1 Type 2B

1.1 Basic Examples

Who is in [noun]? Return list
(ie. Who is in Justice League? Who plays for the Avalanche?)

Who knows [noun]? Return list
(ie. Who knows Bruce?)

1.2 Pseudo Pseudo-Code

Turn string into array of words

Query verb is "static"
if verb is "[verb a]" then query is [noun] [verb a] [noun]
else if verb is "[verb b]" then query is [noun] [verb b] [noun]
else if ...
(ie. [noun] knows/is in/plays/etc... [noun])

Find noun $->$ desired node
Set noun in spot 1
Find other noun $->$ desired nodes
Set other noun in spot 2

Return list of all nodes connected to the first node (query)

1.3 Thoughts

Query results need to be put into a list (array) that can be read by the console (Python).
Style of list can be determined (simple, non-existent, whatever)
GENERAL THOUGHT: Error handling should maybe be used to handle potential misses in the desired verbs. Should allow for re-input of the string sentence.

2 Type 3

2.1 Basic Examples

Who has [quantitative value]?
ie.) Who has appeared in the most movies?, Who has shot more than 10 goals?

2.2 Pseudo Pseudo-Code

Similar to the design of Type 2B, however verb if-else statement would be set to the desired verbs.
Pull desired information from the node OR who has the most edges to certain types of nodes.
Return list of results

2.3 Thoughts

Probably the most difficult type (so far :3).
Need a way to pull the nodes information from Cypher.