**Quick Start Guide for   
AllenGUI & AllenSPARQL**

# 1 Introduction

This document describes how to run temporal queries with AllenGUI and AllenSPARQL based on the minimal test dataset with 13 patients. This dataset has been constructed to test the correctness of the implemented software.

This document guides you through “Method 3: No D2RQ, i2b2 and Oracle database” as described in **Test Dataset\README.txt**. To make running the tools easier, we’ve automated the steps with batch files. The overall workflow is a bit awkward (e.g., you have to manually restore the Fuseki triple store by uploading the RDF file), which is owed to the fact that there is no i2b2 Oracle database where the data can be pulled from. If you do have an Oracle database or even an i2b2 installation, please refer to the README file on how to use this. But for a small demo following this guide is easier.

This document assumes that you are using a recent version of Windows, therefore the helper files used in this script are Windows batch files (.bat). However, everything also works on any Linux or Mac computer, as long as you adopt and run these commands manually.

# 2 Prerequisites

To use AllenGUI and AllenSPARQL, you need to install the following software on your computer:

* Java SE Development Kit 8: <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
* Apache Maven (for building the source code): <https://maven.apache.org/download.cgi>
* also see: <https://maven.apache.org/install.html>
* Python 3 (for running **sagecell-client.py**): <https://www.python.org/downloads/windows/>  
  When installing Python, make sure the set PATH option is ticked!

# 3 Initialization

This section describes how to compile the software and how to upload data into the Fuseki triple store. To initialize the software stack, please execute the following batch files one after another:

**01-InstallPrerequisites.bat**

This installs two Python packages required by **sagecell-client.py**.

**02-InstallD2RQLibrary.bat**

This installs the D2RQ library into your local Maven repository.

**03-BuildAllenSPARQL.bat**

This compiles AllenSPARQL using Maven.

**04-DownloadFuseki.bat**

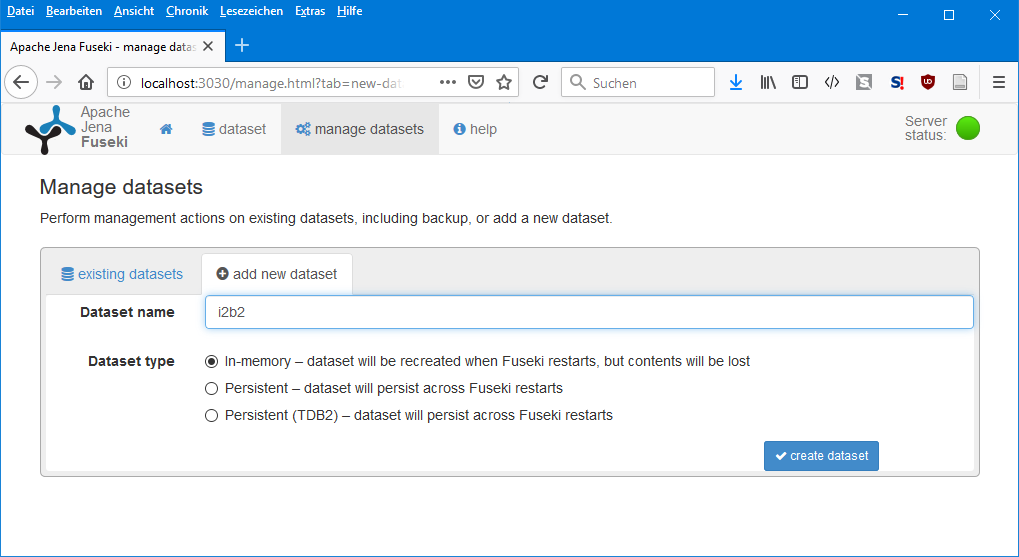
This downloads and extracts Apache Jena Fuseki, the RDF store. If it fails, you may need to modify the download URL in the file.

**05-StartFuseki.bat**

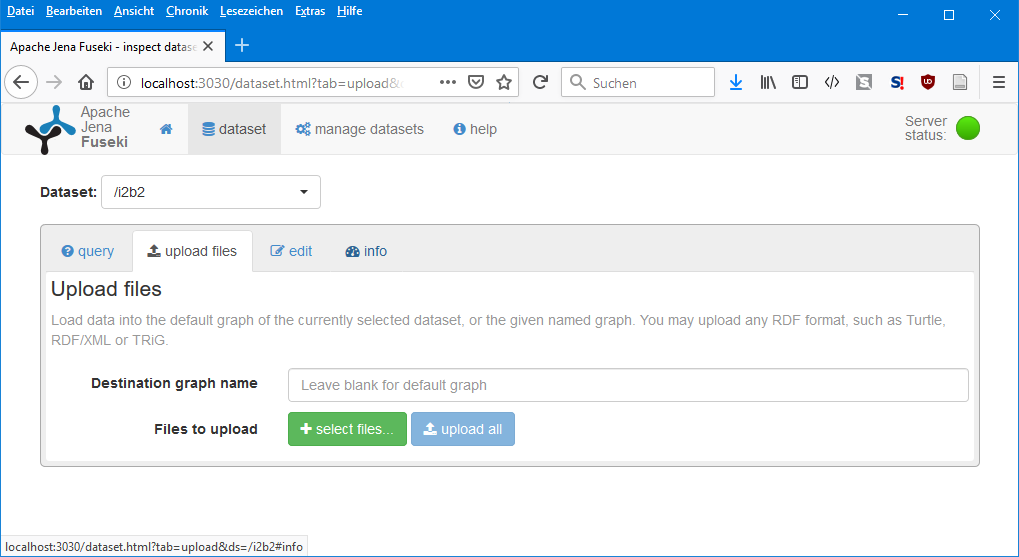
This starts the Fuseki server.

The next step is to upload the test data set into Fuseki. To do this, go to <http://localhost:3030/> with your browser. If you can’t access this page, Fuseki is not running.

Click on “manage datasets”, “add new dataset” and enter “i2b2” as new dataset name. Select “In-memory”.

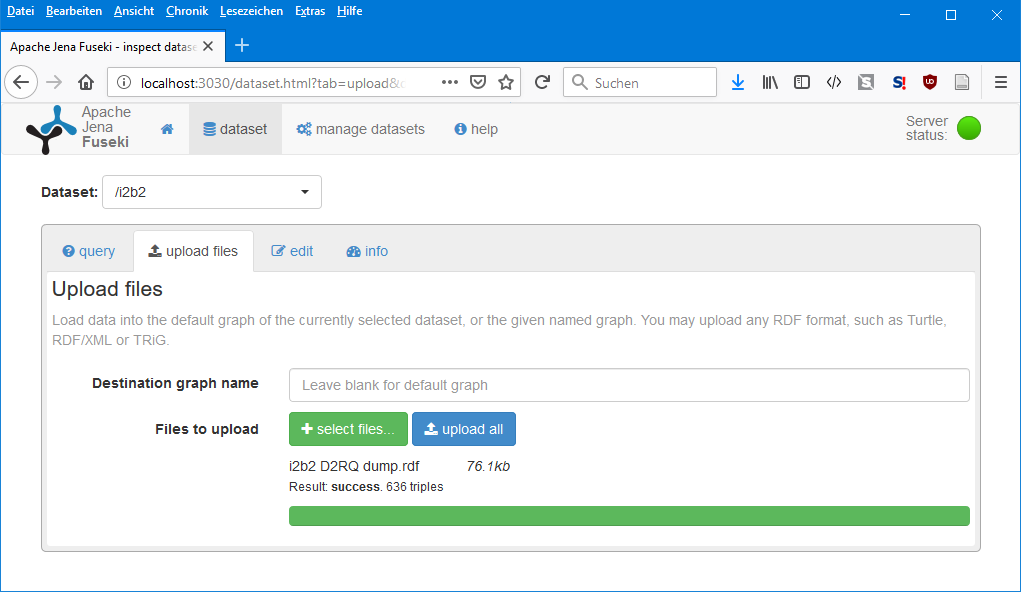


Click on “create dataset”. On the next screen, click on “upload data” right to the “/i2b2” entry:



Click on “select files” and select **Test Dataset\i2b2 D2RQ dump.rdf**.

Click on “upload now”.



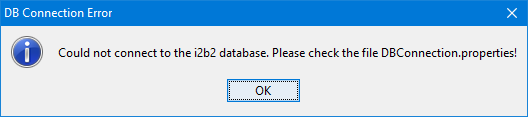
The data in Fuseki is now ready for use.

Finally, run **06-StartTemporalTools.bat**

This starts AllenSPARQL and AllenGUI.

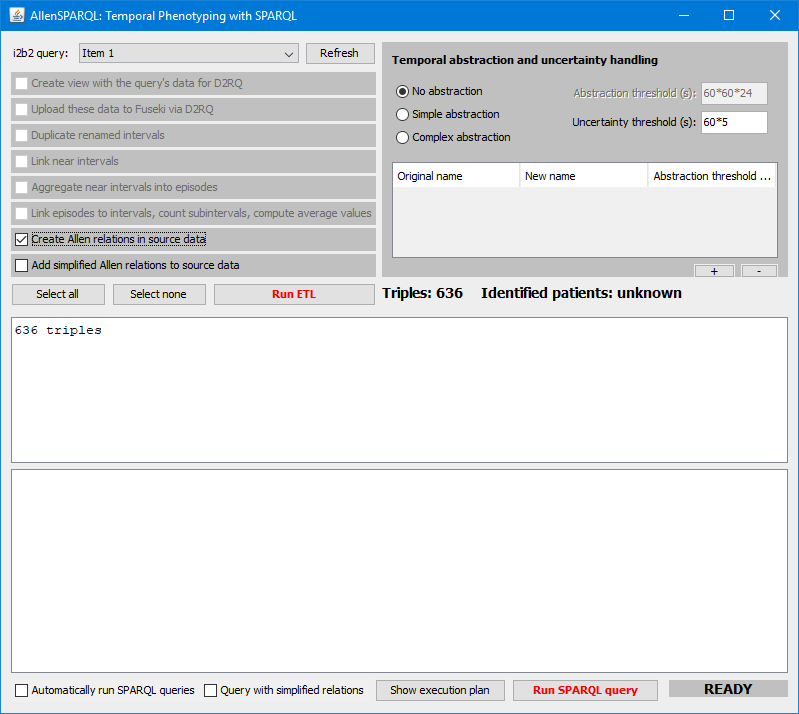
# 4 Querying without Temporal Abstraction

After starting AllenSPARQL with the default configuration, it will show a database connection error:



This is the expected and correct behavior, as this quick start guide does not use an i2b2 system. Instead we will directly querying the Fuseki server. Ignore all errors ☺ and click on OK.

The main Window should look like this:

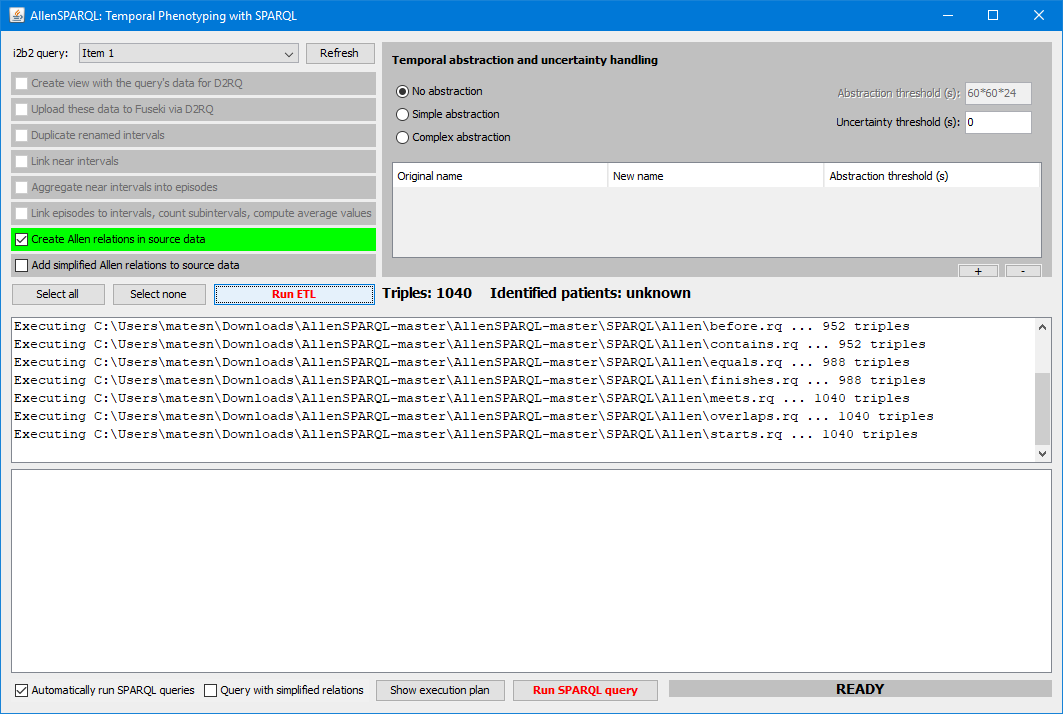


Notice that it displays 636 triples. If nothing is shown, the Fuseki server may not be running.

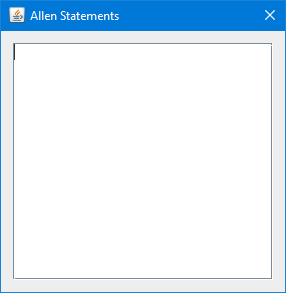
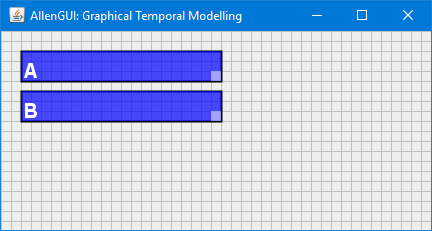
Tick “Automatically run SPARQL queries” in the lower left corner of the window.

Set “Uncertainty threshold (s)” to “0”.

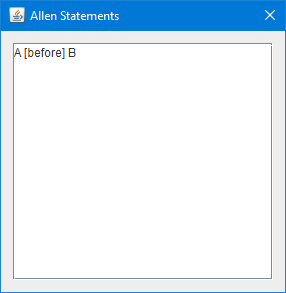
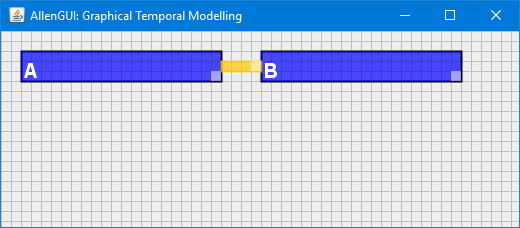
Tick “Create Allen relations in source data” and then press the button “Run ETL”. This creates the Allen relations among the data in the triple store. The ticked ETL step should turn green and the program should then report 1040 triples:



Now to AllenGUI. The tool should display the following two windows:

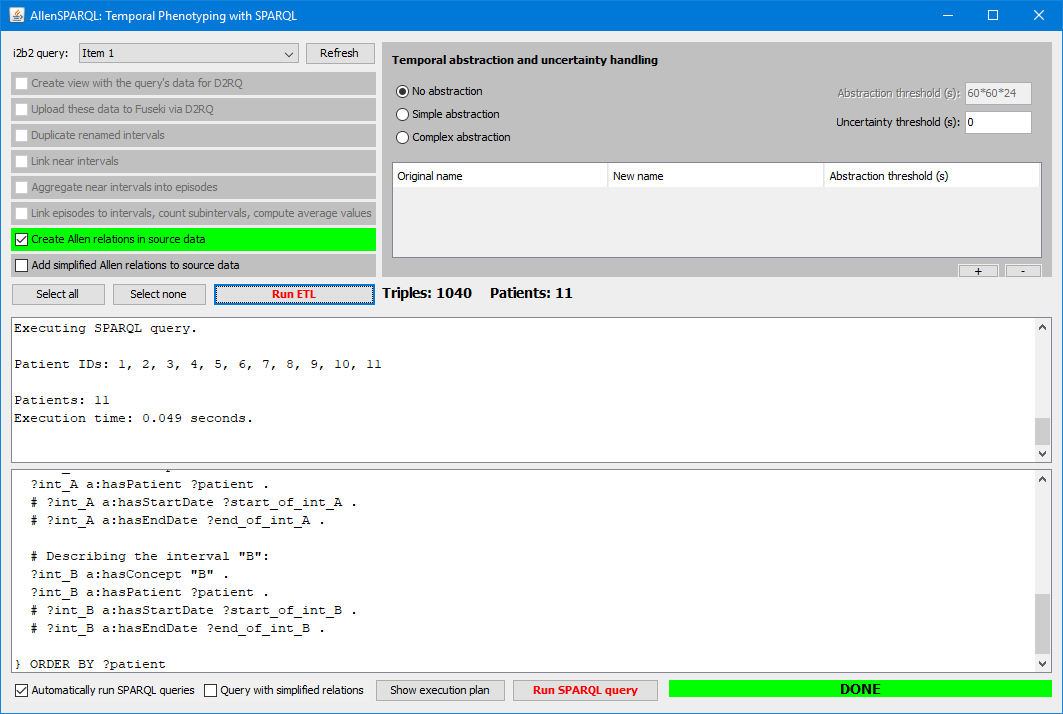
Use the main window with the grid to model temporal patterns, such as “A before B”:

This can be done with the mouse:

* Use the left mouse button to move items on the canvas.
* To create a new block (or a yellow connector line), click on an empty space on the grid.
* To delete an item, click on it with the right mouse button.
* To rename an item, press the middle mouse button.

AllenSPARQL should have already executed this query. It should report 11 patients (IDs 1 - 11):

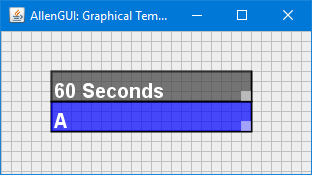


The following tables show the default content of the database. There are thirteen fictional patients. On three consecutive days (Day 1 = 2017-01-01, Day2 = 2017-01-02, Day 3 = 2017-01-03), there may be two different types of intervals, “A” and “B”. Whenever there are intervals of any type, these take place at the start of the day (00:00:00 to 00:01:00) and at the end of the day (23:59:00 to 00:00:00). Each interval has a length of 1 Minute.

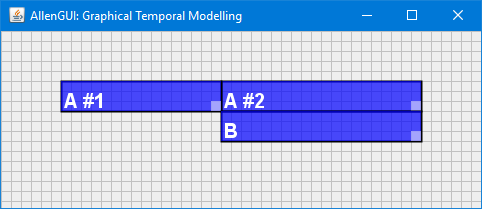


The “A before B” query from above returned all patients except 12 and 13, because the latter do not have any A before B (notice that for patients 12 and 13, all “A”s are after all “B”s).

Because each patient has at least one “A” interval that has a duration of one minute, the following query will also return all 13 patients:



This query will return patients 3, 4 and 5:



The reason for this is obvious when reviewing the data:



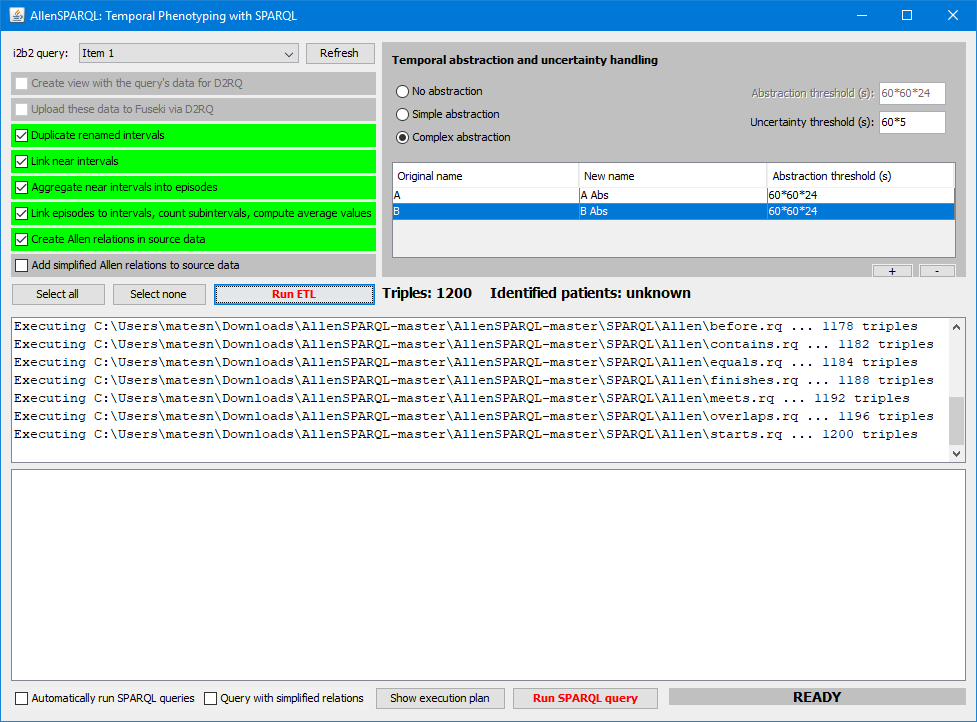
# 5 Querying with Temporal Abstraction

* Close all programs, including Fuseki. This will wipe the Fuseki “i2b2” dataset, as it was configured as “in-memory”.
* Run **05-StartFuseki.bat**
* Open the web interface of Fuseki and upload **Test Dataset\i2b2 D2RQ dump.rdf** again.
* Run **06-StartTemporalTools.bat**

Then, in AllenGUI:

* Tick “Automatically run SPARQL queries” in the lower left corner of the window.
* Make sure that “Duplicate renamed intervals” is ticked
* Select “Complex Abstraction” on the right. With the small “+” Button, create two new entries that abstract all nearby “A” intervals into “A Abs” and all “B” intervals into “B Abs”.
* Click on “Run ETL”.

After the execution, the program should report 1200 triples:

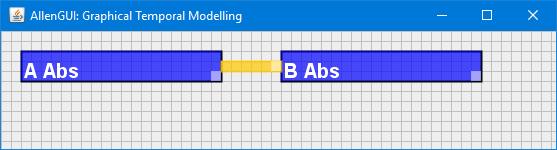


This abstraction does the following: Whenever there is a series of intervals of one type, such as “A”, and whenever the temporal distance between any two of these intervals is less than the “Abstraction threshold”, a new interval of the type “A Abs” is created that encompasses the original series.

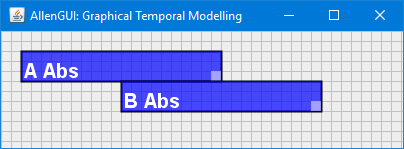
Using a threshold of 60\*60\*24 seconds = 24 hours as in the example above, this generates the “A Abs” and “B Abs” intervals as shown on the right:



Now querying with AllenGUI (while using the abstracted intervals), the following query only returns patient 1:



This query returns patient 3:



Feel free to test all other Allen relations. ☺

Similarly, the following query will return patient 5, because this is the only patient that has a three-day interval of type “A Abs”:

