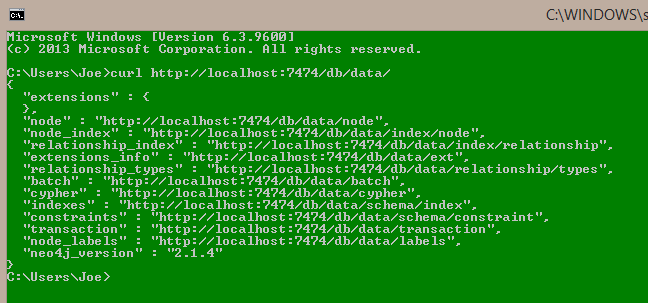
**Portfolio Part 4: Neo4J**

**T00155775**

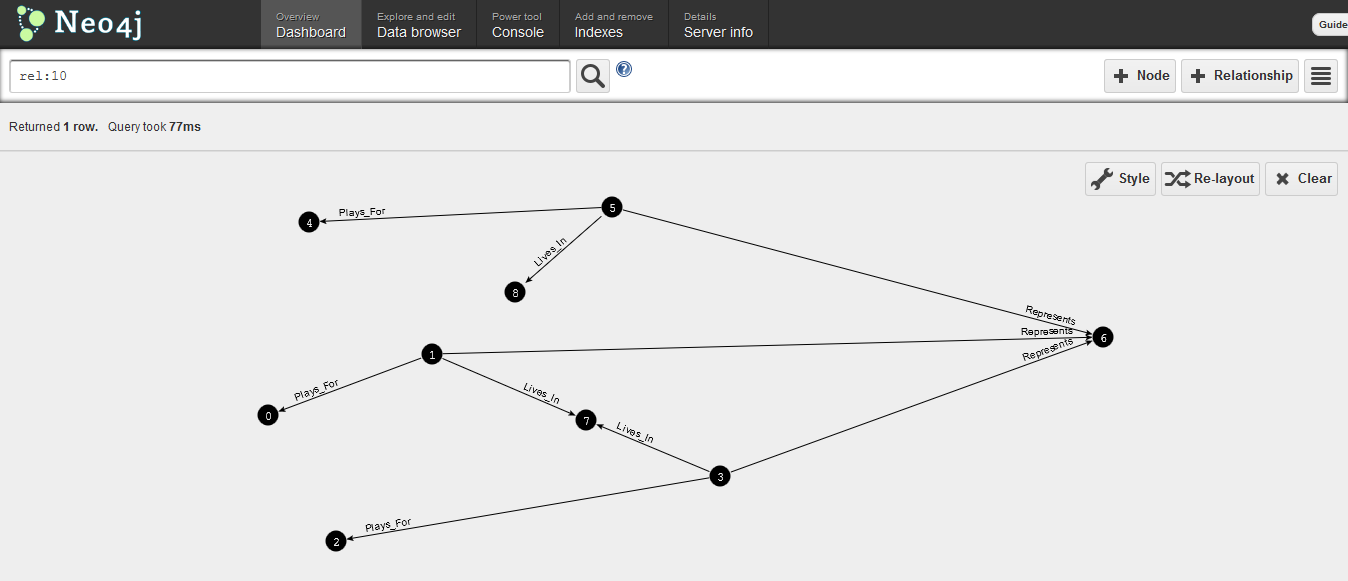
**Joe O Flaherty**

**BSc (Hons) Computing with Multimedia**

## Build your own graph in Neo4J with at least 6 nodes with edges between them.



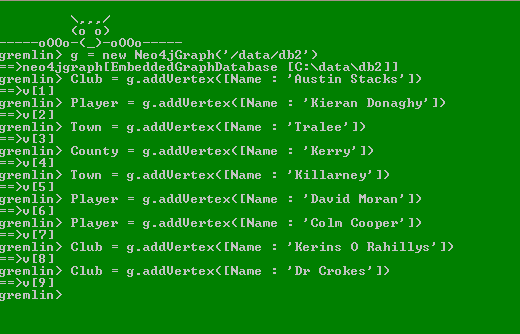
Neo4J installed and running.



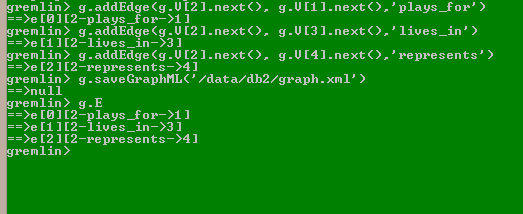
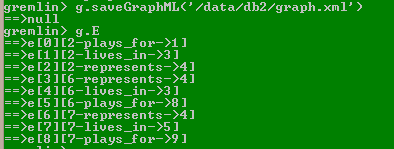
Nodes and relationships created based on football clubs, players, county and town lived in. This graph has been created using the web interface and was which provides a very user friendly and intuitive environment.

## Show examples of Gremlin queries using pipelines to access data.

Initially this involved recreating the previous database using Gremlin but storing it in a different folder as it is not possible to have the same database open twice. The most important process when using Gremlin is to actually save the database as an XML file to persist it, failure to perform this step results in loss of previous work and data.



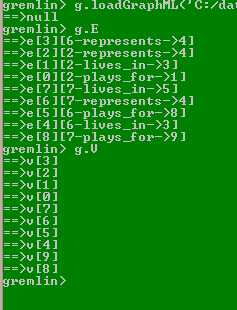
Database created with vertices added.

Added edges (relationships) to the graph and saved as an XML file. The Gremlin command g.E is used to confirm that the edges have been created successfully. Gremlin refers to vertices and edges for what Neo4J terms nodes and relationships.



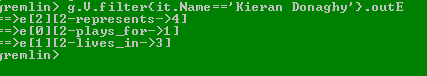
Loading the graph from the previously saved XML file.



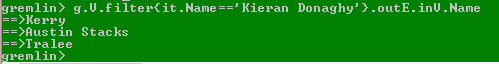
Using the gremlin commands g.E and g.V to confirm that the graph has re-loaded correctly, g.E displays all the edges on the graph, g.V all the vertices.



Isolating a particular vertex and confirming the correct vertex is chosen using the .map( ) method.



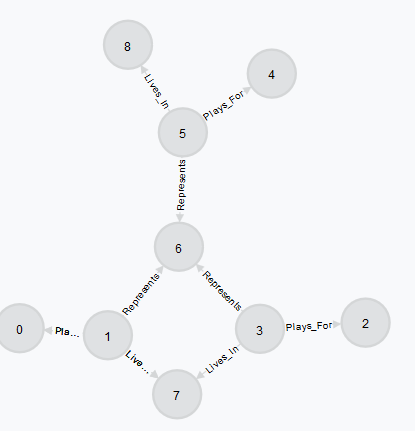
Using g.V.filter in conjunction with .outE method to return all the edges emanating out from a specified vertex.



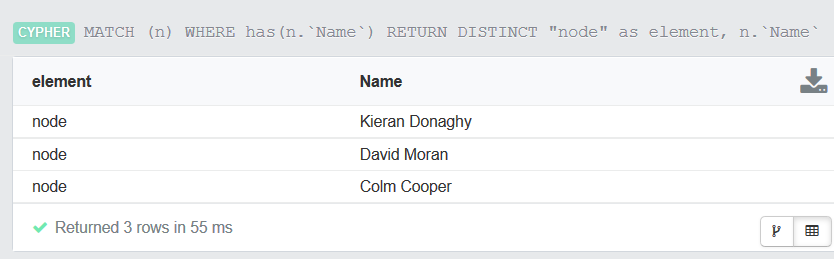
Expanding on the previous command by including the .inV method with the parameter .Name to identify by name all the edges coming into a specified vertex.

## Investigate Cypher and show how to run the same queries against your graph using Cypher

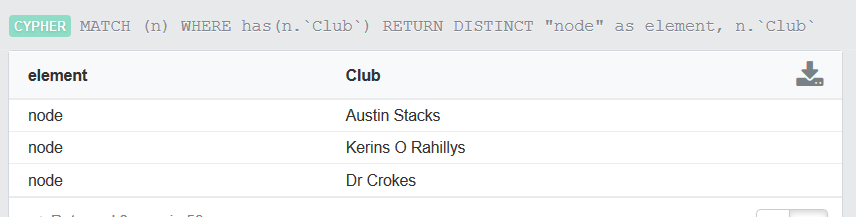
It proved much easier to change the queries slightly and run some more complex queries using Cypher



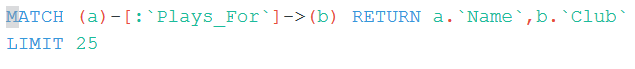
The Cypher view of the graph from the Neo4J GUI interface.



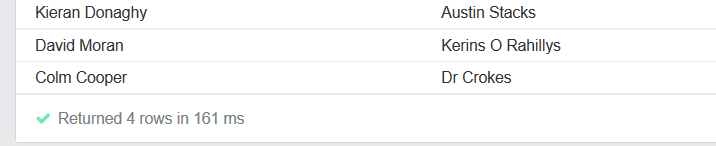
All the nodes that have a ‘Name’ property using the Cypher MATCH and WHERE commands – very similar to SQL commands.

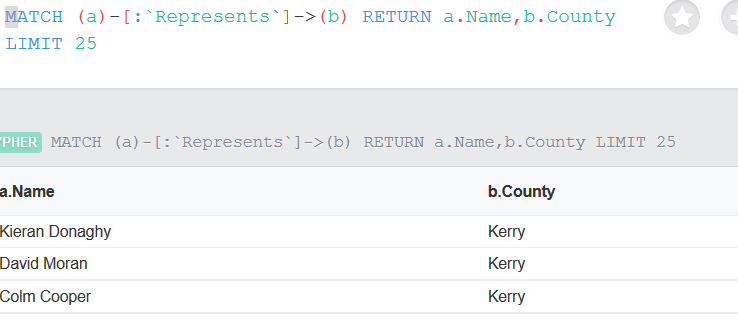


The same query but returning ‘Club’ nodes.

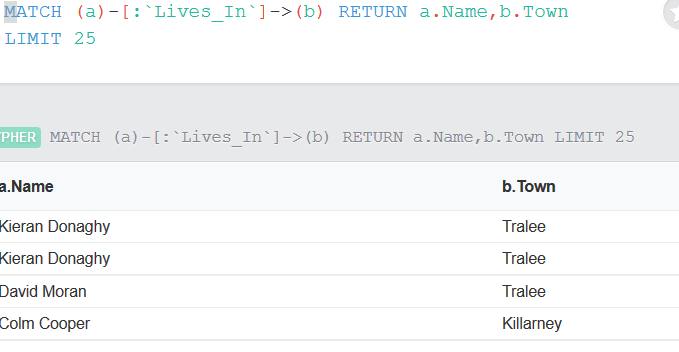


Query to return the name of a player and what club they play for the result is shown below.





What County each player represents

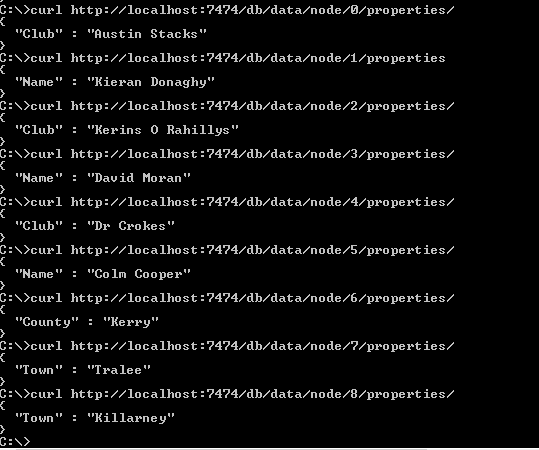


Where each player lives

## Use the Neo4J REST interface from the console to run the same queries.

Complex queries proved impossible with the REST interface as there is no means of retaining is the current node for the purposes of graph walking.

Some simple REST queries were executed as examples:

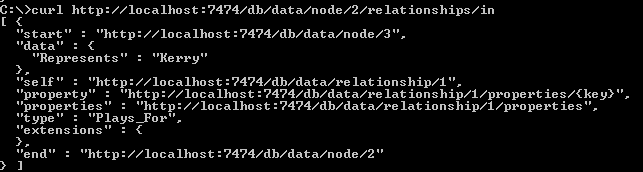


Using the REST interface to find the properties of each node on the graph.

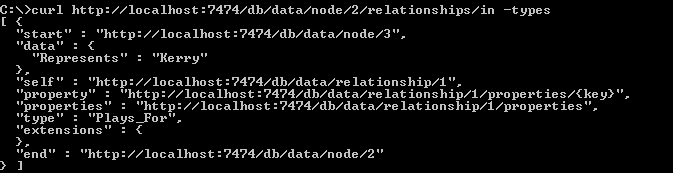


All relationships out of a specified node (1) using

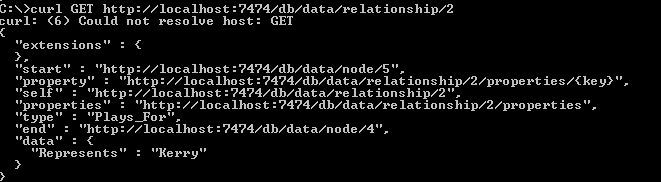




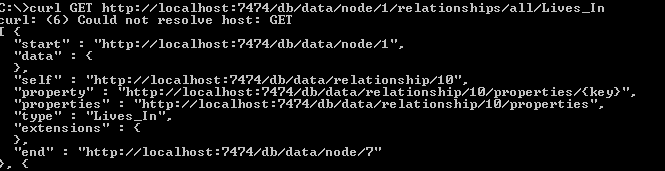
Relationships into a specified node.



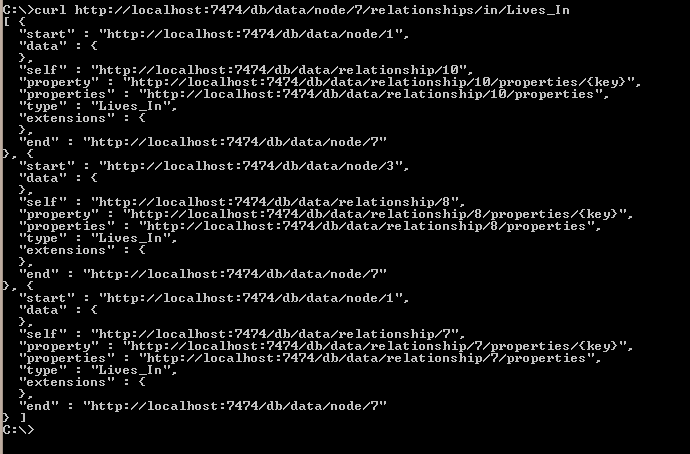
The ‘types’ of relationship a specified node has.



What nodes have relationships to the node specified in the curl command plus what type the relationship is.

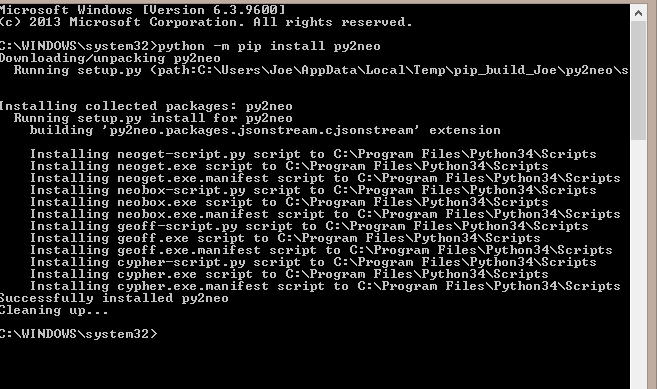


Using a type “Lives\_In” to find where a relationship exists – node 1 Lives\_In node 7 in this example.

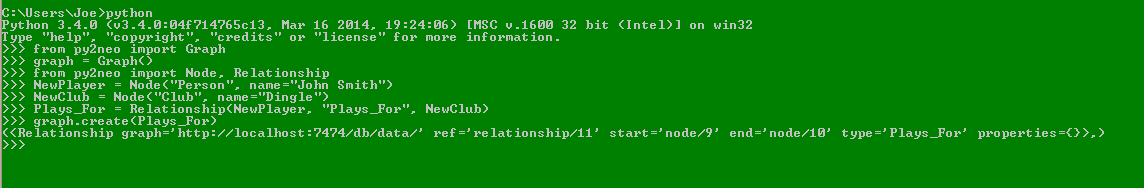


All the nodes that have a relationship ‘Lives\_In’ with node 7.

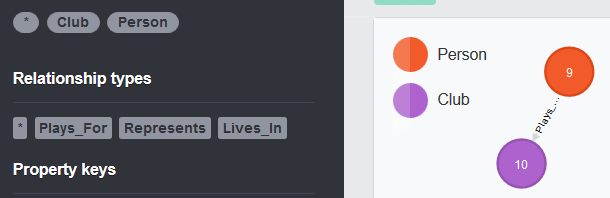
## Download and install a library for a programming language and show how to make a node and an edge from the programming language and show an example of querying the graph from the programming language.



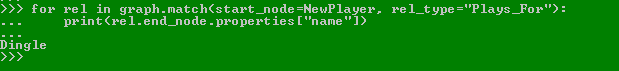
Importing py2neo a Python module using pip – had to run the CLI as admin to complete the installation.



Created 2 nodes and a relationship between them from Python using methods of the py2neo module.



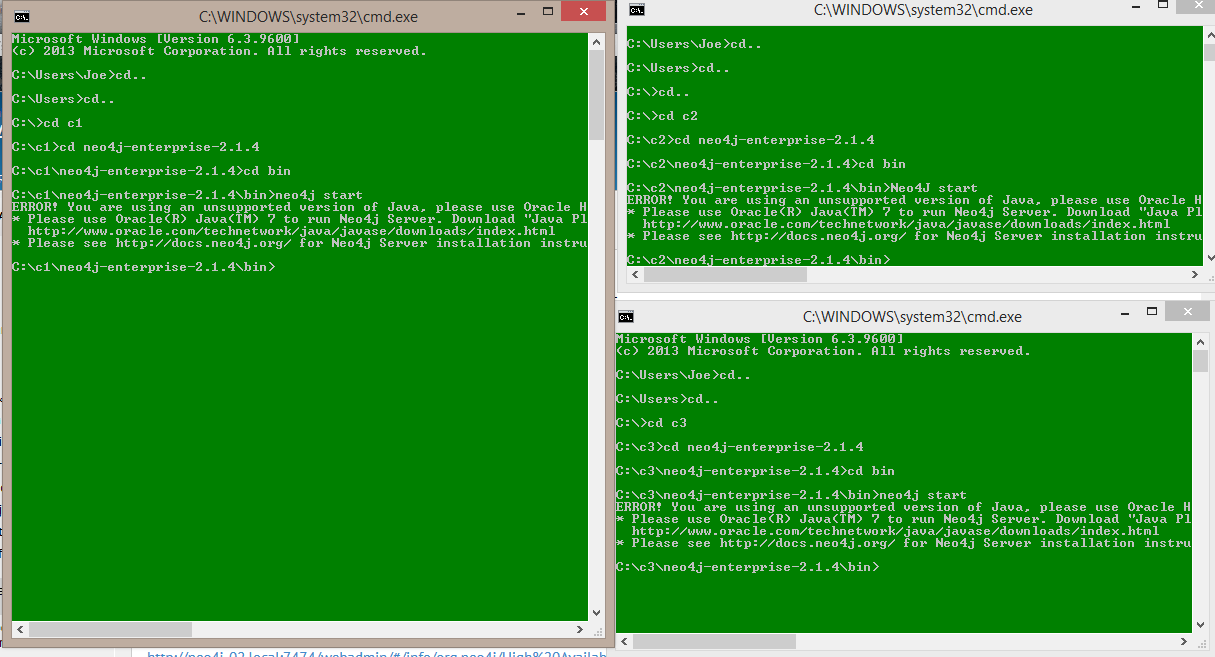
The new nodes and relationship as represented on the web interface.



Query the name of the club the ‘NewPlayer’ node using its relationship type ‘Plays\_For’

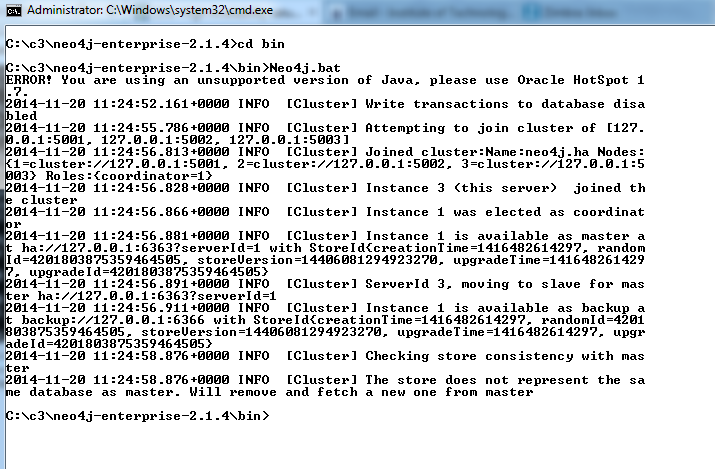
## Install Neo4J HA on Windows and explain with an example how it does High Availability.

The first effort at this question on personal machine caused problems due to the Java jdk version installed on the machine



3 server folders have been created with Neo4J Enterprise installed in each with required changes made to configuration files – checked on Neo4J official documentation. Servers in this instance failed to start due to the JDK problems mentioned.

Instead of changing the JDK on the personal laptop, the remainder of this question was attempted in a PC lab in ITTralee, immediately problems with JDK versions were eliminated.



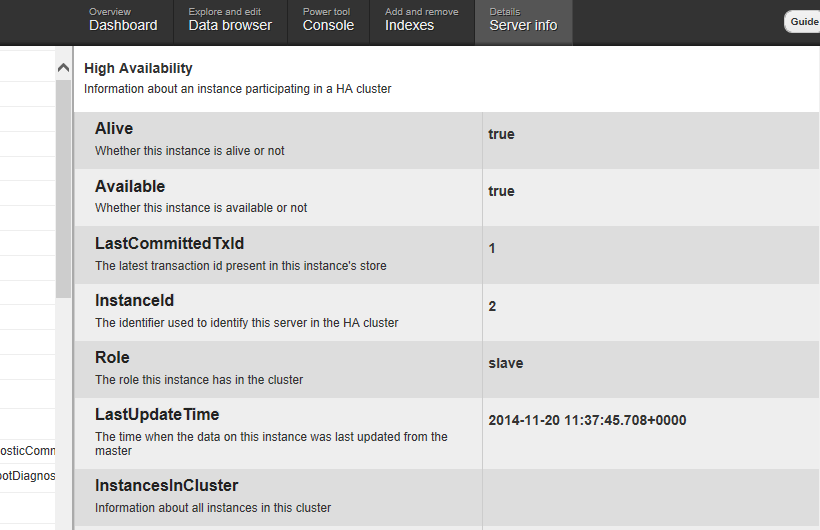




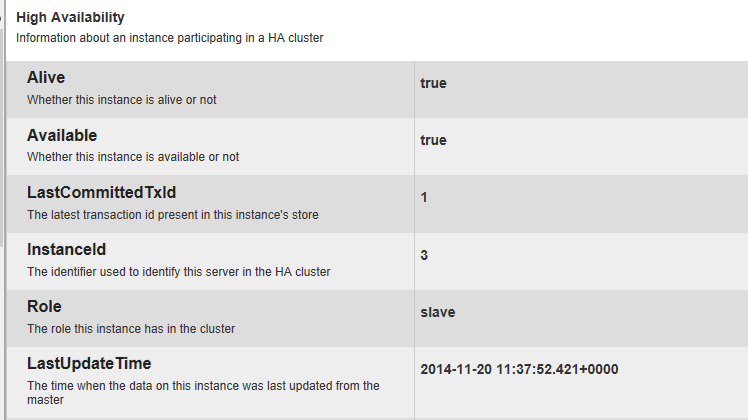
The first instance run attempted to join the cluster but only succeed when the 3rd instance was added. The instance first launched is elected as the Master for the cluster (see screens from web interface below).



**Instance 1: Master.**



**Instance 2: Slave**

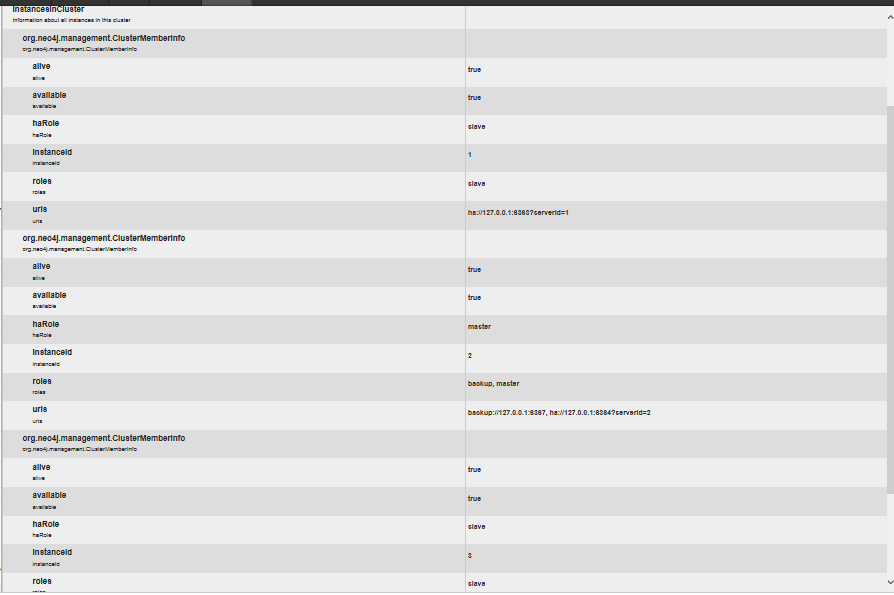


**Instance 3: Slave**

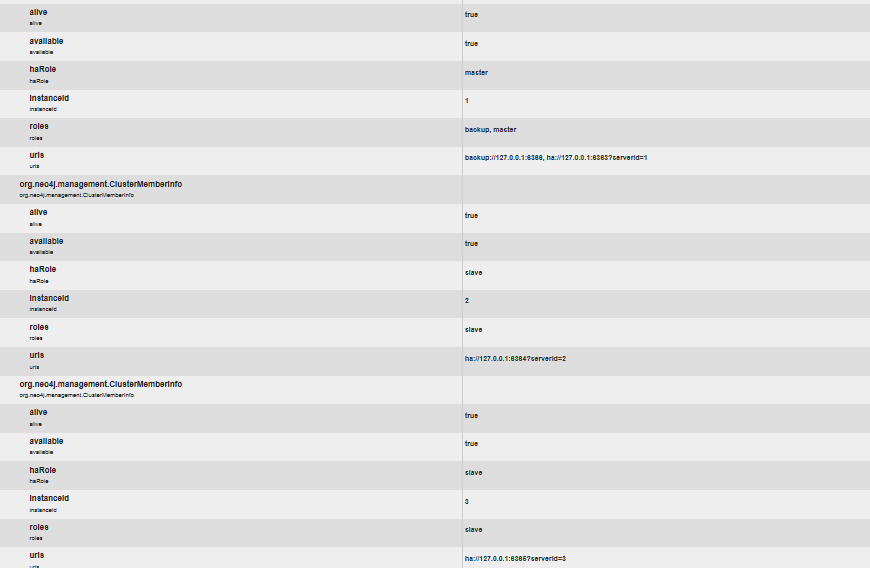
Now we will shut-down the current master server and should see that another server has been elected as a new master, this will be shown via the web interface for clarity.



When the previous master is re-started it re-joins the cluster as a slave.



It will only become **master** again when another server is shut-down and restarted causing a new election to take place, in this example instance 2 has been shut-down and re-started, server 1 has been re-elected as Master for the cluster.



## Conclusion

Unquestionably the most difficult portfolio section to complete due the sheer volume of languages and approaches it contains. Creating a graph was very manageable via the web interface once the graph had been designed on paper. Gremlin, Cypher and using REST to interact with graphs was hugely frustrating for a variety of reasons, mainly lack of knowledge of each method and in Gremlins case forgetting to save the graph. Interacting with Neo4J via a programming language was also an issue for a long period with much trial and error and research required, Python again proving the easiest but not best documented choice. The Neo4J website proved invaluable as a source of information and it comes with a strong recommendation.

<http://neo4j.com/>