# Database Architectures

## Practical Assessment #2 (PR2):

## XML Extension

***Students:*** *Carlos Del Blanco Garcia*

*Jordi Bericat Ruz*

***Professor:*** *Maria Teresa Bordas Garcia*

***Term:*** *Autumn 2020-21 (Aula 1)*

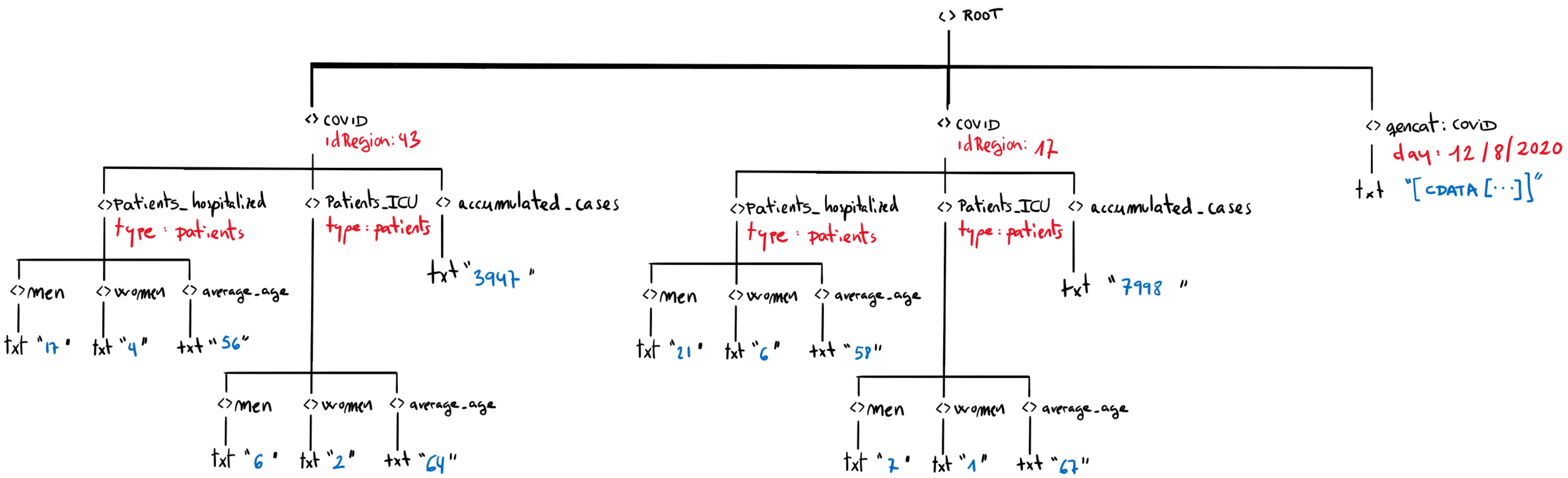
# Table of Contents

# Activity 1

## a)

*XML-tree Structure:*

Before proceeding with the xml file creation itself, we thought that It would be a good practice to graphically model the xml structure given in the assessment statement as an xml-tree[1]. This way we can have a”one-sight” outlook of the whole xml structure and therefore make the later translation into xml code easier:



[1] As suggested in bibliography [#1], page 42.

## *XML Code:*

```
<?xml version = "1.0" encoding = "ISO-8859-1" ?> <!-- See comments section [1] -->

<metadata <!-- See comments section [2] -->
    xmlns="http://www.uoc.edu/subjects/adb/ns" <!-- See comments section [3] -->
    xmlns:gencat="http://www.gencat.cat/dadesobertes/ns"> <!--comm. section [4]-->
    <COVID idregion="43">
        <patients_hospitalized type="patients">
            <men>17</men>
            <women>4</women>
            <average_age>56</average_age>
        </patients_hospitalized>
        <patients_ICU type="patients">
            <men>6</men>
            <women>2</women>
            <average_age>64</average_age>
        </patients_ICU>
        <accumulated_cases>3947</accumulated_cases>
    </COVID>
    <COVID idregion="17">
        <patients_hospitalized type="patients">
            <men>21</men>
            <women>6</women>
            <average_age>58</average_age>
        </patients_hospitalized>
        <patients_ICU type="patients">
            <men>7</men>
            <women>1</women>
            <average_age>67</average_age>
        </patients_ICU>
        <accumulated_cases>7998</accumulated_cases>
    </COVID>
    <gencat:COVID day="12/8/2020">  <!-- See comments section [5] -->
        <![!CDATA[...]]>    <!-- See comments section [6] -->
    </gencat:COVID>
</metadata>
```

```
<!--

 ################################################################################
 #                                                                              #
 #                                 COMMENTS SECTION                             #
 #                                                                              #
 ################################################################################

 [1] -> The prolog section of the xml document is the place where we can specify
        the encoding type (which in our case must be "ISO-8859-1").

 [2] -> I don't really know if we should use the keyword "root" instead of
        "metadata" to define the xml's tree root (most likely it doesn't matter,
        but better ask it in the forum to be in the safe side).

 [3] -> to set the default namespace we use the "xmlns" attribute.

 [4] -> we define a specific namespace alias (gencat) to avoid name clashes between
        the two elements with identical identification (COVID) but (probably)
        different application data structure.

 [5] -> Here we specify the alias "gencat" to this xml element (COVID) in order
        to refer to the custom namespace provided in the activity statement. This
        is required for the sake of properly integrate the external COVID xml data
        into our xml, due to the fact that this external data MIGHT NOT HAVE the
        very same structure than the defined in the default namespace, from which
        we set the default predefined xml dictionary for this xml document.

 [6] -> the structure of the external "not-xml" data is not provided, so we
        cannot define it (we use the [...] placeholder instead).
        (maybe we have to add a refernce to the "summary of the latest data")

-->
```

## b)

The main purpose of the extensive markup language (xml) is to stablish a proper communication mechanism among applications. To achieve this, it is necessary to strictly define a structure of elements (known as vocabulary) which implies a set rules and constraint. Here is where xml schemas come into play, since they allow defining that so-called vocabulary with a very high degree of details regarding the application data particularities.

That said, to define the xml schema that will stablish the required vocabulary and set of rules for the xml structure proposed in this activity statement, we will proceed as follows:

```
<!-- ###################  definition of the xml schema ################### -->

<?xml version = "1.0" encoding = "ISO-8859-1" ?>

<!-- see comments section [0] -->

<xsd:schema> <!-- see comments section [1] -->



<!-- ###################  definition of simple elements  ################## -->

<xsd:element name="id_type"> <!--see comments section [2.1.3]-->
    <xsd:simpleType>
        <xsd:restriction base="xsd:positiveInteger">
            <xsd:maxInclusive value="9999"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xs:element name="year" type="xs:integer"/> <!--see comments section [2.2.1]-->
<xs:element name="month" type="xs:integer"/> <!--see comments section [2.2.1]-->
<xs:element name="day" type="xs:integer"/> <!--see comments section [2.2.1]-->
<xs:element name="idRegion" type="xs:integer"/> <!--see comments section [2.3.1]-
->
<xs:element name="description" type="xs:string"/> <!--see comm. section [2.3.2]--
>
<xs:element name="hospitalized" type="xs:integer"/> <!--see comm. section[2.4.1]-
->
<xs:element name="ICU" type="xs:integer"/> <!-- see comments section [2.4.2]-->
<xsd:element name="gender"> <!--see comments section [2.4.3]-->
    <xsd:simpleType>
        <xsd:restriction base="xsd:NMTOKEN">
            <xsd:enumeration value="female" />
            <xsd:enumeration value="male" />
            <xsd:enumeration value="other" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

```
<!-- #####################    definition of attributes    ##################### -->

<xs:attribute name="id" type="xs:id_type"/> <!-- see comments section [2.1.3] -->


<!-- ##################### definition of complex elements ##################### -->

<xsd:element name="date"> <!-- see comments section [2.2] -->
    <xsd:complexType>
        <xsd:sequence>
          <xs:element ref="year"/>
          <xs:element ref="month"/>
          <xs:element ref="day"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="region"> <!-- see comments section [2.3] -->
    <xsd:complexType>
        <xsd:sequence>
          <xs:element ref="idRegion"/>
          <xs:element ref="description"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="patients"> <!-- see comments section [2.4] -->
    <xsd:complexType>
        <xsd:sequence>
            <xs:element ref="hospitalised"/>
            <xs:element ref="ICU"/>
            <xs:element ref="gender"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>


<!-- #####################          root element          ##################### -->

<xs:element name="COVID"> <!-- see comments section [2.1.1] -->
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="date"/>
      <xs:element ref="region"/>
      <xs:element ref="patients" maxOccurs="10"/>
    </xs:sequence>
    <xs:attribute ref="id" use="required"/> <!--comm. sections [2.1.2] & [2.1.3]-->
  </xs:complexType>
</xs:element>

<!—xml schema definition end -->

</xs:schema>
```

```
<!--

 ################################################################################
 #                                                                              #
 #                              COMMENTS SECTION                               #
 #                                                                              #
 ################################################################################

[0] -> First, we should decide on a design method to define the xml schema.
       As seen at Bibliography [#2], there are three available approaches:

       1- "Simplest-yet-messy" approach: This way, "to create the schema we could
          simply follow the structure in the XML document and define each element
          as we find it". (footnote: literal citation from Bibliography [#2],
          section: "Create an XML Schema").

       2- "Divided Schema" approach: "The next design method is based on defining
          all elements and attributes first, and then referring to them using the
          ref attribute" (footnote: literal citation from Bibliography [#2],
          section: "Divide the Schema"). This way gets easier to read and
          maintain the xml code in complex structures.

       3- "Use of Named Types" approach: "The third design method defines classes
          or types, that enables us to reuse element definitions"(footnote: literal
          citation from Bibliography [#2], section: "Named Types").

       After analyzing the characteristics of each of all three approaches, we
       decided to use the 2nd one (Divide the Schema) since we won't be upgrading
       the xml structure (the assessment statement says nothing about scalability
       and reusability of types), but at the same time we wanted to design a
       readable (it has to be assessed) as well as maintainable xml schema (so both
       team members / students who participate on its elaboration can better
       understand and eventually improve it).

[1] -> The activity statement does not give any information about namespaces
       (xmlns), hence we neither include any reference to it in the schema root
       element declaration, nor to the "targetNamespace" attribute (the XML Schema
       will be assigned to the NULL namespace).

[2] -> NOTES ABOUT ELEMENTS & ATTRIBUTES DEFINITIONS:

  [2.1] -> NODE "COVID":

    [2.1.1] -> The "COVID" node will be the root element of this xml schema.
               On the other hand, we'll be considering that its sub-elements will
               be appearing in the same order on the instance xml documents.
               Therefore, will have to use the primitive "sequence" in the
               "COVID" element definition.

    [2.1.2] -> ATTRIBUTE "id" ("is mandatory" constraint): it must be declared
               as a mandatory attribute since it contains a "foreign key" value.
               To do so, we must explicitly declare it with the "use"
               attribute, as well as the "required" attribute (references:
               Bibliography [#1], page 49 & Bibliography [#3]).

    [2.1.3] -> ATTRIBUTE "id" ("1-9999" range constraint): On the other hand,
               this attribute needs to be restricted to values in between 1-9999,
               thus, we also might have to declare this integer attribute as a
               simpleType and then apply a "range" restriction to its values by
               means of the "minInclusive" & "maxInclusive" attributes
               (references: Bibliography [#1], page 52).
```

**[2.2] -> NODE "date":** Must be a complex type defined as a SEQUENCE (references: Bibliography [#1], page 48).

**[2.2.1] -> ELEMENTS "year", "month" & "day" (in that order):** We could define these as simple type sub-elements (derived from integer built-in types) of the complex type element "date". This way we make sure these three elements contain an Integer value (and nothing else).

**[2.3] -> NODE "region":** The activity statement does not stablish any constraint about this element, hence we can simply declare it as a complexType with no constraints at all.

**[2.3.1] -> ELEMENT "idRegion":** "integer" built-in simple type (references: Bibliography [#1], page 50)

**[2.3.2] -> ELEMENT "description":** "string" built-in simple type (references: Bibliography [#1], page 50)

**[2.4] -> NODE "patients":** Since this node can only be repeated as much as 10 times, we must stablish the cardinality of this complexType element by means of the minOccurs and maxOccurs attributes (references: Bibliography [#1], page 49).

**[2.4.1] -> ELEMENT "hospitalized":** "integer" built-in simple type (references: Bibliography [#1], page 50)

**[2.4.2] -> ELEMENT "ICU":** "integer" built-in simple type (references: Bibliography [#1], page 50)

**[2.4.3] -> ELEMENT "gender":** Must be a simple Type derived from an NTOKEN existing built-in type defined as an ENUMERATION (references: Bibliography [#1], page 48)

# Activity 2

## a)

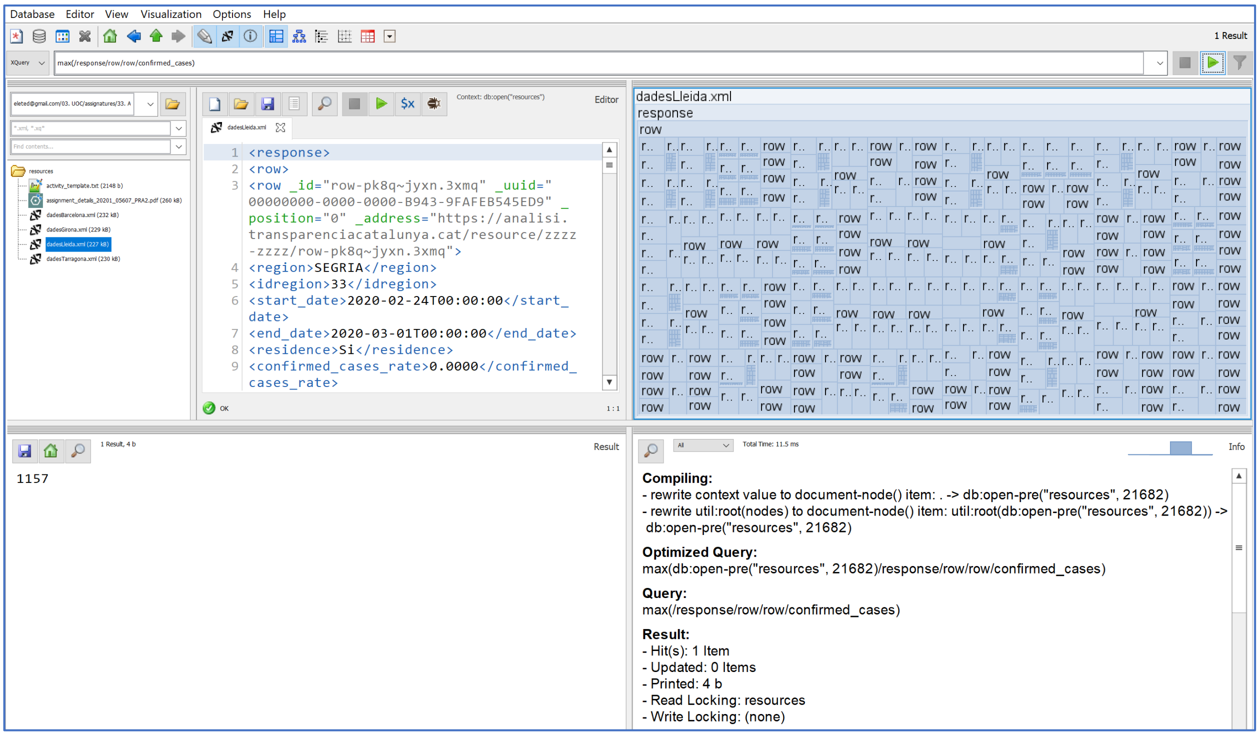
To define an *xPath* expression that complies with the constraints stablished on this activity statement, we came-up with the following strategy:

### STEP 1 → DEFINE A XPATH SUB-QUERY

First, we find a way to retrieve the "*confirmed_cases*" node that contains the maximum[2] value among all repetitions of the same element in the provided xml:

```
max(/response/row/row/confirmed_cases)
```

If we run the above's *xPath* expression, the only result we got is actually the correct "*confirmed_cases*" node with maximum value:


Database Editor View Visualization Options Help
1 Result
XQuery
eleted@gmail.com/03. UOC/assignatures/33. A
*.xml, *.xq*
Find contents...
resources
activity_template.txt (2148 b)
assignment_details_20201_05607_PRA2.pdf (260 kB)
dadesBarcelona.xml (232 kB)
dadesGirona.xml (229 kB)
dadesLleida.xml (227 kB)
dadesTarragona.xml (230 kB)
Editor
dadesLleida.xml
OK
1 : 1
1 Result, 4 b
Result
All
Total Time: 11.5 ms
Info
dadesLleida.xml
response
row

```
max(/response/row/row/confirmed_cases)
```

Context: db:open("resources")

```
<response>
<row>
<row _id="row-pk8q~jyxn.3xmq" _uuid="00000000-0000-0000-B943-9FAFEB545ED9" _position="0" _address="https://analisi.transparenciacatalunya.cat/resource/zzzz-zzzz/row-pk8q~jyxn.3xmq">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_date>
<end_date>2020-03-01T00:00:00</end_date>
<residence>Si</residence>
<confirmed_cases_rate>0.0000</confirmed_cases_rate>
```

```
1157
```

**Compiling:**
- rewrite context value to document-node() item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node() item: util:root(db:open-pre("resources", 21682)) -> db:open-pre("resources", 21682)

**Optimized Query:**
max(db:open-pre("resources", 21682)/response/row/row/confirmed_cases)

**Query:**
max(/response/row/row/confirmed_cases)

**Result:**
- Hit(s): 1 Item
- Updated: 0 Items
- Printed: 4 b
- Read Locking: resources
- Write Locking: (none)

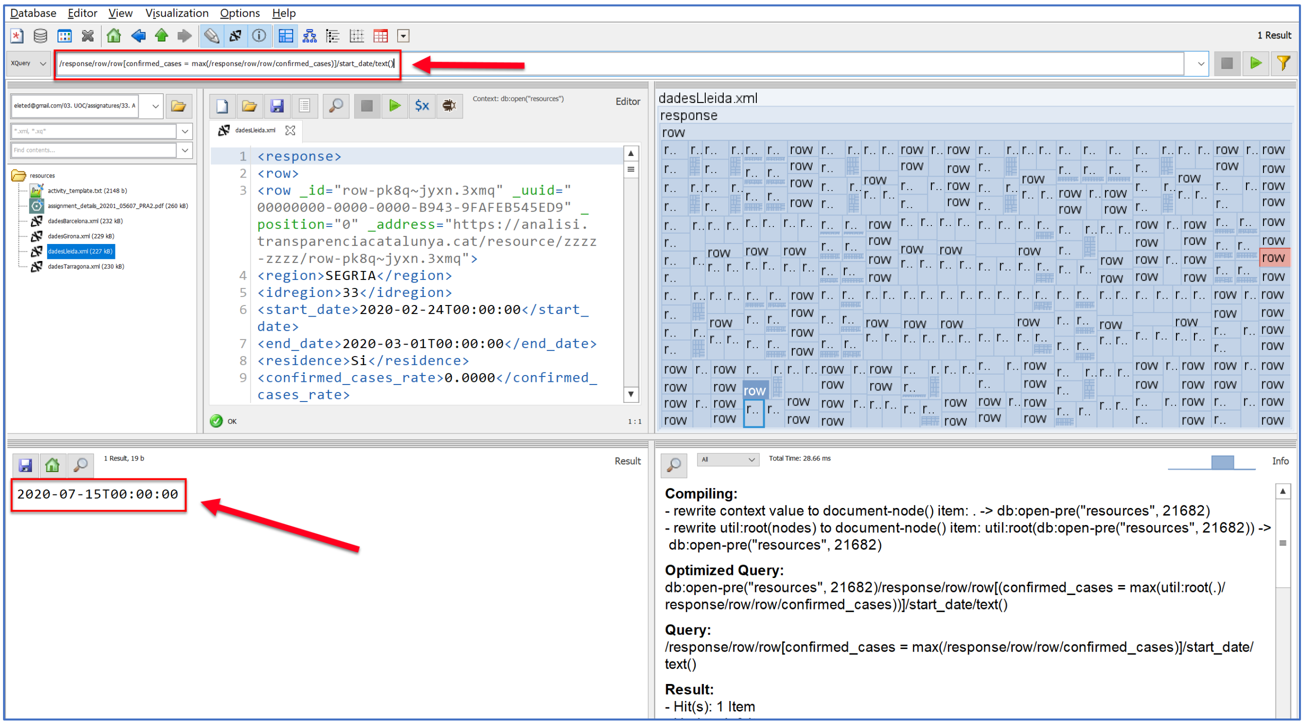
[2] *The xPath 2.0 version includes the **max()** built-in function which allowed us to select the maximum value of an element of the hierarchy, compared with other repetitions of the same element (references: Bibliography [#4]). However, In order to run xPath 2.0 expressions, we also had to download the newest version of the "BaseX" xml processor from Bibliography [#5], (the one provided in the subject's resources area was unstable in our systems).*

## STEP 2 → XPATH MAIN EXPRESSION

On this step, first we should stablish the direction on which the evaluation is going to proceed by setting the *axe*[3] (**green**) in the path expression. In our scenario, we will be going down in the hierarchy (from top) to its 4th level (*/response/row/row/start_date/*). Afterwards, we just have to tweak the *predicate* expression (**red** + **pink**) in order to retrieve the element that matches the xpath sub-query (**pink**) return value defined in the step #1 (that is; the maximum value contained in a "*confirmed_cases*" element). Finally, to exclude all the nodes selected by the *axe* but the *start-date* element, we will have to specify it in the expression's *node-test*[4] (yellow). Putting it all together we obtain the following *xPath* expression:

```
/response/row/row[confirmed_cases = max(/response/row/row/confirmed_cases)]/start_date/text()
```

Finally, if we run it in the xml processor, we get the requested information:



```
/response/row/row[confirmed_cases = max(/response/row/row/confirmed_cases)]/start_date/text()
```

eleted@gmail.com/03. UOC/assignatures/33. A

resources
- activity_template.txt (2148 b)
- assignment_details_20201_05607_PRA2.pdf (260 kB)
- dadesBarcelona.xml (232 kB)
- dadesGirona.xml (229 kB)
- dadesLleida.xml (227 kB)
- dadesTarragona.xml (230 kB)

Context: db:open("resources")

dadesLleida.xml

```
<response>
<row>
<row _id="row-pk8q~jyxn.3xmq" _uuid="
00000000-0000-0000-B943-9FAFEB545ED9" _
position="0" _address="https://analisi.
transparenciacatalunya.cat/resource/zzzz
-zzzz/row-pk8q~jyxn.3xmq">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_
date>
<end_date>2020-03-01T00:00:00</end_date>
<residence>Si</residence>
<confirmed_cases_rate>0.0000</confirmed_
cases_rate>
```

OK 1 : 1

1 Result, 19 b

```
2020-07-15T00:00:00
```

Total Time: 28.66 ms

**Compiling:**
- rewrite context value to document-node() item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node() item: util:root(db:open-pre("resources", 21682)) -> db:open-pre("resources", 21682)

**Optimized Query:**
db:open-pre("resources", 21682)/response/row/row[(confirmed_cases = max(util:root(.)/response/row/row/confirmed_cases))]/start_date/text()

**Query:**
/response/row/row[confirmed_cases = max(/response/row/row/confirmed_cases)]/start_date/text()

**Result:**
- Hit(s): 1 Item

You will find the .txt file requested for this activity on the delivery zip file (*/activity_2/activity_2A.txt*)

[3] *Bibliography [#1], page 57*

[4] *Bibliography [#1], page 59*

## b)

To simplify the process of defining the xPath expression requested in this activity, we are going to follow a "divide & conquer" strategy:

### Step #1:

Right now, we will only retrieve the “*confirmed_cases*” element value corresponding to the 2nd previous element (row) to the one that contains the maximum “*confirmed_cases*” element value (we can achieve this simply by modifying the *xPath* expression defined in the activity 2-A). The difference is that we now include in our selection a set of data belonging to a previous xml element (from the evaluation point of view) by means of the **"preceding::row[2]"**[5] “*axe component*” (**blue**). More precisely, this axe allows us to go back two elements in the evaluation direction, starting from the element selected by the left part of the axe (**dark green**). Finally, we add to the final xPath expression axe (underlined) a last “*axe component*” (**light green**) to indicate the next level in the hierarchy we want to select (“*confirmed_cases*” node) along with the node-test (**yellow**) that allows us to retrieve only the value we need. Putting it all together we get the following *xPath* expression:

```
/response/row/row[confirmed_cases =
max(/response/row/row/confirmed_cases)]/preceding::row[2]/confirmed_cases/text()
```

### Step #2:

At this point we have the means to individually obtain the necessary values to calculate the increment of confirmed cases between date ranges (that is, by using the *xPath* expressions we defined in the activity 2-A and the Step #1 on this activity). To implement this calculation, we can use the operators defined in the xPath specification[6] to calculate the increment of confirmed cases (minus operation). In “pseudo-code”, what we are going to perform is the following subtract operation:

```
((xPath_from_activity2A)-(xPath_from_step#1))
```

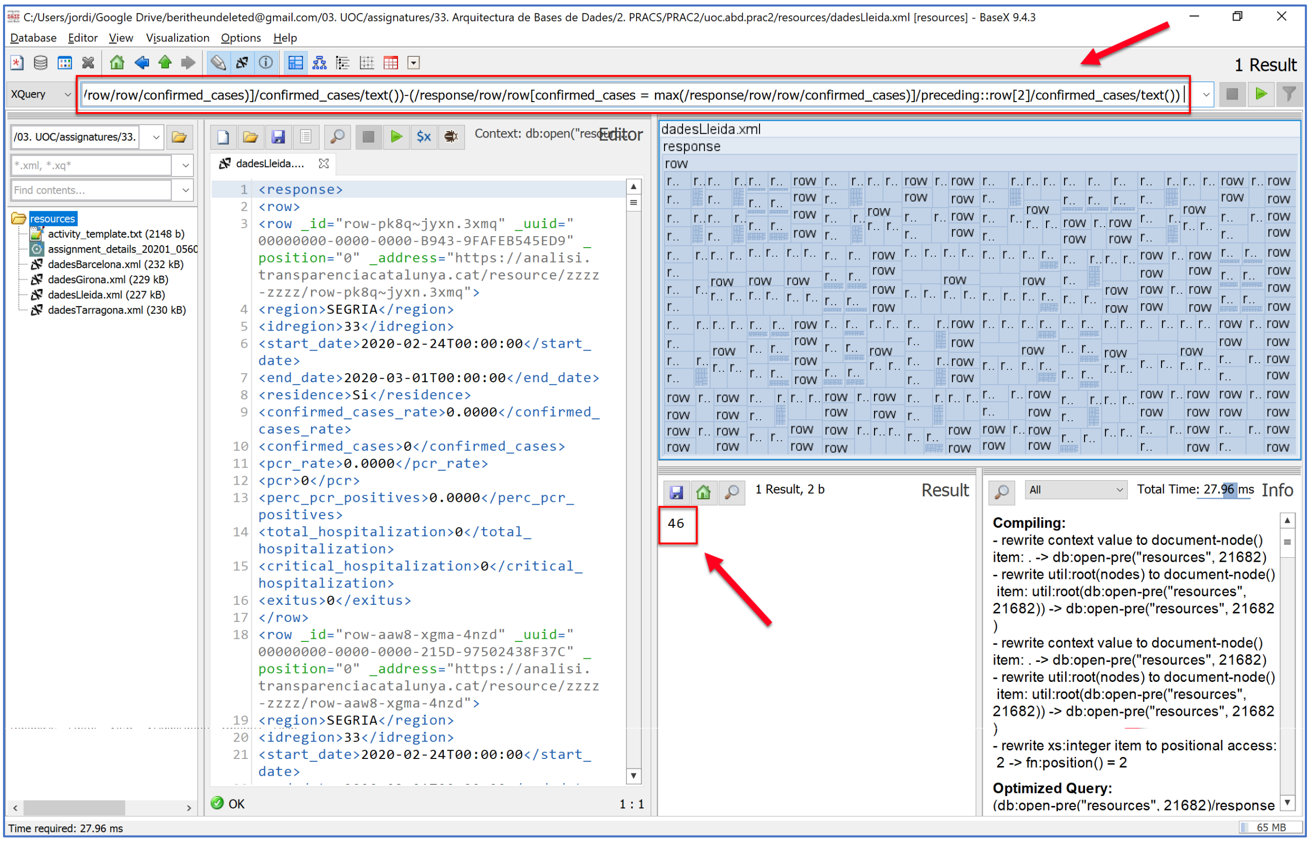
[5] *References: Bibliography [#6]*

[6] *References: Bibliography [#1], page 59*

In the above's template, replacing the *xPath* "placeholders" (in color) for the actual expressions will result in the following *xPath* "composite" expression:

```
(/response/row/row[confirmed_cases =
max(/response/row/row/confirmed_cases)]/confirmed_cases/text())-
(/response/row/row[confirmed_cases =
max(/response/row/row/confirmed_cases)]/preceding::row[2]/confirmed_cases/text
())
```

As we can see below, the result we obtain after running it is exactly what we were looking for:



C:/Users/jordi/Google Drive/beritheundeleted@gmail.com/03. UOC/assignatures/33. Arquitectura de Bases de Dades/2. PRACS/PRAC2/uoc.abd.prac2/resources/dadesLleida.xml [resources] - BaseX 9.4.3

XQuery: `/row/row/confirmed_cases)]/confirmed_cases/text())-(/response/row/row[confirmed_cases = max(/response/row/row/confirmed_cases)]/preceding::row[2]/confirmed_cases/text())`

```
<response>
<row>
<row _id="row-pk8q~jyxn.3xmq" _uuid="
00000000-0000-0000-B943-9FAFEB545ED9" _
position="0" _address="https://analisi.
transparenciacatalunya.cat/resource/zzzz
-zzzz/row-pk8q~jyxn.3xmq">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_
date>
<end_date>2020-03-01T00:00:00</end_date>
<residence>Si</residence>
<confirmed_cases_rate>0.0000</confirmed_
cases_rate>
<confirmed_cases>0</confirmed_cases>
<pcr_rate>0.0000</pcr_rate>
<pcr>0</pcr>
<perc_pcr_positives>0.0000</perc_pcr_
positives>
<total_hospitalization>0</total_
hospitalization>
<critical_hospitalization>0</critical_
hospitalization>
<exitus>0</exitus>
</row>
<row _id="row-aaw8-xgma-4nzd" _uuid="
00000000-0000-0000-215D-97502438F37C" _
position="0" _address="https://analisi.
transparenciacatalunya.cat/resource/zzzz
-zzzz/row-aaw8-xgma-4nzd">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_
date>
```

1 Result, 2 b

```
46
```

Total Time: 27.96 ms

```
Compiling:
- rewrite context value to document-node()
item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node()
 item: util:root(db:open-pre("resources",
21682)) -> db:open-pre("resources", 21682
)
- rewrite context value to document-node()
item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node()
 item: util:root(db:open-pre("resources",
21682)) -> db:open-pre("resources", 21682
)
- rewrite xs:integer item to positional access:
 2 -> fn:position() = 2

Optimized Query:
(db:open-pre("resources", 21682)/response
```

Time required: 27.96 ms

## step #3:

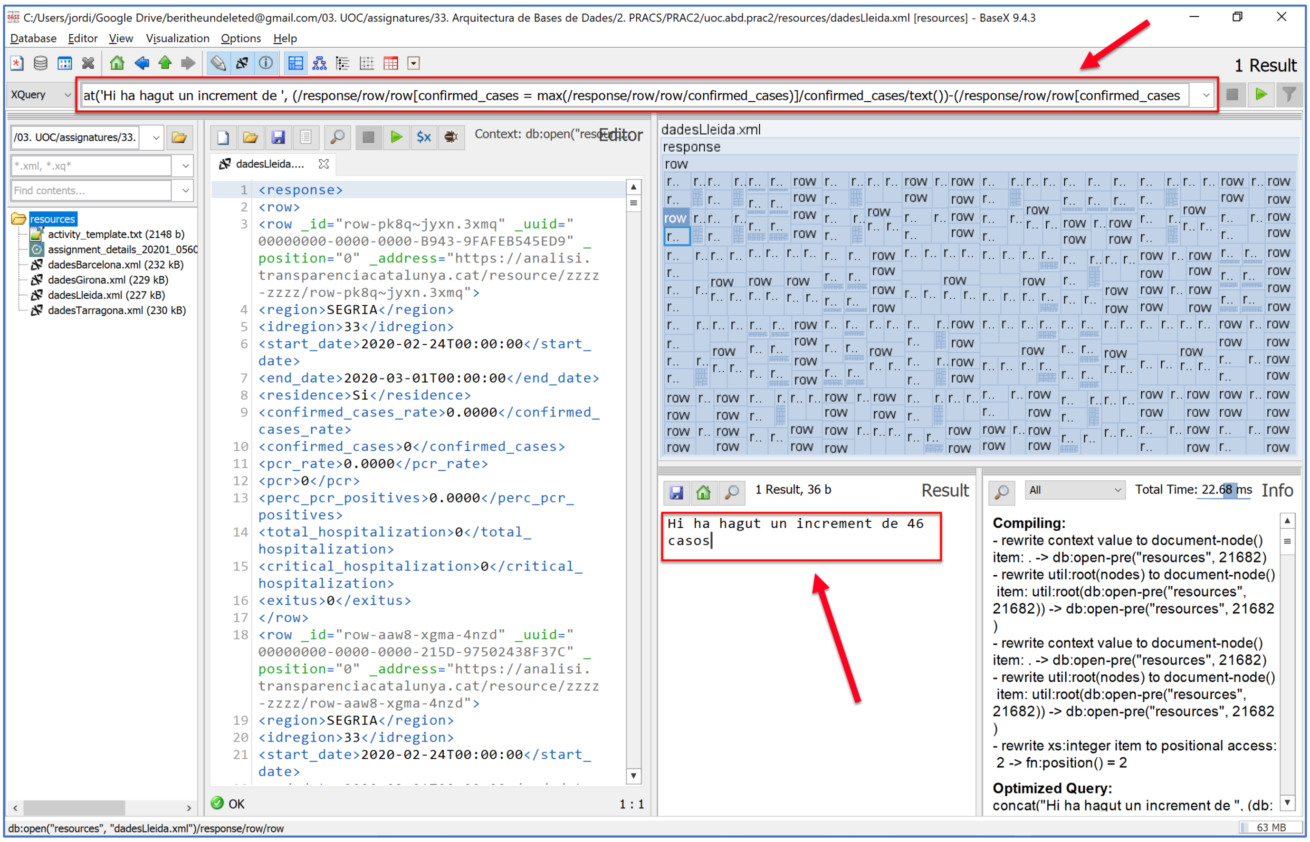
This step consists on using the built-in xPath *concat()* function to concatenate different xPath expressions to generate the output as requested in the assessment statement (we still do not include the xPath expression defined in the previous step, since we want to describe the whole process in a more detailed way):

```
concat('Hi ha hagut un increment de ',step-2_xPath_expression,' casos')
```

Finally, replacing the xPath *placeholder* (**pink**) for the expression defined in the step #2 will result in the *xPath* expression that gives us exactly the result requested on this assessment statement.

```
concat('Hi ha hagut un increment de ',(/response/row/row[confirmed_cases =
max(/response/row/row/confirmed_cases)]/confirmed_cases/text())-
(/response/row/row[confirmed_cases =
max(/response/row/row/confirmed_cases)]/preceding::row[2]/confirmed_cases/
text()),' casos')
```

If we run the xPath expression in the XML processor we get the expected result:



C:/Users/jordi/Google Drive/beritheundeleted@gmail.com/03. UOC/assignatures/33. Arquitectura de Bases de Dades/2. PRACS/PRAC2/uoc.abd.prac2/resources/dadesLleida.xml [resources] - BaseX 9.4.3

Database Editor View Visualization Options Help
1 Result
XQuery
Context: db:open("resources
Editor
dadesLleida....
/03. UOC/assignatures/33.
*.xml, *.xq*
Find contents...
resources
activity_template.txt (2148 b)
assignment_details_20201_0560
dadesBarcelona.xml (232 kB)
dadesGirona.xml (229 kB)
dadesLleida.xml (227 kB)
dadesTarragona.xml (230 kB)
dadesLleida.xml
response
row
Result
All
Info
63 MB

```
at('Hi ha hagut un increment de ', (/response/row/row[confirmed_cases = max(/response/row/row/confirmed_cases)]/confirmed_cases/text())-(/response/row/row[confirmed_cases
```

```
<response>
<row>
<row _id="row-pk8q~jyxn.3xmq" _uuid="00000000-0000-0000-B943-9FAFEB545ED9" _position="0" _address="https://analisi.transparenciacatalunya.cat/resource/zzzz-zzzz/row-pk8q~jyxn.3xmq">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_date>
<end_date>2020-03-01T00:00:00</end_date>
<residence>Si</residence>
<confirmed_cases_rate>0.0000</confirmed_cases_rate>
<confirmed_cases>0</confirmed_cases>
<pcr_rate>0.0000</pcr_rate>
<pcr>0</pcr>
<perc_pcr_positives>0.0000</perc_pcr_positives>
<total_hospitalization>0</total_hospitalization>
<critical_hospitalization>0</critical_hospitalization>
<exitus>0</exitus>
</row>
<row _id="row-aaw8-xgma-4nzd" _uuid="00000000-0000-0000-215D-97502438F37C" _position="0" _address="https://analisi.transparenciacatalunya.cat/resource/zzzz-zzzz/row-aaw8-xgma-4nzd">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_date>
```

OK 1 : 1

1 Result, 36 b

```
Hi ha hagut un increment de 46 casos
```

Total Time: 22.68 ms

**Compiling:**
- rewrite context value to document-node() item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node() item: util:root(db:open-pre("resources", 21682)) -> db:open-pre("resources", 21682 )
- rewrite context value to document-node() item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node() item: util:root(db:open-pre("resources", 21682)) -> db:open-pre("resources", 21682 )
- rewrite xs:integer item to positional access: 2 -> fn:position() = 2

**Optimized Query:**
concat("Hi ha hagut un increment de ", (db:

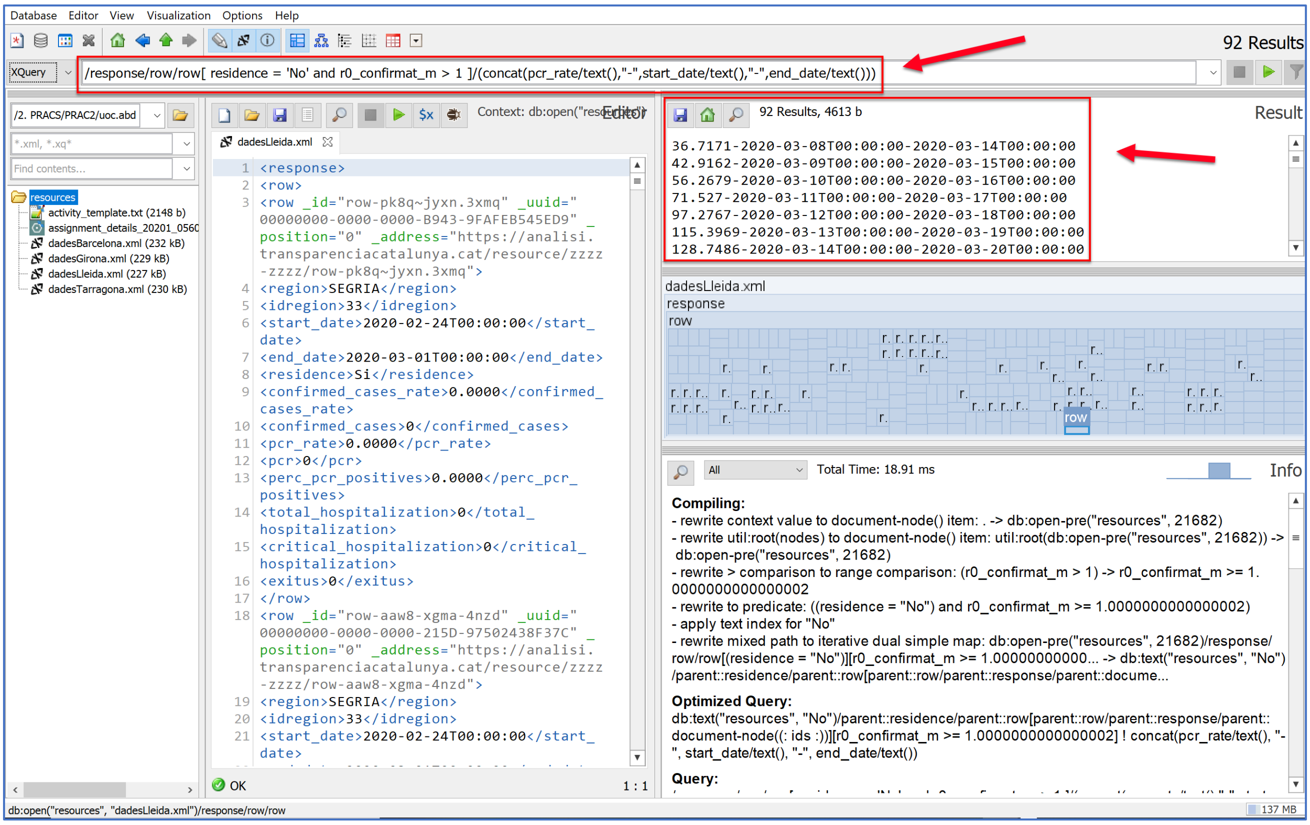
db:open("resources", "dadesLleida.xml")/response/row/row

## c)

In the same way we did in the previous activity, we can define an axe (**green**) that allows us to select all the elements contained in each inner “row” node, so we can operate with the whole set of selected elements. More precisely; this way we can define a “custom” node-test expression (**pink**) that retrieves an union of element values under the same node. To do so, we use the *concat()* xPath built-in string function along with every element axe + node-test selection we want to include in the union (underlined). The predicate (**light green**) restricts the results to this activity’s statement specifications:

```
/response/row/row[residence = 'No' and r0_confirmat_m > 1 ]
/(concat(pcr_rate/text(),"-",start_date/text(),"-",end_date/text()))
```

As we can see below, if we run the avobe’s xPath expression in our XML processor, we get the expected results:


Database Editor View Visualization Options Help
92 Results
XQuery

```
/response/row/row[ residence = 'No' and r0_confirmat_m > 1 ]/(concat(pcr_rate/text(),"-",start_date/text(),"-",end_date/text()))
```

/2. PRACS/PRAC2/uoc.abd
*.xml, *.xq*
Find contents...
resources
activity_template.txt (2148 b)
assignment_details_20201_0560
dadesBarcelona.xml (232 kB)
dadesGirona.xml (229 kB)
dadesLleida.xml (227 kB)
dadesTarragona.xml (230 kB)
Context: db:open("resources")
Editor
dadesLleida.xml

```
<response>
<row>
<row _id="row-pk8q~jyxn.3xmq" _uuid="
00000000-0000-0000-B943-9FAFEB545ED9" _
position="0" _address="https://analisi.
transparenciacatalunya.cat/resource/zzzz
-zzzz/row-pk8q~jyxn.3xmq">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_
date>
<end_date>2020-03-01T00:00:00</end_date>
<residence>Si</residence>
<confirmed_cases_rate>0.0000</confirmed_
cases_rate>
<confirmed_cases>0</confirmed_cases>
<pcr_rate>0.0000</pcr_rate>
<pcr>0</pcr>
<perc_pcr_positives>0.0000</perc_pcr_
positives>
<total_hospitalization>0</total_
hospitalization>
<critical_hospitalization>0</critical_
hospitalization>
<exitus>0</exitus>
</row>
<row _id="row-aaw8-xgma-4nzd" _uuid="
00000000-0000-0000-215D-97502438F37C" _
position="0" _address="https://analisi.
transparenciacatalunya.cat/resource/zzzz
-zzzz/row-aaw8-xgma-4nzd">
<region>SEGRIA</region>
<idregion>33</idregion>
<start_date>2020-02-24T00:00:00</start_
date>
```

```
36.7171-2020-03-08T00:00:00-2020-03-14T00:00:00
42.9162-2020-03-09T00:00:00-2020-03-15T00:00:00
56.2679-2020-03-10T00:00:00-2020-03-16T00:00:00
71.527-2020-03-11T00:00:00-2020-03-17T00:00:00
97.2767-2020-03-12T00:00:00-2020-03-18T00:00:00
115.3969-2020-03-13T00:00:00-2020-03-19T00:00:00
128.7486-2020-03-14T00:00:00-2020-03-20T00:00:00
```

Total Time: 18.91 ms

**Compiling:**
- rewrite context value to document-node() item: . -> db:open-pre("resources", 21682)
- rewrite util:root(nodes) to document-node() item: util:root(db:open-pre("resources", 21682)) -> db:open-pre("resources", 21682)
- rewrite > comparison to range comparison: (r0_confirmat_m > 1) -> r0_confirmat_m >= 1.0000000000000002
- rewrite to predicate: ((residence = "No") and r0_confirmat_m >= 1.0000000000000002)
- apply text index for "No"
- rewrite mixed path to iterative dual simple map: db:open-pre("resources", 21682)/response/row/row[(residence = "No")][r0_confirmat_m >= 1.00000000000... -> db:text("resources", "No")/parent::residence/parent::row[parent::row/parent::response/parent::docume...

**Optimized Query:**
db:text("resources", "No")/parent::residence/parent::row[parent::row/parent::response/parent::document-node((: ids :))][r0_confirmat_m >= 1.0000000000000002] ! concat(pcr_rate/text(), "-", start_date/text(), "-", end_date/text())

**Query:**

db:open("resources", "dadesLleida.xml")/response/row/row

# Activity 3

## a)

## b)

# Bibliography

1. **UOC Resources →** *Databases Architectures Module 2: Relational Extensions*
2. **XML Schema Example at w3.org →** *https://www.w3schools.com/xml/schema_example.asp*
3. **XML Attributes definition examples →** *https://www.w3schools.com/xml/schema_simple_attributes.asp*
4. **Oracle xPath Reference (Number functions) →** https://docs.oracle.com/cd/E35413_01/doc.722/e35419/dev_xpath_functions.htm#autoId41
5. **BaseX XML Processor →** http://files.basex.org/releases/9.4.3/BaseX943.zip
6. **xPath expressions →** https://stackoverflow.com/questions/9857756/how-to-get-the-preceding-element
7. **Oracle xPath Reference (Functions on strings) →** https://docs.oracle.com/cd/E35413_01/doc.722/e35419/dev_xpath_functions.htm#autoId14