Full name:	
Immatriculation Number:	

Day 2: Using Cross-References to Retrieve Structural Data from the Protein Data Bank (PDB)

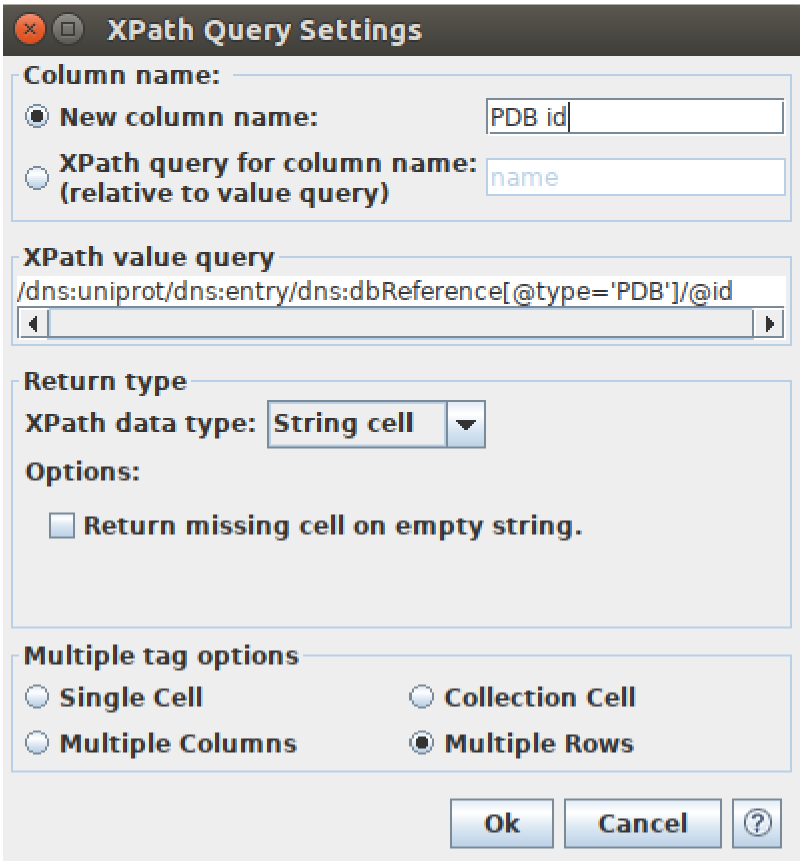
As we learned yesterday, KNIME can be handy to query databases in an automated fashion via API requests. UNIPROT contains a multitude of useful information about a given protein. Today we will learn how to interconnect different databases by using cross-references. More specifically, we will use UNIPROT IDs to retrieve and analyze available 3D protein structures from the PDB.

2.1 Example Workflow

UNIPROT provides cross-references to the PDB in the form of PDB IDs for a given UNIPROT entry (e.g, 3o7q, 4zw9, 3wdo). PDB cross-references are integrated in XML files as `<dbReference>` elements. However, there are multiple `<dbReference>` elements which provide cross-references to **different** databases, such as:

```
<dbReference type="PubMed" id="12730500"/>
<dbReference type="GO" id="GO:0039579">
<dbReference type="InterPro" id="IPR036333">
<dbReference type="Pfam" id="PF06478">
<dbReference type="PDB" id="6NUR">
```

The question arises how to enforce KNIME to only keep XML elements which do possess the ‘PDB’ attribute. We can to this by a proper definition of the Xpath query. We will use a similar syntax like in the previous exercise where we extracted the length of the protein sequence. However, we have to generalize the query in a way that it solely retrieves all ‘id’ values which correspond to the ‘type=”PDB”’ attribute. A correct definition to achieve this operation is shown in the following screenshot:



XPath Query Settings

Column name:

● New column name: PDB id

○ XPath query for column name: (relative to value query) name

XPath value query

/dns:uniprot/dns:entry/dns:dbReference[@type='PDB']/@id

Return type

XPath data type: String cell

Options:

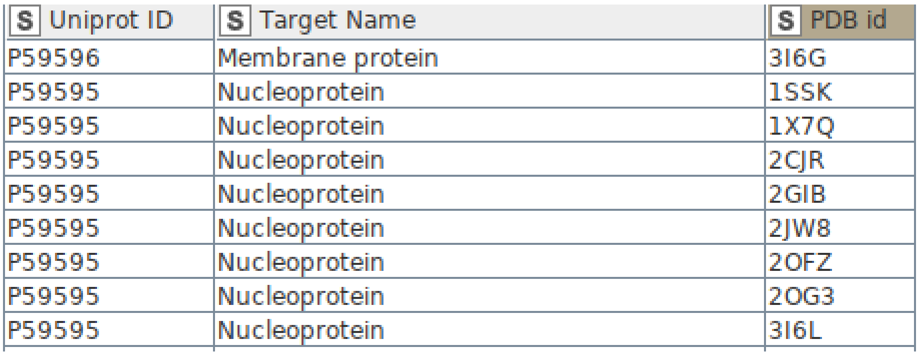
□ Return missing cell on empty string.

Multiple tag options

○ Single Cell ○ Collection Cell

○ Multiple Columns ● Multiple Rows

Please note that there can be multiple PDB IDs per single UNIPROT entry available. Therefore, we select the option ‘Multiple Rows’ in ‘Multiple tag options’ to list all unique PDB IDs into a separate row in table. The output table will look as follows:



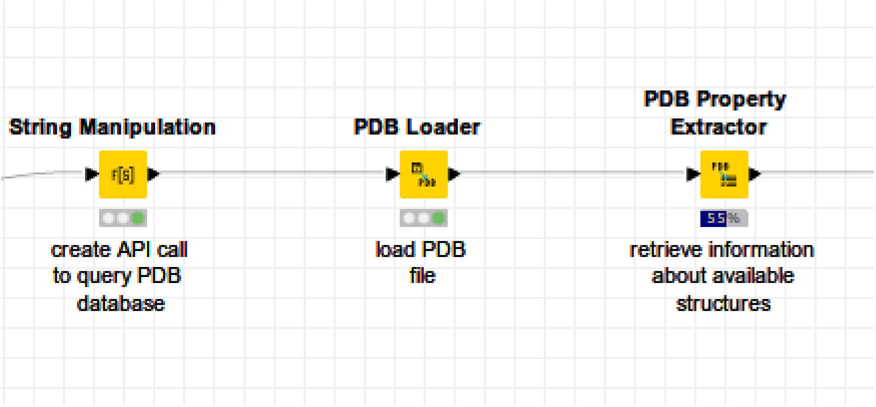
S Uniprot ID	S Target Name	S PDB id
P59596	Membrane protein	3I6G
P59595	Nucleoprotein	1SSK
P59595	Nucleoprotein	1X7Q
P59595	Nucleoprotein	2CJR
P59595	Nucleoprotein	2GIB
P59595	Nucleoprotein	2JW8
P59595	Nucleoprotein	2OFZ
P59595	Nucleoprotein	2OG3
P59595	Nucleoprotein	3I6L

In the given example, there is only a single PDB ID (3I6G) for ‘Membrane protein’, while there are eight PDB IDs for ‘Nucleoprotein’, and so on.

In the following step, we use the ‘PDB id’ column to create an API request to extract useful information about the protein structures from PDB. We use the ‘String Manipulation’ node to create the API calls as follows:

```
https://files.rcsb.org/view/3I6G.pdb
```

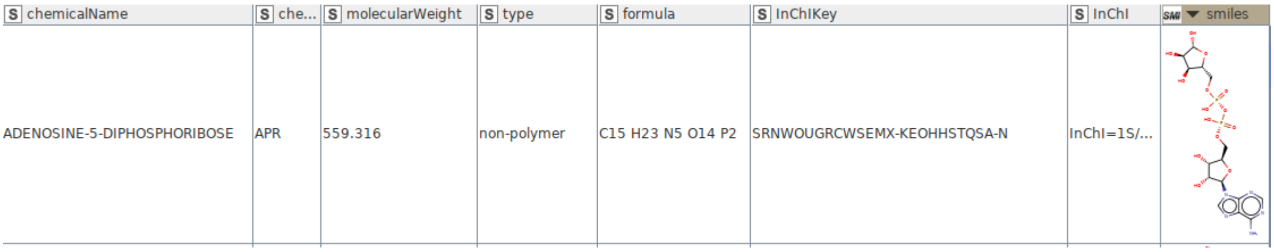
Since specialized PDB nodes are an integral part of KNIME, we can use them to load PDB files via API calls (‘PDB Loader’ node) and extract different structural properties (‘PDB Property Extractor’ node) directly:


String Manipulation
create API call to query PDB database
PDB Loader
load PDB file
PDB Property Extractor
55%
retrieve information about available structures

The PDB webservices provide many diverse utilities. For example, we can specify an API call to request detailed information about co-resolved ligands. An example of such an API call is provided below:

```
https://www.rcsb.org/pdb/rest/ligandInfo?structureId=3I6G
```

After fetching co-resolved ligands with KNIME, our aim will be to examine ligand properties in greater detail:



S chemicalName	S che...	S molecularWeight	S type	S formula	S InChIKey	S InChI	SMI smiles
ADENOSINE-5-DIPHOSPHORIBOSE	APR	559.316	non-polymer	C15 H23 N5 O14 P2	SRNWOUGRCWSEMX-KEOHHSTQSA-N	InChI=1S/...	

2.2 Questions & Challenges

1. In the example workflow we learned how to link UNIPROT entries to the available PDB structures. After retrieval of available PDB structures use appropriate nodes to calculate the number of unique PDB IDs per UNIPROT entry and fill in the table below:

Uniprot ID	Number of Unique PDB IDs
P0C6X7	
P0C6U8	
P59594	
Q9BYF1	
O15393	
P50052	
A0A220F1P8	

2. After execution of the PDB nodes in your workflow, you should get a list of PDB IDs with their structural properties. Your task is to create a list of available experimental methods (‘Experimental Method’ column) and calculate the number of unique PDB IDs which were resolved by the respective methods.

Experimental Method	Number of Unique PDB IDs

3. Which KNIME nodes would you use to calculate basic statistics, such as mininum, maximimum, mean, and median values? Name at least two different nodes:

4. In case of the PDB structures resolved via X-Ray diffraction, use any available KNIME node to calculate the mininum, maximimum, mean, and median values of crystal resolution [Å].

Statistic Value Type	Crystal resolution [Å]
Minimum	
Maximum	
Mean	
Median	

5. After retrieval of co-resolved ligands, look at different ligand properties. What is the minimum, maximum, and median molecular weight for those ligands?

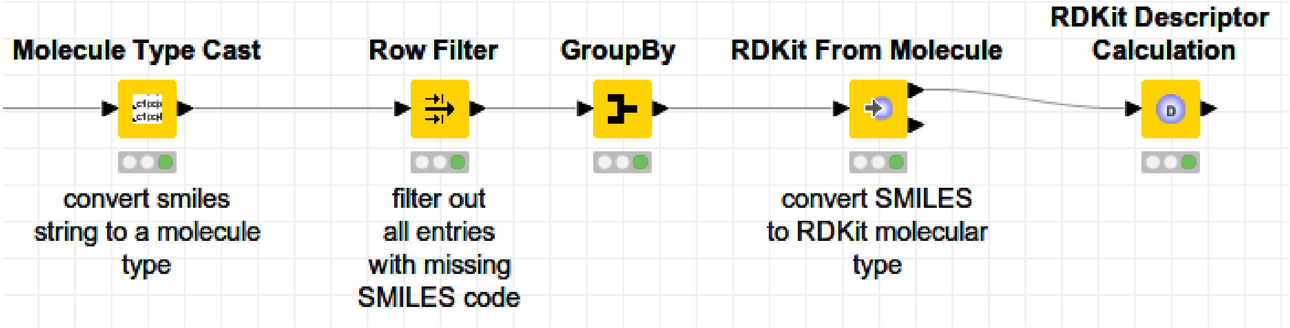
Statistic Value Type	Molecular Weight
Minimum	
Maximum	
Mean	
Median	

6. Calculate the number of unique ligands per protein target and fill in the table.

Uniprot ID	Number of Unique ligands
P0C6X7	
P0C6U8	
P59594	
Q9BYF1	
O15393	
P50052	
A0A220F1P8	

7. Provide additional ligand properties by calculating RDKit descriptors for the ligands. First, convert the SMILES string format into a molecular type (‘Molecule Type Cast’ node) format. Now you can visually inspect the ligand structure in a table. As a next step, remove all rows with missing molecular type (‘Row Filter’ node) and keep only unique ligands (‘GroupBy’ node using ligand molecular type as a group). Afterwards, convert smiles columns with visualized structures into the RDKit format (‘RDKit from Molecule’). Then,

calculate RDKit descriptors by using the ‘RDKit descriptor calculation’ node. Below you can see the visual depiction of such a workflow:


Molecule Type Cast
convert smiles string to a molecule type
Row Filter
filter out all entries with missing SMILES code
GroupBy
RDKit From Molecule
convert SMILES to RDKit molecular type
RDKit Descriptor Calculation

Finally, use any KNIME node of your choice to calculate different statistical measures for the respective RDKit descriptors (table below):

	Rdkit descriptor					
Statistic Value Type	SlogP	SMR	TPSA	Number of Heteroatoms	Number of Rotatable Bonds	Number of Aromatic Rings
Minimum						
Maximum						
Mean						
Median						