = 123]'), it will return 1. This 1 equal SQL WHERE clause 1.

--Then SQL Condition true

--Printing document id.

-- Example: Using Exist() Method

SELECT id

FROM AdminDocs

WHERE xDoc.exist ('/doc[@id = 123]') = 1

--<doc id="123">

SELECT id

FROM AdminDocs

WHERE xDoc.exist ('//product[@dept = "ACC"]') = 1

--<product dept="WMN">

------------------------------------------------------------------------------------Value

--Example: Using Value() Method

SELECT xDoc.value('(/doc//section[@num = 3]/title)[3]', 'varchar(100)')

FROM AdminDocs

------------------------------------------------------------Show

--Example: Insertion of Subtree into XML Instances

select \* from AdminDocs where id=2

select \* from AdminDocs where id=3

select \* from AdminDocs where id=6

--------------------------------------------------------------------------Update (Insert)

--Insert a branch, <section num="2">

UPDATE AdminDocs SET xDoc.modify('

insert

<section num="2">

<title>Background</title>

</section>

after (/doc//section[@num=1])[1]')

------------------------------------------------------------------------Update (Delete)

UPDATE AdminDocs SET xDoc.modify('

delete

//section[@num="2"]')