UNIVERSITY OF WESTMINSTER

5COSC002W DATABASE SYSTEMS
2021-2022 Tutorial 10
Querying XML documents – XQuery
Model Answer

Case Study

Westmuni is a university specialised in the delivery of wide range of IT courses to Undergraduate students.

Setting Up: Re-creating the XML database on BaseX

i. Access the xml_westmuni directory

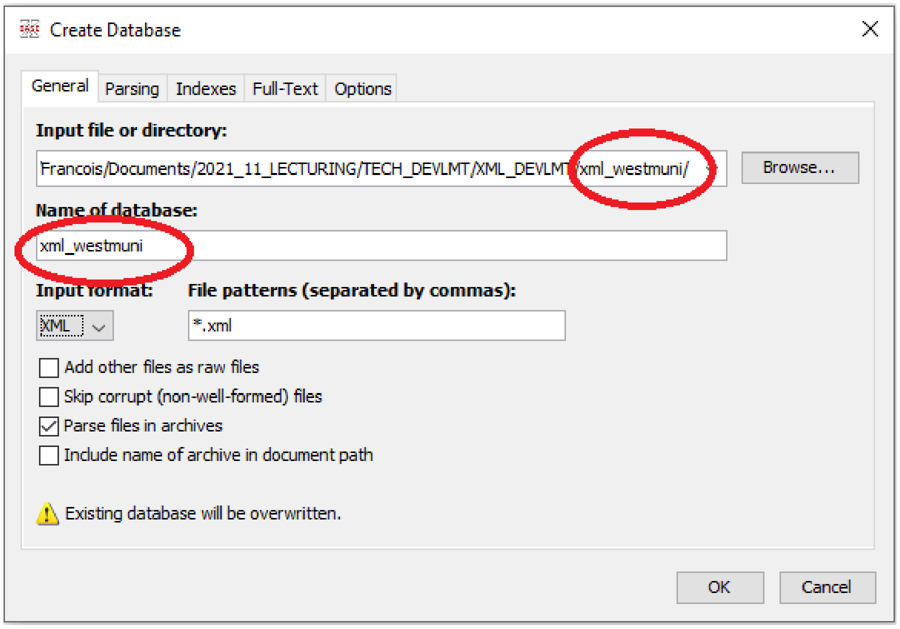
1. Access your previously created **directory** locally on your machine called **xml_westmuni**.

2. Check that your **xml_westmuni directory** now contains the following files (see tutorial 09)

   - The XML document **uni_marks.xml**: the XML source document.
   - The XPath file **uni_marks_XPath.xqy:** the file that contains the XPath expressions.
   - The XSLT file **uni_marks_modules.xsl:** the file that contain the transformations for the first rendering.
   - The XSLT file **uni_marks_students.xsl** (only if you managed to completely finish tutorial 09, it is OK if you have not, as it is not needed for this present tutorial 10): the file that contain the transformations for the second rendering.

ii. Re-Create your XML database on BaseX

1. Locate **BaseX** on AppsAnywhere https://appsanywhere.westminster.ac.uk and launch it.
2. Alternatively, download BaseX from https://basex.org/ if you are using your own machine.
3. Create a **New Database** on BaseX. Click on "Database" on the top nav bar and select "New".
4. Click browse and locate your xml_westmuni directory.
5. Name your **XML database** as **xml_westmuni** (same name as your directory) and click OK.


Create Database

Input file or directory: Francois/Documents/2021_11_LECTURING/TECH_DEVLMT/XML_DEVLMT/xml_westmuni/

Name of database: xml_westmuni

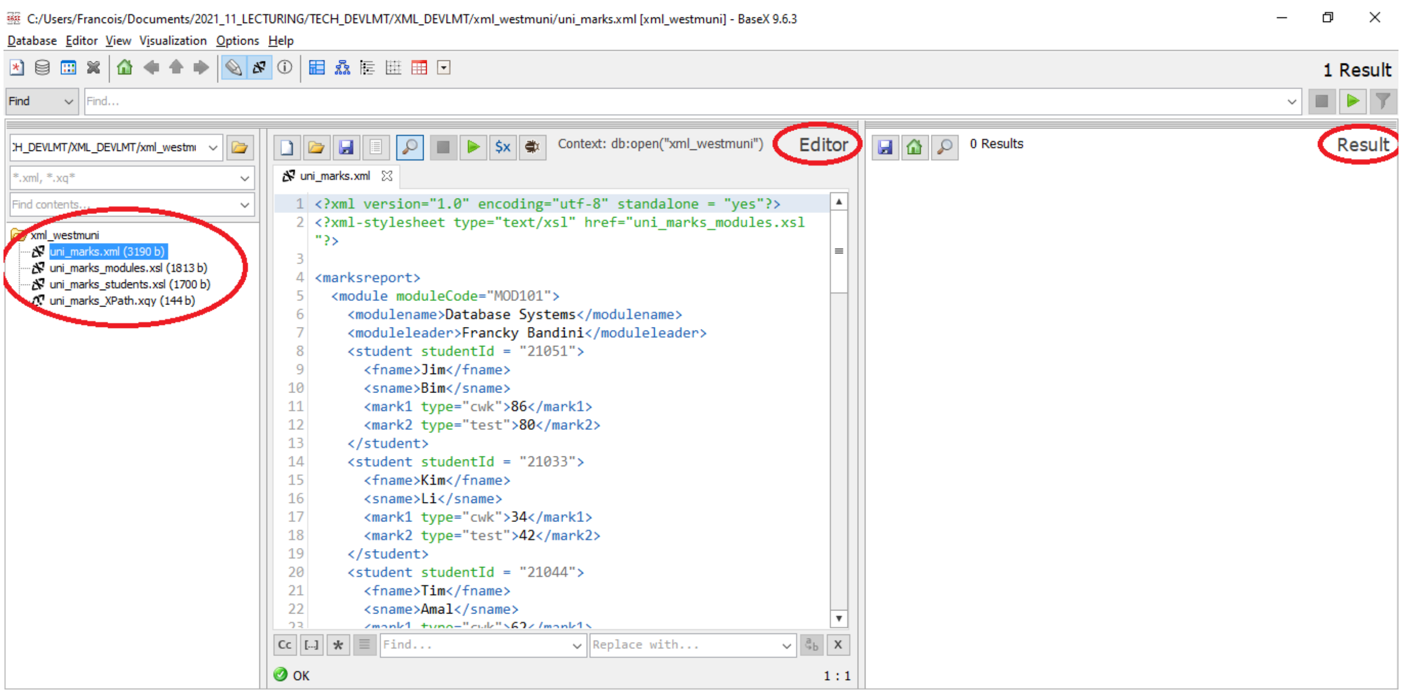
Input format: XML

File patterns (separated by commas): *.xml

- [ ] Add other files as raw files
- [ ] Skip corrupt (non-well-formed) files
- [x] Parse files in archives
- [ ] Include name of archive in document path

Existing database will be overwritten.

6. Use the browsing tool on the left hand-side to bring up your **xml_westmuni directory** and open the **uni_marks.xml** document in the editor. You can also use the View tab on the menu on the top nav bar to display the result pane on the right hand-side and hide any other panes, if you so wish (see tutorial 09).


Database Editor View Visualization Options Help
1 Result
Find
Find...
H_DEVLMT/XML_DEVLMT/xml_westmi
*.xml, *.xq*
Find contents...
xml_westmuni
uni_marks.xml (3190 b)
uni_marks_modules.xsl (1813 b)
uni_marks_students.xsl (1700 b)
uni_marks_XPath.xqy (144 b)
Context: db:open("xml_westmuni")
Editor
uni_marks.xml
0 Results
Result
Find...
Replace with...
OK
1 : 1

C:/Users/Francois/Documents/2021_11_LECTURING/TECH_DEVLMT/XML_DEVLMT/xml_westmuni/uni_marks.xml [xml_westmuni] - BaseX 9.6.3

```xml
<?xml version="1.0" encoding="utf-8" standalone = "yes"?>
<?xml-stylesheet type="text/xsl" href="uni_marks_modules.xsl
"?>

<marksreport>
  <module moduleCode="MOD101">
    <modulename>Database Systems</modulename>
    <moduleleader>Francky Bandini</moduleleader>
    <student studentId = "21051">
      <fname>Jim</fname>
      <sname>Bim</sname>
      <mark1 type="cwk">86</mark1>
      <mark2 type="test">80</mark2>
    </student>
    <student studentId = "21033">
      <fname>Kim</fname>
      <sname>Li</sname>
      <mark1 type="cwk">34</mark1>
      <mark2 type="test">42</mark2>
    </student>
    <student studentId = "21044">
      <fname>Tim</fname>
      <sname>Amal</sname>
```

Tutorial 10 Task 01: Creating your XQuery file on BaseX

1. Create a new file (of type .xqy) to write your XPath expressions. Click on the “New” icon to open a new tab.

2. Click on the “Save” icon to save the new file as **uni_marks_XQuery.xqy** in the same directory xml_westmuni.

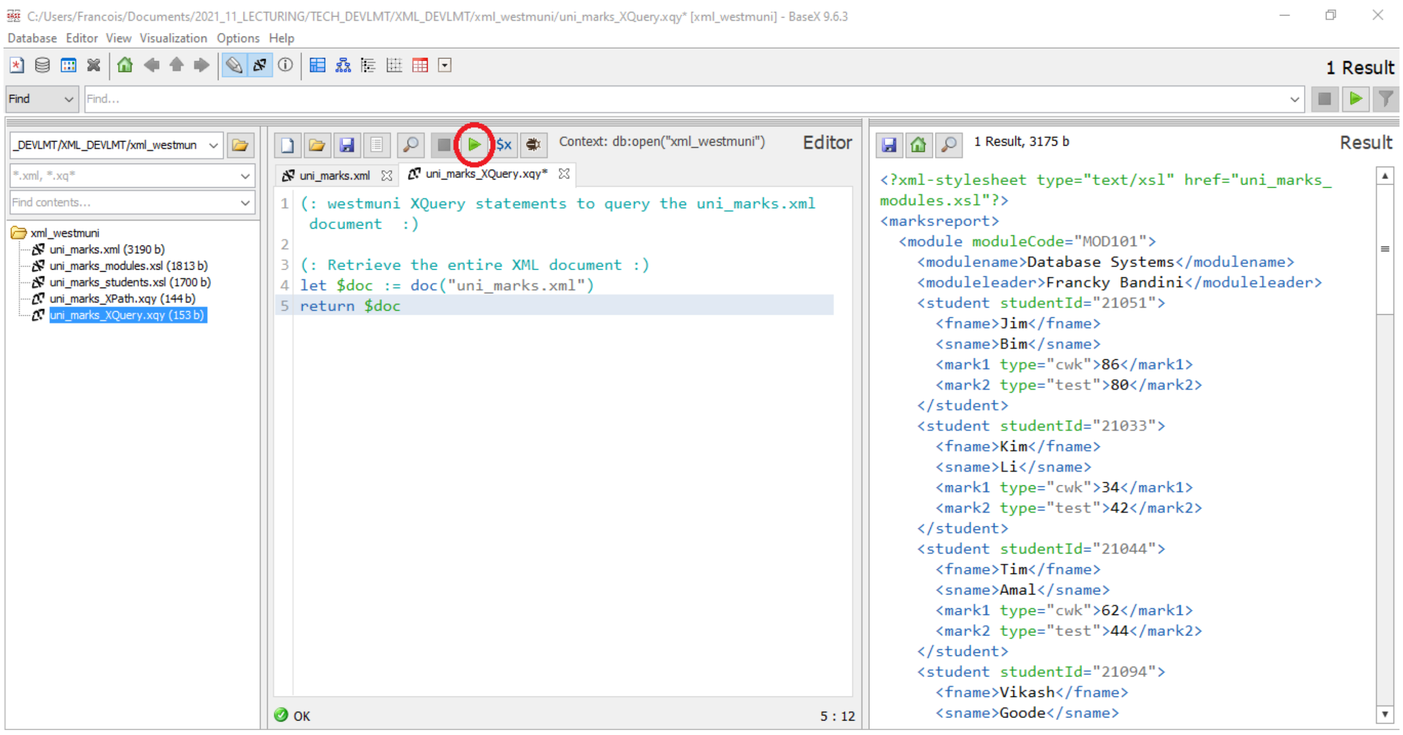
3. Type in a comment between (: and :) at the top of your **uni_marks_XQuery.xqy** file e.g.

   ```
   (: westmuni XQuery statements to query the uni_marks.xml document :)
   ```

4. Write your first XQuery statement in your editor to retrieve the whole XML document and a comment just above it.

   ```
   (: Retrieve the entire XML document :)
   let $doc := doc("uni_marks.xml")
   return $doc
   ```

5. Click on the green “run query” button immediately above your tab to execute it. View the result on the pane on the righthand-side. You should be able to view the full XML document being retrieved (see screenshot below).


C:/Users/Francois/Documents/2021_11_LECTURING/TECH_DEVLMT/XML_DEVLMT/xml_westmuni/uni_marks_XQuery.xqy* [xml_westmuni] - BaseX 9.6.3
Database Editor View Visualization Options Help
1 Result
Find
_DEVLMT/XML_DEVLMT/xml_westmun
*.xml, *.xq*
Find contents...
xml_westmuni
uni_marks.xml (3190 b)
uni_marks_modules.xsl (1813 b)
uni_marks_students.xsl (1700 b)
uni_marks_XPath.xqy (144 b)
uni_marks_XQuery.xqy (153 b)
Context: db:open("xml_westmuni")
Editor
uni_marks.xml
uni_marks_XQuery.xqy*
OK
5 : 12
1 Result, 3175 b
Result

```
(: westmuni XQuery statements to query the uni_marks.xml
 document  :)

(: Retrieve the entire XML document :)
let $doc := doc("uni_marks.xml")
return $doc
```

```
<?xml-stylesheet type="text/xsl" href="uni_marks_
modules.xsl"?>
<marksreport>
  <module moduleCode="MOD101">
    <modulename>Database Systems</modulename>
    <moduleleader>Francky Bandini</moduleleader>
    <student studentId="21051">
      <fname>Jim</fname>
      <sname>Bim</sname>
      <mark1 type="cwk">86</mark1>
      <mark2 type="test">80</mark2>
    </student>
    <student studentId="21033">
      <fname>Kim</fname>
      <sname>Li</sname>
      <mark1 type="cwk">34</mark1>
      <mark2 type="test">42</mark2>
    </student>
    <student studentId="21044">
      <fname>Tim</fname>
      <sname>Amal</sname>
      <mark1 type="cwk">62</mark1>
      <mark2 type="test">44</mark2>
    </student>
    <student studentId="21094">
      <fname>Vikash</fname>
      <sname>Goode</sname>
```

7. Continue writing your XQuery statements in the editor. For every expression, you can comment and uncomment it to execute it, one by one, by clicking on the green “run query” button.

Tutorial 10 Question 01: Simple XQuery statements

a) Return the modules nodes and descendants.

```
let $doc := doc("uni_marks.xml")
for $M in $doc//module
return $M
```

b) Return the module names and module leaders.

```
let $doc := doc("uni_marks.xml")
for $M in $doc//module
return ($M/modulename,$M/moduleleader)
```

c) Return the student nodes and descendants. 2 possible answers: one iterates through the student node, the other one iterates through the module node.

```
(: Answer 1 - Iterate through students :)
let $doc := doc("uni_marks.xml")
for $S in $doc//student
return $S

(: Answer 2 - Iterate through modules :)
let $doc := doc("uni_marks.xml")
for $M in $doc//module
return $M/student
```

d) Return the student first names and surnames of all students taking modules. 2 possible answers: one iterates through the student node, the other one iterates through the module node.

```
(: Answer 1 - Iterate through students :)
let $doc := doc("uni_marks.xml")
for $S in $doc//student
return ($S/sname, $S/fname)

(: Answer 2 - Iterate through modules :)
let $doc := doc("uni_marks.xml")
for $M in $doc//module
return ($M//sname, $M//fname)
```

Tutorial 10 Question 02: XQuery statements with condition on element

a) Return all the details for the module called "Database Systems".

```
let $doc := doc("uni_marks.xml")
let $modname := "Database Systems"
for $M in $doc//module
where $M/modulename = $modname
return $M
```

b) Return the surnames of the students on the "Database Systems" module. 2 possible answers: one iterates through the student node, the other one iterates through the module node.

```
(: Answer 01 - Iterate through module :)
let $doc := doc("uni_marks.xml")
let $modname := "Database Systems"
for $M in $doc//module
where $M/modulename = $modname
return $M//sname

(: Answer 02 - Iterate through students :)
let $doc := doc("uni_marks.xml")
let $modname := "Database Systems"
for $S in $doc//student
where $S/../modulename = $modname
return $S//sname
```

c) Return the details of the module and students who have scored more than 40 in the first component of the assessment.

```
let $doc := doc("uni_marks.xml")
let $passmark := 40
for $S in $doc//student
where $S/mark1 >=$passmark
return $S
```

d) Return the details of the module and students on the Database Systems module who have scored more than 40 in the first component of the assessment.

(: Iterate through students as you need to iterate for the condition that involves the descendent :)
```
let $doc := doc("uni_marks.xml")
let $passmark := 40
let $modname := "Database Systems"
for $S in $doc//student
where $S/../modulename = $modname and $S/mark1 >= $passmark
return ($S/../modulename, $S)
```

e) Return the details of the module and students on the Database Systems module who have scored more than 40 in the first component of the assessment.

```
let $doc := doc("uni_marks.xml")
let $passmark := 40
let $modname := "Database Systems"
for $S in $doc//student
where $S/../modulename = $modname and ($S/mark1 >= $passmark or $S/mark2 >= $passmark)
return $S
```

Tutorial 10 Question 03: XQuery statements with condition on attribute

a) Return the details of the students on the module identified by the code "MOD102".

```
(: Answer 01 - Iterate through students and use predicate at module level:)
let $doc := doc("uni_marks.xml")
let $modcode := "MOD102"
for $M in $doc//module
where $M[@moduleCode = $modcode]
return $M/student

(: Answer 02 - Iterate through students and use condition at module code level :)
let $doc := doc("uni_marks.xml")
let $modcode := "MOD102"
for $M in $doc//module
where $M/@moduleCode = $modcode
return $M/student

(: Answer 03  - Iterate through module :)
let $doc := doc("uni_marks.xml")
let $modcode := "MOD102"
for $S in $doc//student
where $S/../@moduleCode = $modcode
return $S
```

b) Return a list of students and their marks on the MOD102 module for the students who have scored more than 70 either on the first or second component.

```
let $doc := doc("uni_marks.xml")
let $modcode := "MOD102"
let $topmark := 70
for $S in $doc//student
where ($S/../@moduleCode = $modcode
and ($S/mark1 >= $topmark or $S/mark2 >= $topmark))
return $S
```

Tutorial 10 Question 04: XQuery statements with sequence functions

a) Write a report that retrieves the name of the module, the surname of the student, the mark scored in both components and a calculation of the final mark if every component is worth 50%.

(PTO)

```
let $doc := doc("uni_marks.xml")
for $S in $doc//student
return
<report>
      {$S/../modulename}
      {$S/sname}
      {$S/mark1}
      {$S/mark2}
      <finalmark>{0.5*($S/mark1+$S/mark2)}</finalmark>
</report>
```

b) Write a report that retrieves the name of the module and the number of students on each module.

```
let $doc := doc("uni_marks.xml")
for $M in $doc//module
return
<report>
      {$M/modulename}
      <nbofstudents>{count($M/student)}</nbofstudents>
</report>
```

c) Write a statistical report that provides the following info: for each module, the lowest, highest and average mark on the first component, as well as the lowest, highest and average mark on the second component. Use distinct values to group by and calculate min, max and average.

```
let $doc := doc("uni_marks.xml")
for $MC in distinct-values($doc//@moduleCode)
let $minMark1 := min($doc//module[@moduleCode = $MC]//mark1)
let $maxMark1 := max($doc//module[@moduleCode = $MC]//mark1)
let $avgMark1 := avg($doc//module[@moduleCode = $MC]//mark1)
let $minMark2 := min($doc//module[@moduleCode = $MC]//mark2)
let $maxMark2 := max($doc//module[@moduleCode = $MC]//mark2)
let $avgMark2 := avg($doc//module[@moduleCode = $MC]//mark2)
return
<stats>
      <modulecode>{$MC}</modulecode>
      <lowestmark1>{$minMark1}</lowestmark1>
      <highestmark1>{$maxMark1}</highestmark1>
      <averagemark1>{$avgMark1}</averagemark1>
      <lowestmark2>{$minMark2}</lowestmark2>
      <highestmark2>{$maxMark2}</highestmark2>
      <averagemark2>{$avgMark2}</averagemark2>
</stats>
```