Graph-based Machine Learning

# Overview:

Graphs are an excellent way of encoding domain knowledge for your business data from disease detection, genetics, and healthcare to banking and engineering, graphs are emerging as a powerful analysis paradigm for hard problems.

A graph is a collection of nodes (e.g. people) and relationships between them (e.g. Haris is a friend of Osama). Often those nodes have properties (e.g. Haris is age 26).



Neo4j is a popular graph database with native graph storage and processing. Neo4j has its own query language called “Cypher” which is SQL equivalent to a graph database. Below is the snippet of cypher:

MATCH (n1)-[:IS\_FRIEND\_OF]-(n2)

WHERE n1.name = "Haris"

RETURN n2.name

Graph-based machine learning is getting popular nowadays. We can also use the neo4j knowledge graph for machine learning.

But it was always a challenge how we can utilize the knowledge encoded in the knowledge graph to use in machine learning models. But now this is possible with the **Node2vec** algorithm.

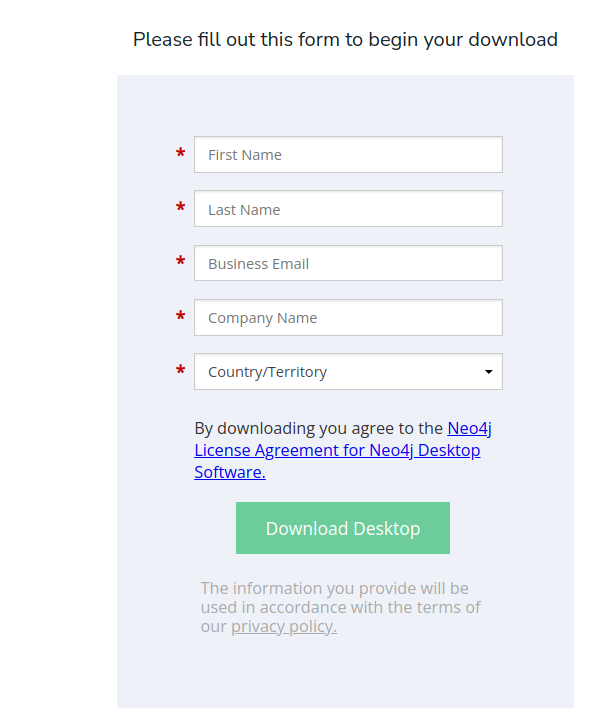
**Node2vec** encodes graph information in node embeddings by taking the graph and its edges. Node embeddings are inspired by the word embeddings and are created using the ***skip-gram*** *model*.

# User Manual:

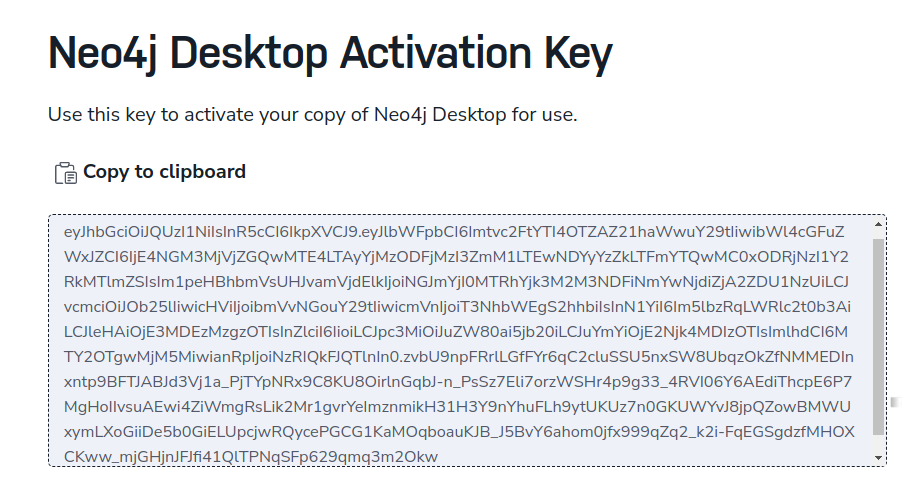
In this manual, we will create a simple graph from the movie lens dataset and then use the graph to build a node2vec model which then will be used to create a movie similarity model.

## Step 1-Neo4j Installation:

* To install neo4j using the .appimage file directly from [Neo4j Desktop](https://neo4j.com/download/)
* Click on download and the form will appear on the screen fill in the form then click on the download button.



* The desktop file will start downloading and you will get the activation key for the Neo4j desktop which you have to save for future use.

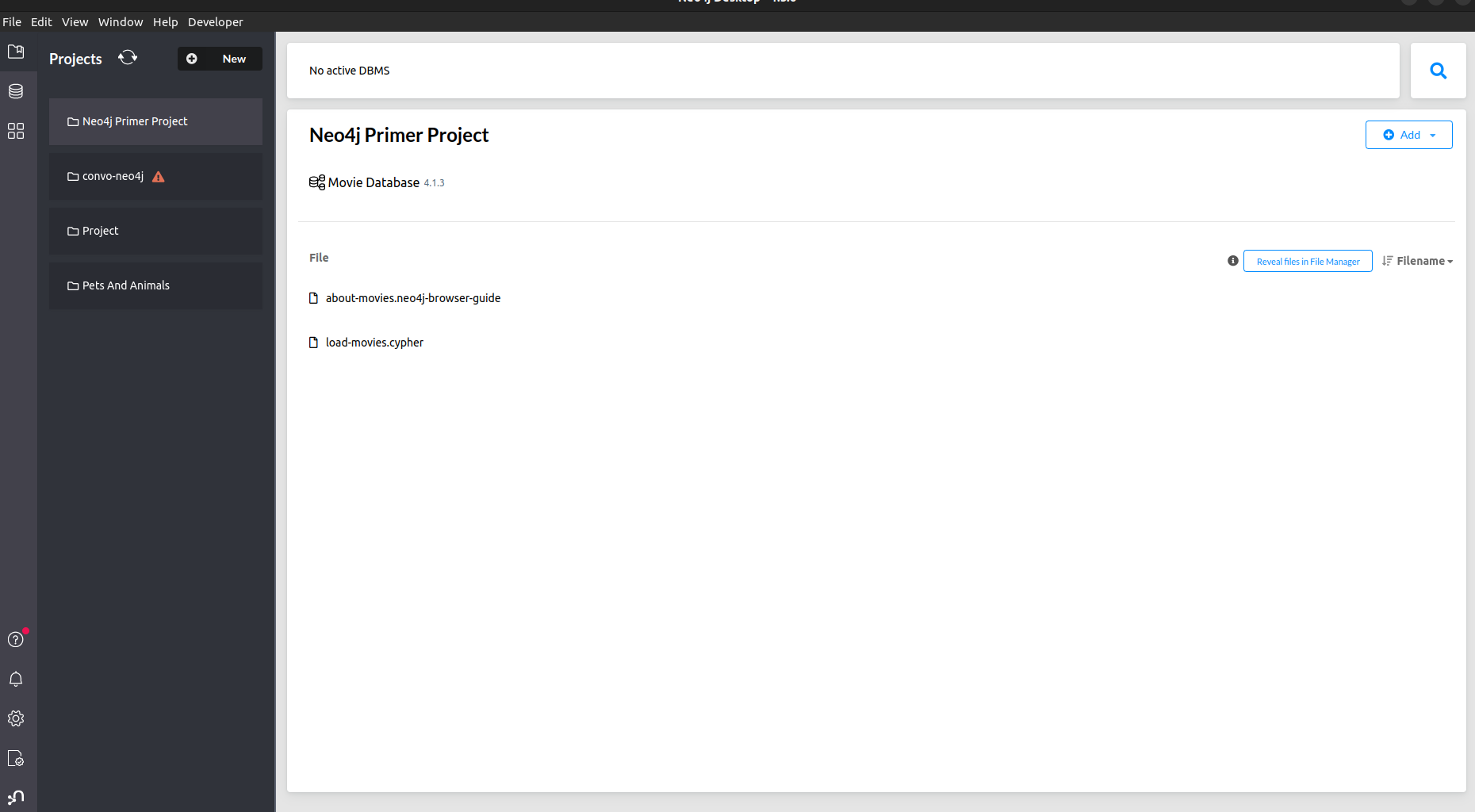


* Once the file is downloaded then create a folder and move the app image into that folder, After moving the folder open terminal and run the command given below:

**chmod a+x neo4j-desktop-1.0.5-x86\_64.AppImage**

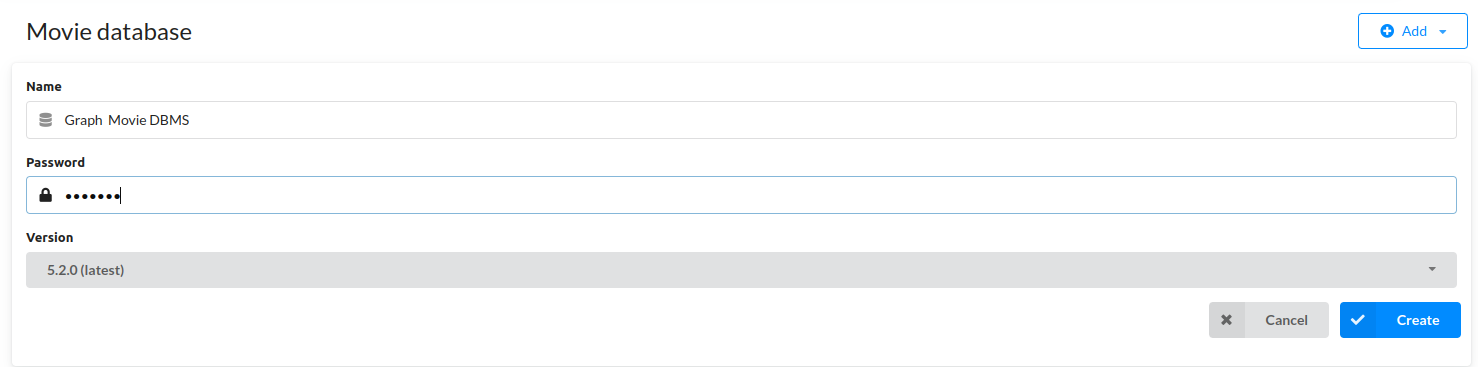
**./neo4j-desktop-1.0.5-x86\_64.AppImage**

* The neo4j will start up and takes some time to configure the settings and it will show the screen.

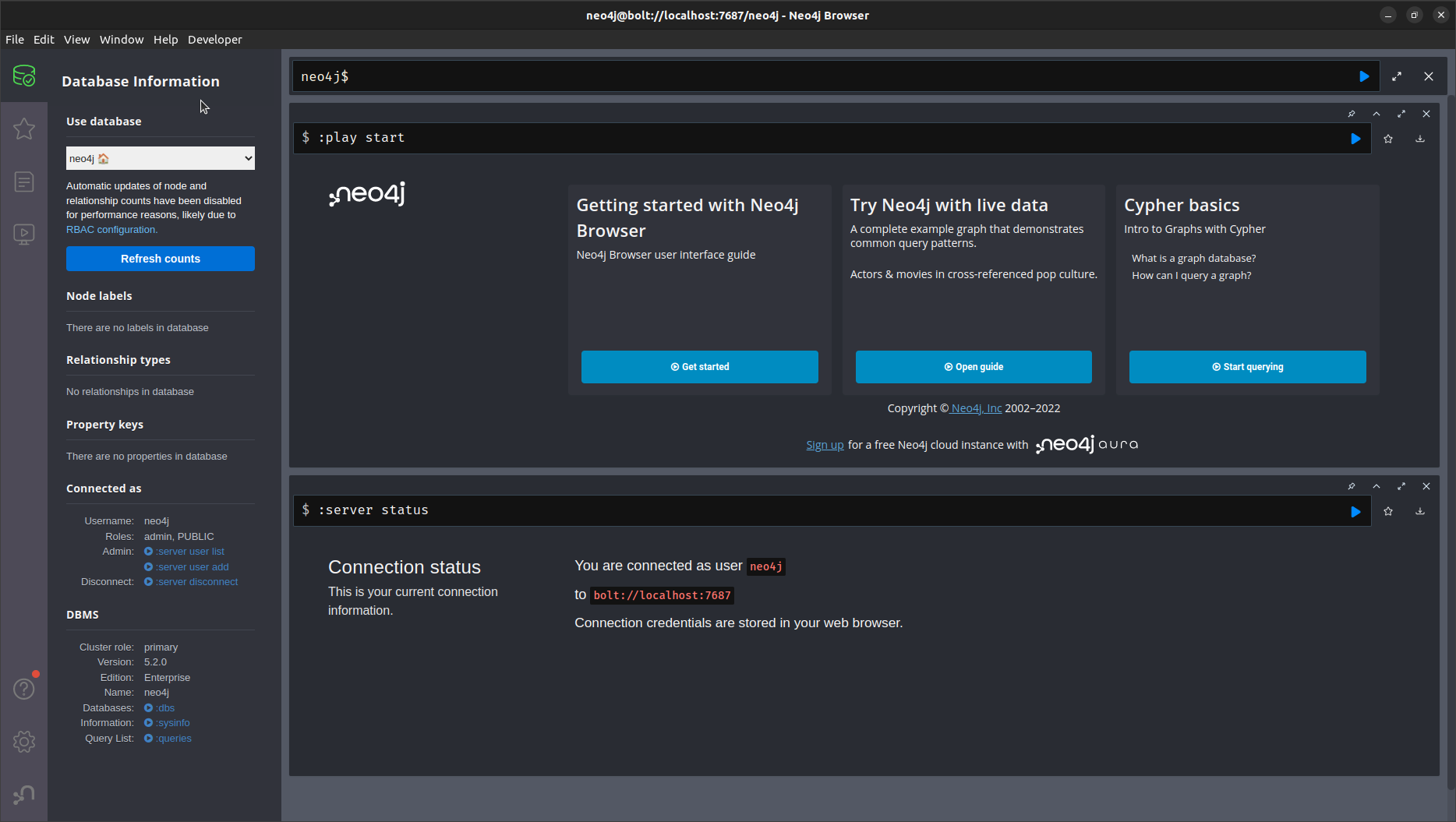


## Step 2-Setting up DBMS and Project:

* Click on the "**New**" button. Select "Create a project” from the options presented.
* Next, Click on the “**Add**” button select “Local DBMS” and then select enter the "Database name" and "Password" in the field and click on the "Create" button.

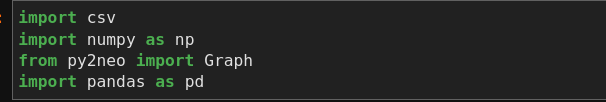


* Once created, Click on the “**Open**” button it will start up the neo4j browser.

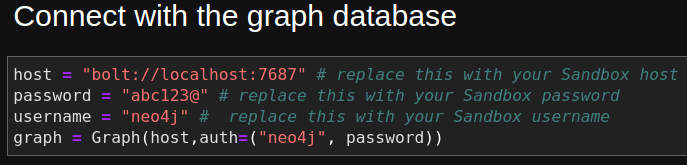


## Step 3- Uploading data into the database:

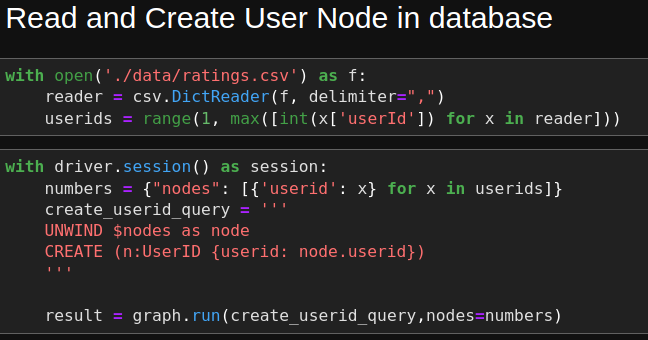
* Setup python environment and install the required packages which are mentioned in requirement.txt.
* Once the environment gets created, Create a folder and create jupyter-notebook in the folder.
* Import the required packages as shown below:



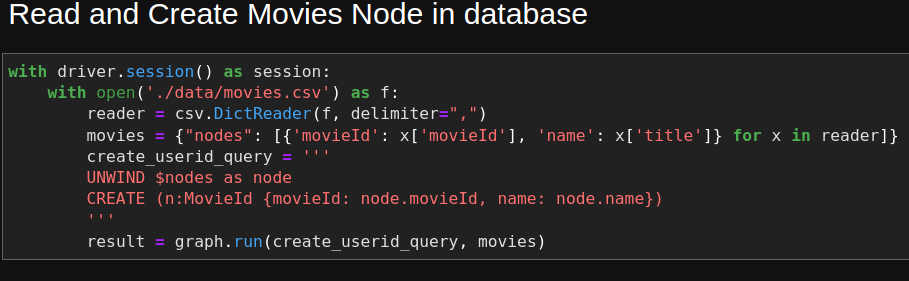
* After that, create a connection with the graph database as shown below:



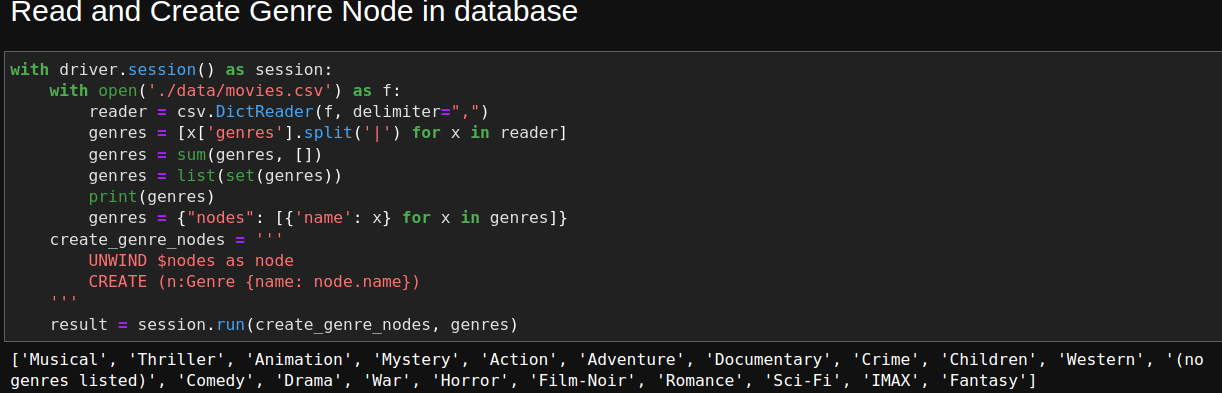
* Now, Create user nodes in the graph by using the ‘Rating.csv’ file as shown below:



* Now, we will create movie nodes in the database by using the “movies.csv” file as shown below:

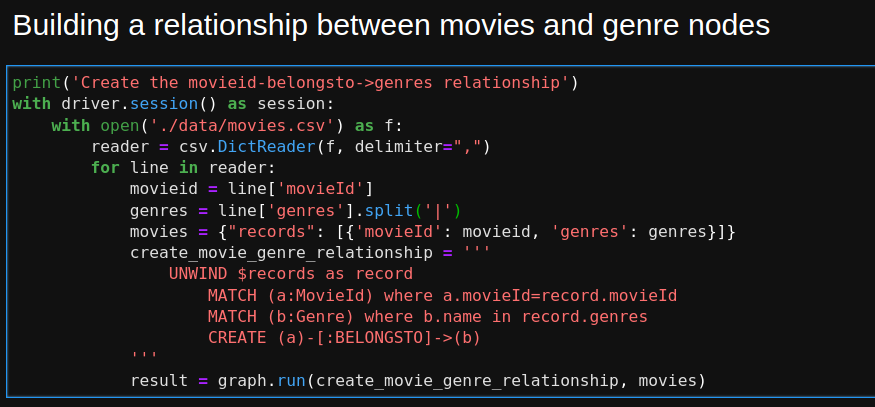


* Now, we will create genre nodes in the database by using the “movies.csv” file as shown below:



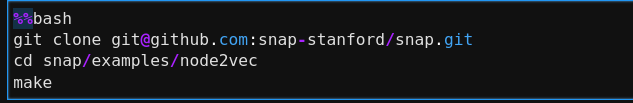
## Step 4- Building Relation between data:

* We have now three types of nodes in the database and will have to build a relationship between them. Below are the types of nodes.
  + Users
  + Movies
  + Genre
* Now we will create relationships between each other as shown below:

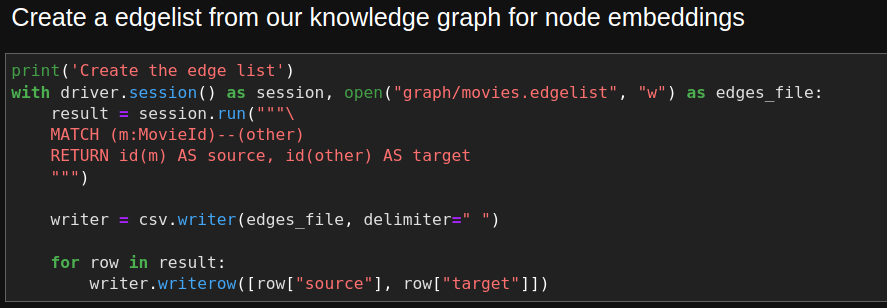


## Step 5- Building node2vec model:

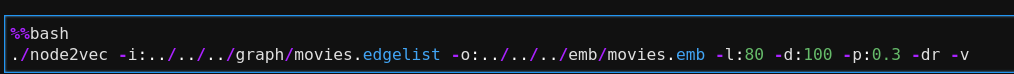
* We will now build node2vec model on our movie knowledge graph which you have built right now. For this we have to install node2vec algorithm as shown below:



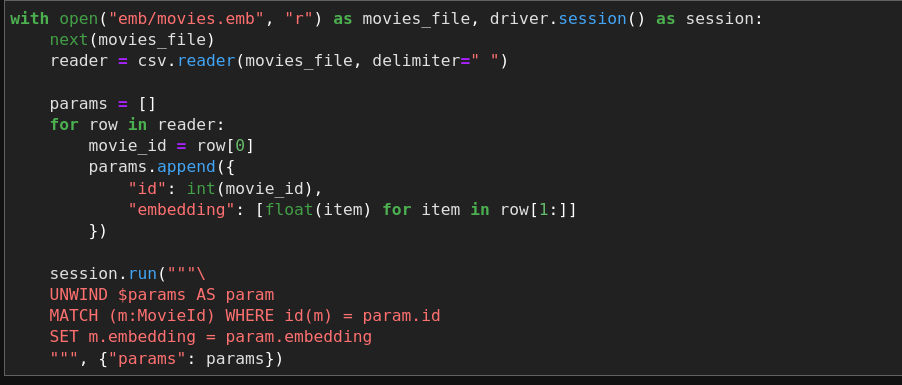
* Now we are going to create an edge list of our knowledge graph for creating embeddings on them.



* Now we build an embedding model using node2vec as shown below:

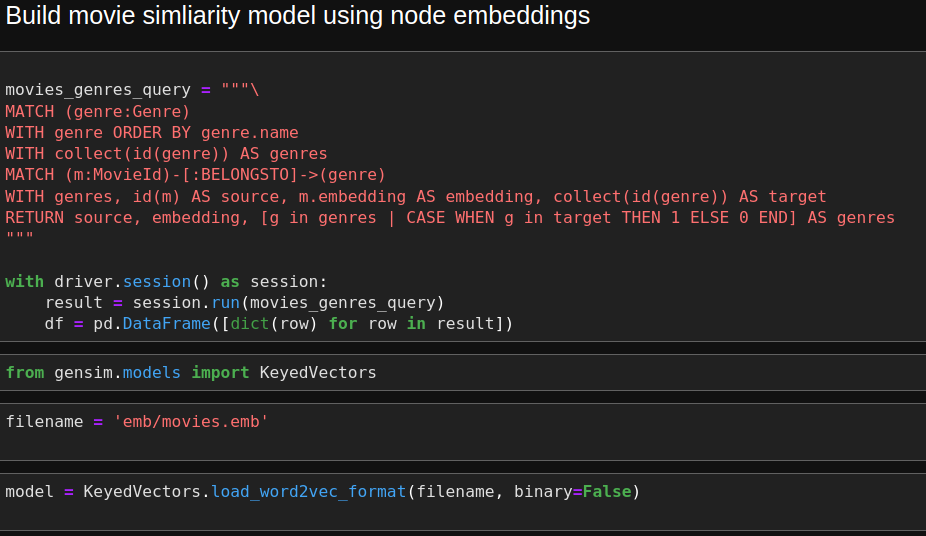


* Now we will create nodes for embeddings in our knowledge graph as shown below:



## Step 6- Building topic similarity model:

* As we created embeddings in our knowledge graph we will now be built our topic similarity model on top of it as shown below:



* Now we will test the model which we have created now as shown below:

