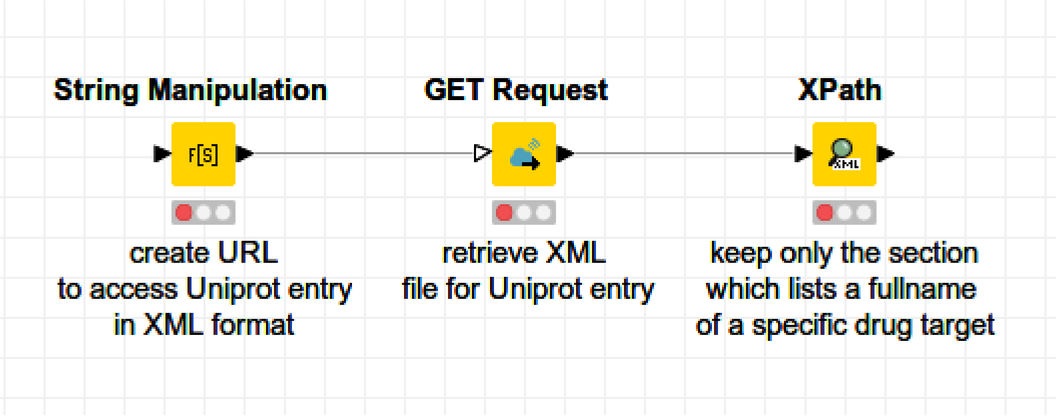
Full name:	
Immatriculation Number:	

Day 1: Programmatic Access to UniProt Database using KNIME

The Universal Protein Resource (UniProt) is a comprehensive resource for protein sequence and annotation data. The Uniprot ID (e.g, P59596, P59637, P0C6X7) is a protein identifier which can be used to retrieve information about a given protein, including protein names and synonyms, function, cellular localization, available 3D structures, as well as cross-references to other databases and many more. In the KNIME workflow provided today, we aim to access UNIPROT and other databases of interest programmatically, i.e. without the need to manually download and curate the data. Many databases enable to programmatically access their data via web services using APIs (Application Programming Interface). At the first stage, familiarize yourself with the API syntax for UNIPROT. Once you know how to define an API request to retrieve the information of interest, you can extract useful information from the XML UNIPROT entry. The 'String Manipulation', 'GET Request', and 'Xpath' nodes are all you need to create and execute API requests in your KNIME workfow:



1.1 Example Workflow

First, we specify the drug targets of interest on basis of their UNIPROT IDs. We use the ‘Table Creator’ node where we type UNIPROT IDs into separate cells. This table is the input for the KNIME workflow.

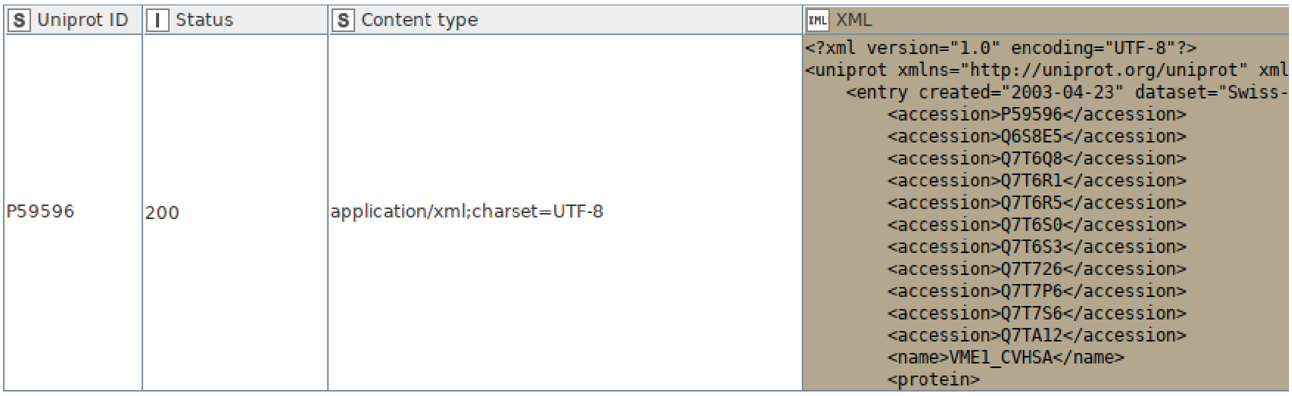
In the next instance, we use the ‘String Manipulation’ node to create an API request to download the XML file for each of the UNIPROT IDs. We use the join() function in the ‘String Manipulation’ node and insert the respective UNIPROT ID as a variable; the strip() function is used to make sure that there is no blank space in front of the UNIPROT ID:

```
join("http://www.uniprot.org/uniprot/",strip($Uniprot ID$),".xml")
```

As output of the ‘String Manipulation’ node, we can see an appended column with API requests included, such as:

http://www.uniprot.org/uniprot/P59596.xml

To get familiar with the XML file, copy one of the created API requests and paste it into a web browser. Have a look at the respective XML file to see what sort of information it provides. In the following step, we want to download the XML file into our KNIME workflow. We use the ‘GET Request’ node for this task. As output, we get an additional column with the received data, its content type, and the HTTP status code:



S Uniprot ID	I Status	S Content type	XML

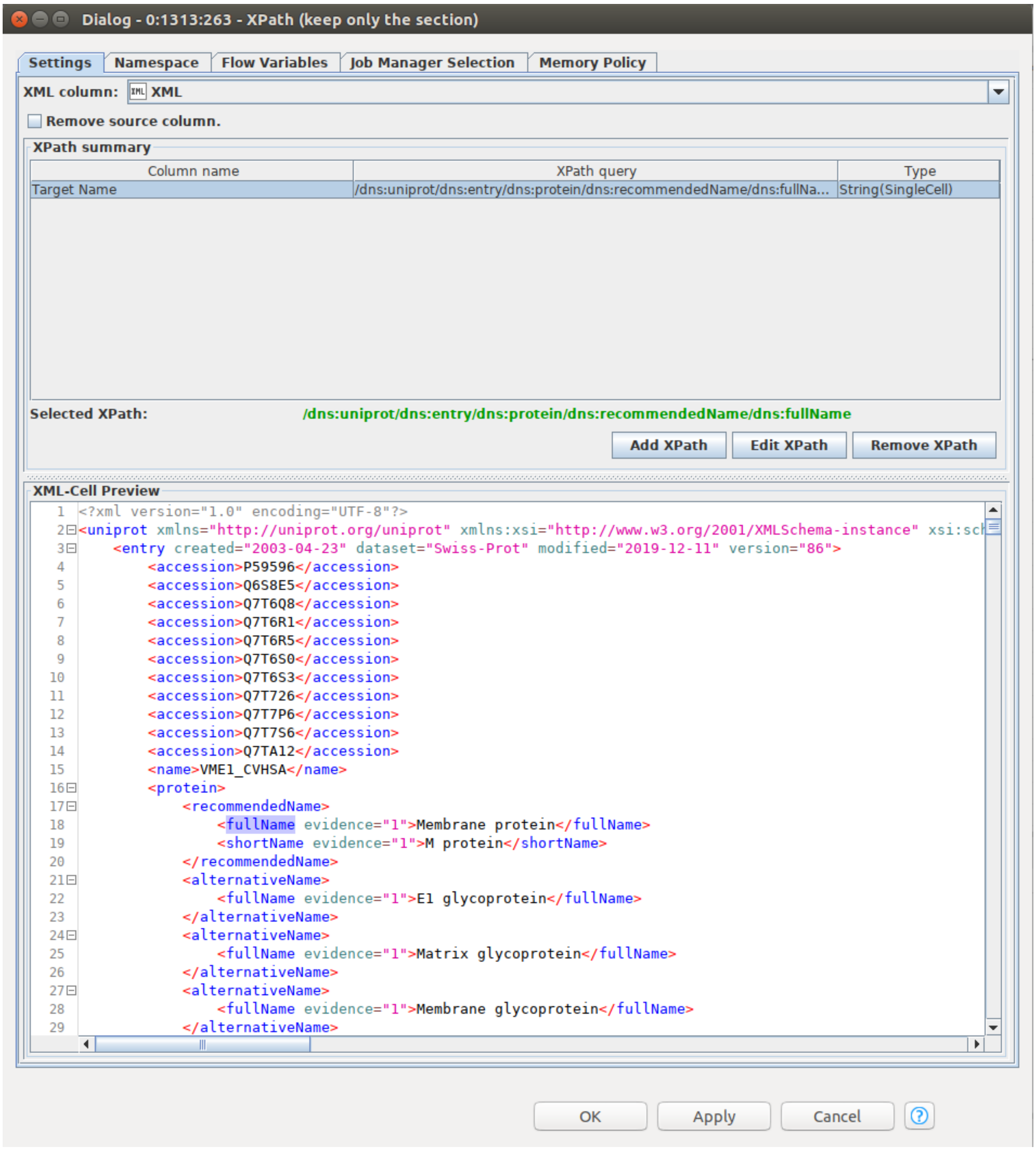
P59596	200	application/xml;charset=UTF-8	<?xml version="1.0" encoding="UTF-8"?>
			<uniprot xmlns="http://uniprot.org/uniprot" xml
			    <entry created="2003-04-23" dataset="Swiss-
			        <accession>P59596</accession>
			        <accession>Q6S8E5</accession>
			        <accession>Q7T6Q8</accession>
			        <accession>Q7T6R1</accession>
			        <accession>Q7T6R5</accession>
			        <accession>Q7T6S0</accession>
			        <accession>Q7T6S3</accession>
			        <accession>Q7T726</accession>
			        <accession>Q7T7P6</accession>
			        <accession>Q7T7S6</accession>
			        <accession>Q7TA12</accession>
			        <name>VME1_CVHSA</name>
			        <protein>

Status ‘200’ indicates that a standard response for the HTTP requests was successful. Familiarize yourself with other types of HTTP status codes.

Once we received the XML file per target in the KNIME table, we can use the ‘Xpath’ node to extract information of interest on basis of different XML elements. We can define a Xpath query by yourselves within the ‘Xpath’ node. The Xpath query below extracts the full name (XML element) of a respective target:

```
/dns:uniprot/dns:entry/dns:protein/dns:recommendedName/dns:fullName
```

A more straigthforward way is to perform a double-click on a specific section in the XML-Cell Preview table and Xpath is created automatically:



XML column: XML

□ Remove source column.

XPath summary

Column name	XPath query	Type
Target Name	/dns:uniprot/dns:entry/dns:protein/dns:recommendedName/dns:fullNa...	String(SingleCell)

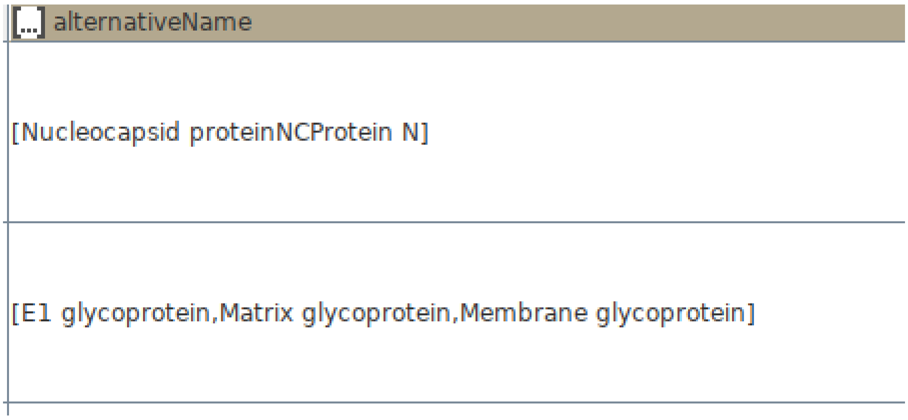
Selected XPath: /dns:uniprot/dns:entry/dns:protein/dns:recommendedName/dns:fullName

XML-Cell Preview

```
<?xml version="1.0" encoding="UTF-8"?>
<uniprot xmlns="http://uniprot.org/uniprot" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:sc
    <entry created="2003-04-23" dataset="Swiss-Prot" modified="2019-12-11" version="86">
        <accession>P59596</accession>
        <accession>Q6S8E5</accession>
        <accession>Q7T6Q8</accession>
        <accession>Q7T6R1</accession>
        <accession>Q7T6R5</accession>
        <accession>Q7T6S0</accession>
        <accession>Q7T6S3</accession>
        <accession>Q7T726</accession>
        <accession>Q7T7P6</accession>
        <accession>Q7T7S6</accession>
        <accession>Q7TA12</accession>
        <name>VME1_CVHSA</name>
        <protein>
            <recommendedName>
                <fullName evidence="1">Membrane protein</fullName>
                <shortName evidence="1">M protein</shortName>
            </recommendedName>
            <alternativeName>
                <fullName evidence="1">E1 glycoprotein</fullName>
            </alternativeName>
            <alternativeName>
                <fullName evidence="1">Matrix glycoprotein</fullName>
            </alternativeName>
            <alternativeName>
                <fullName evidence="1">Membrane glycoprotein</fullName>
            </alternativeName>
```

1.2 Questions & Challenges

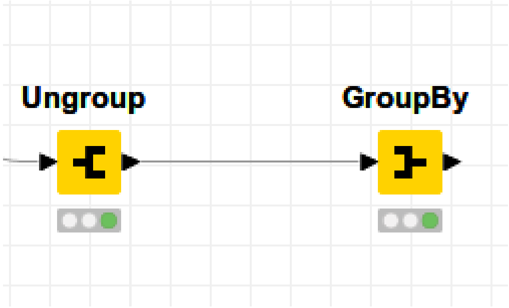
1. In the example workflow we learned how to retrieve the full name of a respective target defined via its UNIPROT ID. Adopt the same strategy to retrieve **alternative names** for all targets in a table. Please note, that a single UNIPROT entry might contain more alternative names for a respective target. Therefore, we want to list all available alternative names as a collection. We can do this in the ‘Xpath’ node configuration as follows: *(1)* Add Xpath query to get alternative name → *(2)* click on “Edit Xpath” → *(3)* in ‘Multiple Tag Options’ section select ‘Collection cell’ → *(4)* execute ‘Xpath’ node and check output Table. You should see that the ‘alternative name’ column contains data with ‘Collection’ data type, as indicated by the **[...]** icon:



[...] alternativeName
[Nucleocapsid proteinNCProtein N]
[E1 glycoprotein,Matrix glycoprotein,Membrane glycoprotein]

There are several alternative ways to process *collection* data types; we can either use the ‘Split Collection Column’ or the ‘Ungroup’ node to separate alternative names into single instances. Your task is to probe both nodes and figure out what is the difference and how the output table looks like. After getting familiar with the nodes, please fill in the box below:

2. Your next task is to calculate the number of alternative names per a given target. To achieve this task, the best option is to use the ‘Ungroup’ node followed by the ‘GroupBy’ node:



The ‘GroupBy’ node groups the rows of a table by the unique values in the selected group columns. In our case, the UNIPROT IDs represent the unique values we want to use for grouping. A row is created for each unique set of UNIPROT IDs. The remaining columns are aggregated based on the specified aggregation settings. We want to **count** all alternative names per given UNIPROT ID. You can perform this task by choosing the ‘Aggregation’ functionality in the section ‘Manual Aggregation’ in the ‘GroupBy’ node configuration.

After execution of your KNIME workflow please fill in number of alternative names per UNIPROT ID into the table below:

Uniprot ID	**Number of Alternative Names**
P0C6X7	
P0C6U8	
P59594	
Q9BYF1	
O15393	
P50052	
A0A220F1P8	

3. Next, we want to extract the protein sequence and its length. Extract this information using the 'Xpath' node, as previously. We will now work not only with XML elements, but also with XML attributes. Attributes are designed to contain data related to a specific element. In the example below 'gender' is an attribute of 'person' XML element::

```
<person gender='female'>
```

For retrieving the sequence length we will need to use a specific attribute of the <sequence> element. For further information look into the KNIME 'Help' section (*hint: you might use the '@' character to specify an attribute in your Xpath query*). Complete Xpath query for sequence length retrieval into the box:

```
/dns:uniprot/dns:entry/dns:[          ]/[     ]
```

3. Filter the output table from the previous step to only keep those targets whose sequence is larger than 250 amino acids. You can use the 'RowFilter' node for this. Read the 'Help' section to figure out how to filter a KNIME table on basis of a number range. However, keep in mind that the sequence length retrieved via 'Xpath' node is included as 'string' data type in your table. Figure out how to convert string to number in KNIME and write down possible solutions into the box:

After filtering the table for the sequence length > 250 amino acids, create a screenshot of your final table which includes the 'Uniprot ID' and 'Length' columns only *(hint: all redundant columns can be removed from the table by using the 'Column Filter' node).* Upload a screenshot as a separate file, alongside with this protocol.