

XQuery Notes

XQuery

- Used to retrieve information from XML documents
- XQuery is a language for finding and extracting elements and attributes from XML documents.
- Used in conjunction with Xpath

For-Let-Where-OrderBy-Return (FLWR)

- For / Let: Used for gathering nodes into sets from a series of queries.
- Where: For filtering nodes into the sets using conditions.
- Order By: For returning your sets in a particular order
- Return: How to return the identified nodes in the set.

LET

- LET <variable> := <xpath expression>, <xpath xpression>

e.g **let \$c:= doc("data/tcd.xml")/assessments/course/mark**

Example LET Clause

```
<?xml version="1.0"?>
<assessments>
  <student name="Smith">
    <mark thecourse="4BA5"> 99
    </mark>
    <mark thecourse="4BA1"> 75
    </mark>
  </student>
  <course name="4BA1"
    takenby="Smith, Jones">
    <mark>60</mark>
  </course>
  <course name="4BA5"
    takenby="Smith, Bond">
    <mark>70</mark>
  </course>
</assessments>
```

XML Source

```
let $c:=
doc("data/tcd.xml")/assessments/co
urse/mark
return
  <list_of_avg_course_marks>
  {$c}
  </list_of_avg_course_marks>
```

XQuery

Curly brackets {} are used for enclosed expressions and indicate that the expression enclosed in the return clause needs to be evaluated by the Xquery processor

```
<list_of_avg_course_marks>
  <mark>60</mark>
  <mark>70</mark>
</list_of_avg_course_marks>
```

Result

FOR

- FOR <variable> IN <xpath expression>, <xpath expression>, <xpath expression>

e.g for \$j in doc("data/tcd.xml")/assessments/course
return
("Course Node:", \$j)

Example FOR Clause

XML Source

```
<?xml version="1.0"?>
<assessments>
  <student name="Smith">
    <mark thecourse="4BA5"> 99
  </mark>
    <mark thecourse="4BA1"> 75
  </mark>
  </student>
  <course name="4BA1"
    takenby="Smith, Jones">
    <mark>60</mark>
  </course>
  <course name="4BA5"
    takenby="Smith, Bond">
    <mark>70</mark>
  </course>
</assessments>
```

XQuery

```
for $j in
doc("data/tcd.xml")/assessments/co
urse
return
("Course Node:", $j)
```

Result

```
Course Node: <course name="4BA1"
takenby="Smith, Jones">
  <mark>60</mark>
</course>
Course Node: <course name="4BA5"
takenby="Smith, Bond">
  <mark>70</mark>
</course>
```

Round Brackets useful for grouping sequence of Operations.

WHERE

- where contains(\$j/@takenby, "Bond")

Example WHERE Clause

XML Source

```
<?xml version="1.0"?>
<assessments>
  <student name="Smith">
    <mark thecourse="4BA5"> 99
  </mark>
    <mark thecourse="4BA1"> 75
  </mark>
  </student>
  <course name="4BA1"
    takenby="Smith, Jones">
    <mark>60</mark>
  </course>
  <course name="4BA5"
    takenby="Smith, Bond">
    <mark>70</mark>
  </course>
</assessments>
```

XQuery

```
for $j in
doc("data/tcd.xml")/assessments/co
urse
where contains($j/@takenby, "Bond")
return
  <Bond_courses_is>
    {string($j/@name)}
  </Bond_courses_is>
```

Result

```
<Bond_courses_is>4BA5</Bond_courses_is>
```

ORDER BY

- order by **\$x/enrolled descending**

Example ORDER BY clause

XML Source

```
<?xml version="1.0"?>
<studentdetails>
  <student name="Smith">
    <address> 101 Pine </address>
    <enrolled> 2011 </enrolled>
  </student>
  <student name="Bond">
    <address> 007 Fleming
  </address>
    <enrolled> 2012 </enrolled>
  </student>
</studentdetails>
```

XQuery

```
for $x in
doc("data/details.xml")/studentdet
ails/student
order by $x/enrolled descending
return
  <byyear>
    {$x}
  </byyear>
```

Result

```
<byyear>
  <student name="Bond">
    <address> 007 Fleming
  </address>
    <enrolled> 2012 </enrolled>
  </student>
</byyear>
<byyear>
  <student name="Smith">
    <address> 101 Pine </address>
    <enrolled> 2011 </enrolled>
  </student>
</byyear>
```

QUERYING OVER SEVERAL DOCUMENTS

-Can simultaneously access multiple documents

Querying over several **interlinked** documents

XML Source Tcd.xml

```
<?xml version="1.0"?>
<assessments>
  <student name="Smith">
    <mark thecourse="4BA5"> 99
    </mark>
    <mark thecourse="4BA1"> 75
    </mark>
  </student>
  <course name="4BA1"
  takenby="Smith, Jones">
    <mark>60</mark>
  </course>
  <course name="4BA5"
  takenby="Smith, Bond">
    <mark>70</mark>
  </course>
</assessments>
```

XML Source details.xml

```
<?xml version="1.0"?>
<studentdetails>
  <student name="Smith">
    <address> 101 Pine </address>
    <enrolled> 2001 </enrolled>
  </student>
  <student name="Bond">
    <address> 007 Fleming </address>
    <enrolled> 2002 </enrolled>
  </student>
</studentdetails>
```

XQuery

```
for $w in
doc("data/details.xml")/studentdet
ails/student,
  $x in
doc("data/tcd.xml")/assessments/st
udent
where $x/@name = $w/@name
return
  <studentpercourse>
    {$w/@name}
    {$w/address}
    {$x/mark/@thecourse}
  </studentpercourse>
```

Result

```
<studentpercourse name="Smith
thecourse="4BA5" thecourse="4BA1">
  <address> 101 Pine </address>
</studentpercourse>
```

CONDITIONAL STATEMENTS

-Can use conditional statements within your queries

Example Conditional clause

```
<?xml version="1.0"?>
<studentdetails>
  <student name="Smith">
    <address> 101 Pine </address>
    <enrolled> 2012</enrolled>
  </student>
  <student name="Bond">
    <address> 007 Fleming
  </address>
    <enrolled> 2011 </enrolled>
  </student>
</studentdetails>
```

XML Source

```
for $x in
doc("data/details.xml")/studentdet
ails/student
return
  <status>
    {$x/@name}
    {if ($x/enrolled = 2012)
    then "new student"
    else if ($x/enrolled < 2008)
      then "should be finished"
    else "student"}
  </status>
```

XQuery

```
<status name="Smith">
student</status>
<status name="Bond">
new student</status>
```

Result

QUANTIFIED EXPRESSIONS

Example Quantified Expression clauses

```
<?xml version="1.0"?>
<assessments>
  <student name="Ledwidge">
    <mark thecourse="4BA5"> 45
  </mark>
    <mark thecourse="4BA1"> 55
  </mark>
  </student>
  <student name="O'Neill">
    <mark thecourse="4BA5"> 85
  </mark>
</student> </assessments>
```

XML Source
tcd2.xml

```
<results>
{if (every $m in
doc("data/tcd2.xml")/assessments/s
tudent satisfies $m/mark > 60)
then "excellent results"
else if (some $t in
doc("data/tcd2.xml")/assessments/s
tudent satisfies $t/mark > 50)
then "average results"
else "bad results"
}
</results>
```

XQuery

```
<results>average results</results>
```

Result

FUNCTIONS

Example Function clause

```
<?xml version="1.0"?>
<assessments>
  <student name="Smith">
    <mark thecourse="4BA5"> 99
    </mark>
    <mark thecourse="4BA1"> 75
    </mark>
  </student>
  <course name="4BA1"
  takenby="Smith, Jones">
    <mark>60</mark>
  </course>
  <course name="4BA5"
  takenby="Smith, Bond">
    <mark>70</mark>
  </course>
</assessments>

<?xml version="1.0"?>
<assessments>
  <student name="Ledwidge">
    <mark thecourse="4BA5"> 45
  </mark>
    <mark thecourse="4BA1"> 55
  </mark>
  </student>
  <student name="ONeill">
    <mark thecourse="4BA5"> 85
  </mark> </student> </assessments>
```

XML Source
Tcd.xml

XML Source
tcd2002.xml

XQuery

```
declare function local:all_students()
{
  for $s in
  doc("data/tcd.xml")/assessments/student
  union
  doc("data/tcd2.xml")/assessments/studen
  t
  return
  <student>
    { $s/@name }
    { $s/mark/@thecourse }
  </student>
};

<all>
{local:all_students()}
</all>
```

Result

```
<all>
  <student name="Smith" thecourse="4BA5"
  thecourse="4BA1"/>
  <student name="Ledwidge" thecourse="4BA5"
  thecourse="4BA1"/>
  <student name="ONeill" thecourse="4BA5"/>
</all>
```

Example Function clause with param

```
<?xml version="1.0"?>
<assessments>
  <student name="Smith">
    <mark thecourse="4BA5"> 99
    </mark>
    <mark thecourse="4BA1"> 75
    </mark>
  </student>
  <course name="4BA1"
  takenby="Smith, Jones">
    <mark>60</mark>
  </course>
  <course name="4BA5"
  takenby="Smith, Bond">
    <mark>70</mark>
  </course>
</assessments>

<?xml version="1.0"?>
<assessments>
  <student name="Ledwidge">
    <mark thecourse="4BA5"> 45
  </mark>
    <mark thecourse="4BA1"> 55
  </mark>
  </student>
  <student name="ONeill">
    <mark thecourse="4BA5"> 85
  </mark> </student> </assessments>
```

XML Source
Tcd.xml

XML Source
tcd2.xml

XQuery

```
declare function local:
find_students($stuname as xs:string)
{
  for $s in
  doc("data/tcd.xml")/assessments/student
  union
  doc("data/tcd2.xml")/assessments/studen
  t
  where $stuname = string($s/@name)
  return
  <student>
    { $s/@name }
    { $s/mark/@thecourse }
  </student>
};

local:find_students("Smith")
```

Result

```
<student name="Smith" thecourse="4BA5"
thecourse="4BA1"/>
```