SDN INTRUSION DETECTION

End-User Instructions:

Step 1 - Open the source file [Folder Name: 0 - Source File]. Note the path of the source file.

DATA PROFILING:

Step 2 - Open the Data_Profiling_SDN.ipynb file [Folder Name: 1 - Data Profiling]

1. Start with installing this package -
   pip install pandas-profiling

2. Import the below libraries:
   import pandas as pd
   import pandas_profiling

3. In the 3rd row import the SDN Intrusion Detection Dataset file by giving the path of the source file.

```
In [2]: #Import Pandas and Pandas Data Profiling packages
        import pandas as pd
        import pandas_profiling
```

```
In [3]: #Importing the SDN Intrusion Detection Dataset
        df = pd.read_csv('/Users/sindhuvaddi/Downloads/SDN_Intrusion/0_Source_File/SDN_Intrusion.csv')
```

```
In [4]: #Reading the first 5 rows of the dataset
        df.head()
```

Out[4]:

	Unnamed: 0	Destination Port	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	...	min_seg_size_forward	Active Mean	Active Std
0	0	80	9865922	5	0	30	0	6	6	6.000000	...	20	1986.00000	0.0000
1	1	443	158423	24	22	703	24564	453	0	29.291667	...	32	0.00000	0.0000
2	2	443	61163904	14	12	993	3445	620	0	70.928571	...	20	147216.66670	152989.5108
3	3	443	110544045	18	18	1213	4216	812	0	67.388889	...	20	75305.72727	133115.4997
4	4	53	185	2	2	104	136	52	52	52.000000	...	20	0.00000	0.0000

Step 3 - Access the .html Data Profiling file that is generated.

DATA CLEANSING AND WRANGLING

Step 4 - Open the Data_Cleansing_SDN.ipynb [Folder Name: 2 - Data Cleansing and Wrangling].

Step 5 - Import the below packages
import pandas as pd
import numpy as np
import seaborn as sns

Step 6 - Import the Source Data File

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
```

Read imput csv file

```
In [2]: df = pd.read_csv('/Users/sindhuvaddi/Downloads/SDN_Intrusion/0_Source_File/SDN_Intrusion.csv')
```

```
In [3]: df
Out[3]:
```

	Unnamed: 0	Destination Port	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	...	min_seg_size_forward	Active Mean	Acti
0	0	80	9865922	5	0	30	0	6	6	6.000000	...	20	1986.00000	
1	1	443	158423	24	22	703	24564	453	0	29.291667	...	32	0.00000	
2	2	443	61163904	14	12	993	3445	620	0	70.928571	...	20	147216.66670	15298
3	3	443	110544045	18	18	1213	4216	812	0	67.388889	...	20	75305.72727	13311
4	4	53	185	2	2	104	136	52	52	52.000000	...	20	0.00000	

Step 7 - After executing all the queries for cleansing the data, download the new and transformed CSV file.

Exporting the clean dataframe into csv file

```
In [28]: df.to_csv("/Users/sindhuvaddi/Downloads/SDN_Intrusion/5_Database_Creation_and_load/Cleansed_SDN_Intrusion.csv")
```

```
In [29]: df
Out[29]:
```

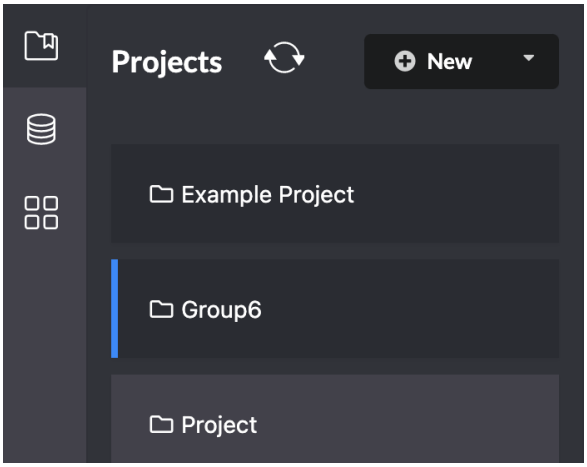
	Intrusion_Id	Destination_Port	Flow_Duration	Total_Fwd_Packets	Total_Backward_Packets	TotalLength_of_Fwd_Packets	Total_Length_of_Bwd_Packets
0	1	80	9865922	5	0	30	0
1	2	443	158423	24	22	703	24564
2	3	443	61163904	14	12	993	3445
3	4	443	110544045	18	18	1213	4216
4	5	53	185	2	2	104	136
...	...	...	...	...	...	...	...
1188328	1188221	138	23	13	0	3029	0
1188329	1188222	50898	7188897	1	5	6	30
1188330	1188223	53	153	2	2	46	46
1188331	1188224	80	1868954	6	0	36	0

The dataset is clean and ready to be imported in the Neo4j graph database.

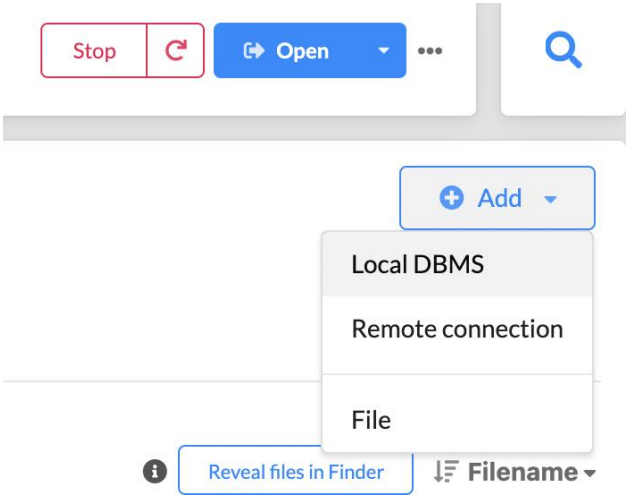
DATABASE CREATION AND LOAD

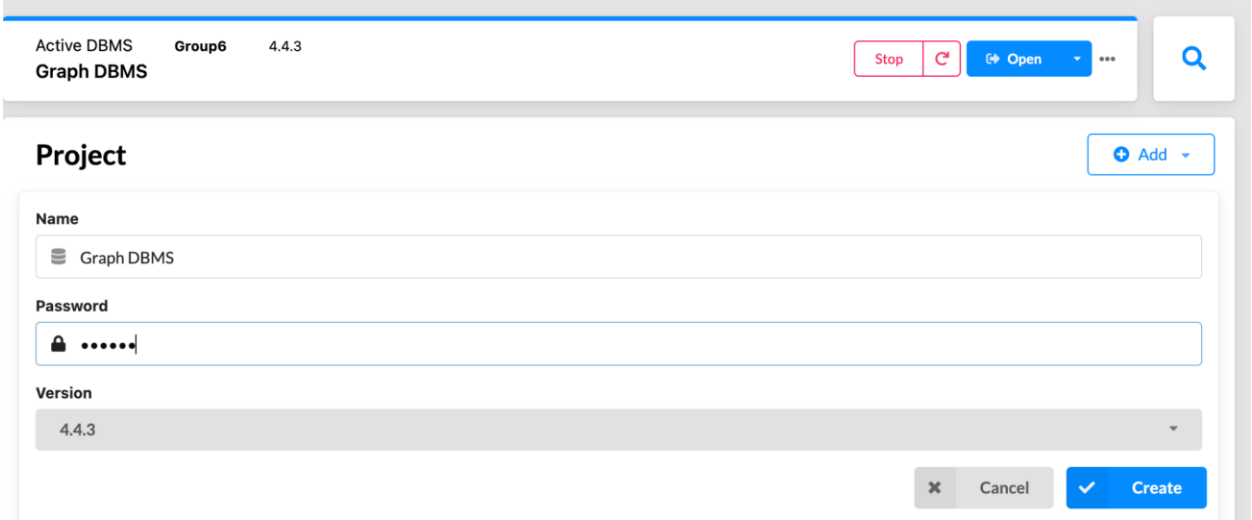
Step 8: Import the extracted dataset from above into NEO4j.
To do so, Open Neo4j

Step 8.1 - From the top left corner, click on create a New Project

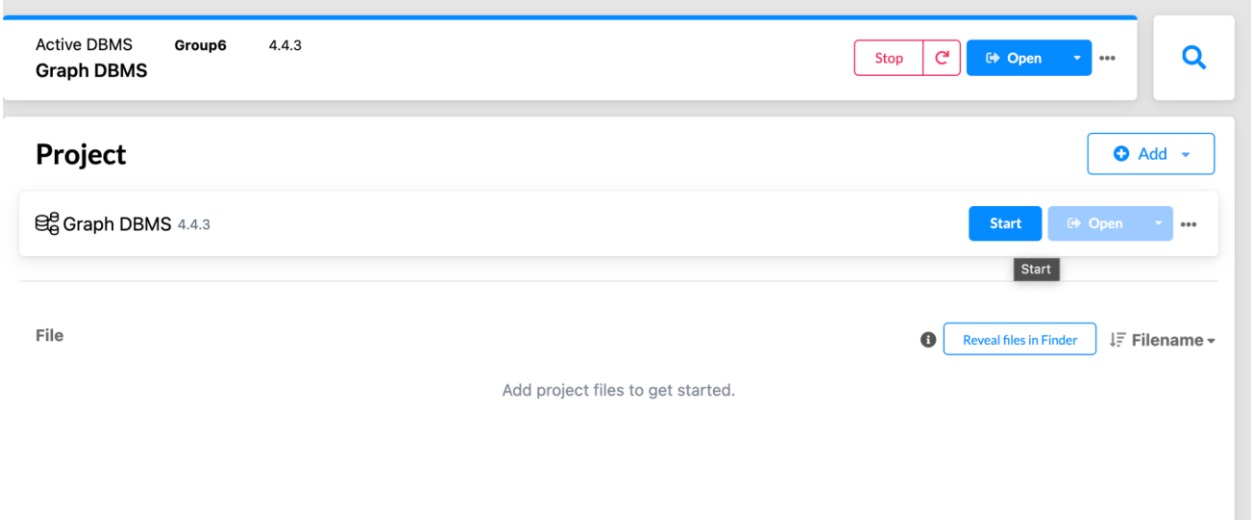


Step 8.2 - From the top right corner, Click on Add → Local DBMS → Set name - Graph DBMS → Set a password (654321) → Create

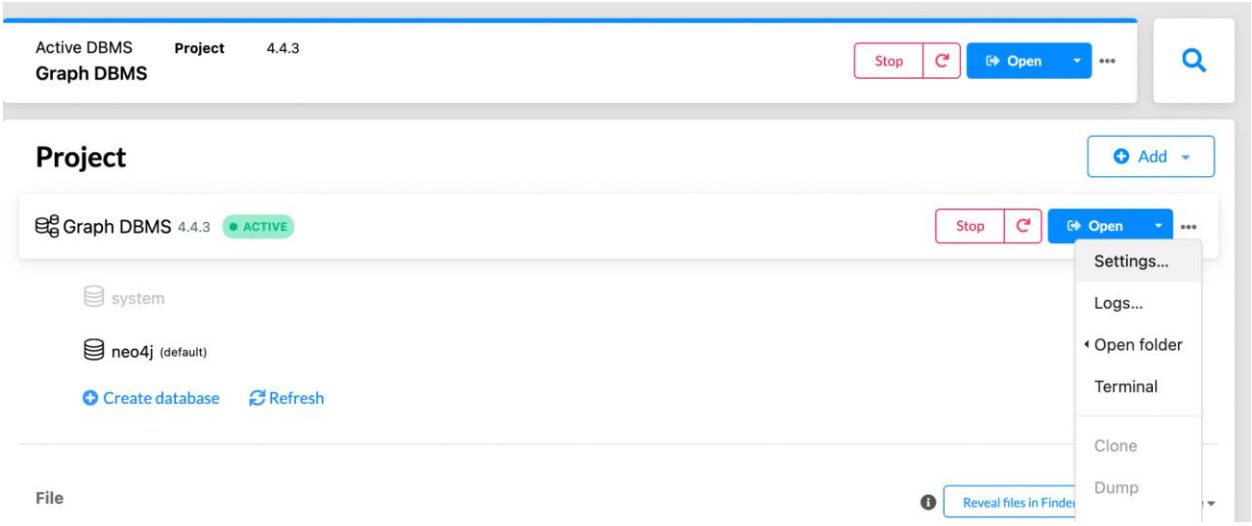




Step 8.3 - Start the Database



Step 8.4 - Click on the three dots (...) in Graph DBMS → Open settings → Disable the security authentication by changing true to false.

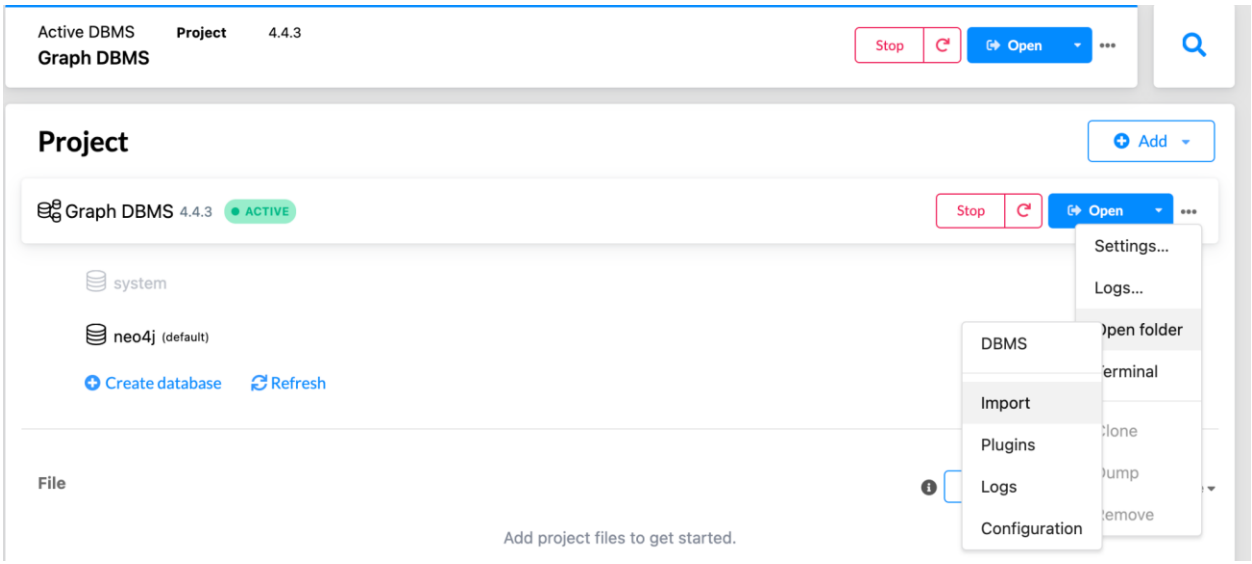


Ignore this step, if done already.

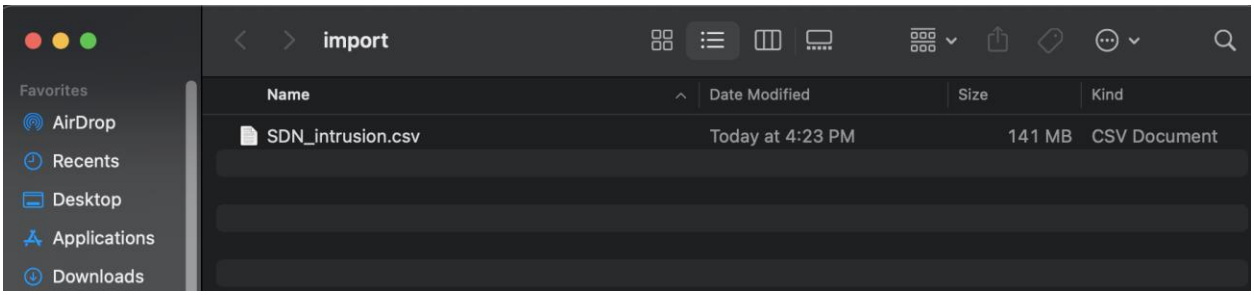
Step 8.5 - Similar to the above step, add **dbms.security.procedures.unrestricted=apoc.*** under # Other Neo4j system properties
Ignore this step, if done already.

```
#*****************************************************************
# Other Neo4j system properties
#*****************************************************************
dbms.security.procedures.unrestricted=apoc.*
dbms.jvm.additional=-Dlog4j2.formatMsgNoLookups=true
```

Step 8.6 - Click on Settings → Open folder → Import

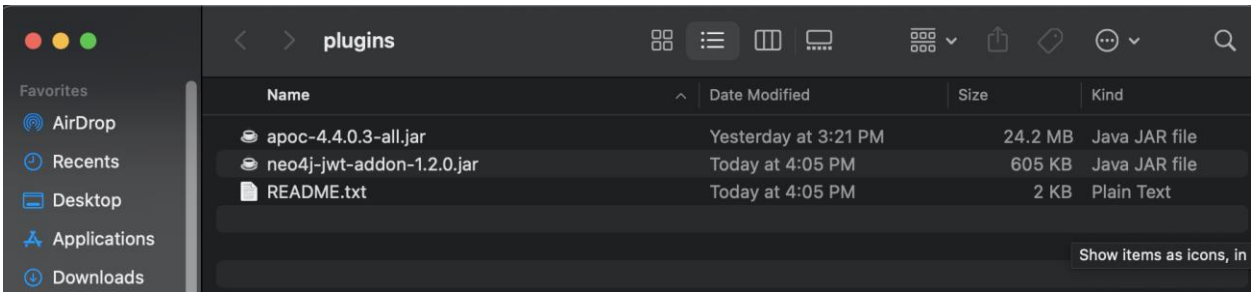


Step 8.7 - Copy the downloaded file (SDN_intrusion.csv) file and paste it into the Import folder.

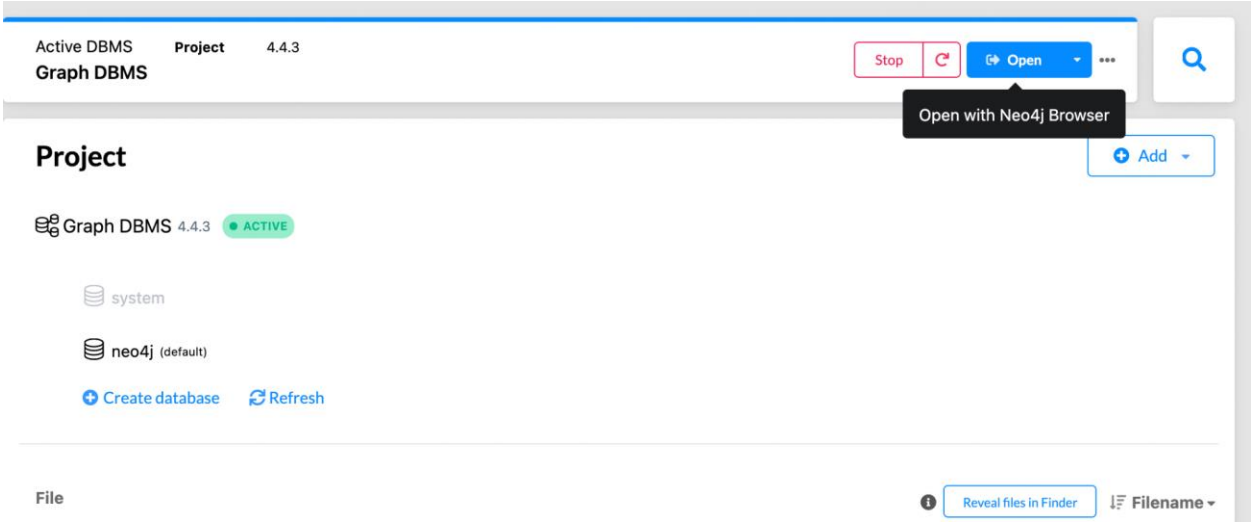


Step 8.8 - Click on Settings → Open folder → Plugins

Copy the apoc-4.4.0.3-all.jar file and paste it into the Plugins folder.



Step 8.9 - Click on Refresh and Finally, click on OPEN in the top right corner.



Neo4j Browser opens now

Step 9 - Copy the code from SDN Intrusion.txt [Folder Name: 5 - Database Creation and load] and execute it in the Neo4j Browser.
Data is finally imported into Neo4j.

DATA VALIDATION:

Step 10 - Run the **SDN_Data_Validation.ipynb** [Folder Name: 6 - Data Validation and Data Visualization] after changing the import path.

```
In [3]: #Importing the SDN Intrusion Detection File after data cleansing

        df = pd.read_csv('/Users/sindhuvaddi/Downloads/SDN_Intrusion/5_Database_Creation_and_load/Cleansed_SDN_intrusion.csv')
```

```
In [4]: #Displaying the top 5 rows of data
        df.head(10)
```

Out[4]:

	Unnamed: 0	Intrusion_Id	Destination_Port	Flow_Duration	Total_Fwd_Packets	Total_Backward_Packets	TotalLength_of_Fwd_Packets	Total_Length_of_Bwd_Pack
0	0	1	80	9865922	5	0	30	
1	1	2	443	158423	24	22	703	245
2	2	3	443	61163904	14	12	993	34
3	3	4	443	110544045	18	18	1213	42
4	4	5	53	185	2	2	104	1
5	5	6	53	672	1	1	79	1
6	6	7	51323	490655	6	2	11607	
7	7	8	53	163154	2	2	70	1
8	8	9	80	1875517	3	6	26	116
9	9	10	53	23800	2	2	64	

Step 11 - Import the original dataset **SDN_Intrusion.csv** [Folder Name: 0 - Source File] to compare the Original dataset with the Final version.

```
In [9]: #Importing the Original data file before cleaing
        Original_df = pd.read_csv('/Users/sindhuvaddi/Downloads/SDN_Intrusion/0_Source_File/SDN_Intrusion.csv')
        Original_df.head(10)
```

Out[9]:

	Unnamed: 0	Destination Port	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	...	min_seg_size_forward	Active Mean	Active Std
0	0	80	9865922	5	0	30	0	6	6	6.000000	...	20	1986.00000	0.0000
1	1	443	158423	24	22	703	24564	453	0	29.291667	...	32	0.00000	0.0000
2	2	443	61163904	14	12	993	3445	620	0	70.928571	...	20	147216.66670	152989.5108
3	3	443	110544045	18	18	1213	4216	812	0	67.388889	...	20	75305.72727	133115.4997
4	4	53	185	2	2	104	136	52	52	52.000000	...	20	0.00000	0.0000
5	5	53	672	1	1	79	161	79	79	79.000000	...	32	0.00000	0.0000
6	6	51323	490655	6	2	11607	26	5840	0	1934.500000	...	20	0.00000	0.0000
7	7	53	163154	2	2	70	168	35	35	35.000000	...	32	0.00000	0.0000
8	8	80	1875517	3	6	26	11601	20	0	8.666667	...	20	0.00000	0.0000
9	9	53	23800	2	2	64	96	32	32	32.000000	...	40	0.00000	0.0000

DATA VISUALIZATION:

Step 12 - Open the python file **Data Visualization_SDN.ipynb** [Folder Name: 6 - Data Validation and Data Visualization]

Step 13 - Install the following packages:

!pip3 install matplotlib --upgrade
!pip install neo4jupyter
pip install neo4j

Step 14 - Import the following libraries:

import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import neo4jupyter
from py2neo import Graph
from pandas import DataFrame

Step 15 - Establish a connection from the Neo4j database.

jupyter Data Visualization_SDN Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Code

In [8]:
```
from neo4j import GraphDatabase
class Neo4jConnection:

    def __init__(self, uri, user, pwd):
        self.__uri = uri
        self.__user = user
        self.__pwd = pwd
        self.__driver = None
        try:
            self.__driver = GraphDatabase.driver(self.__uri, auth=(self.__user, self.__pwd))
        except Exception as e:
            print("Failed to create the driver:", e)

    def close(self):
        if self.__driver is not None:
            self.__driver.close()

    def query(self, query, db=None):
        assert self.__driver is not None, "Driver not initialized!"
        session = None
        response = None
        try:
            session = self.__driver.session(database=db) if db is not None else self.__driver.session()
            response = list(session.run(query))
        except Exception as e:
            print("Query failed:", e)
        finally:
            if session is not None:
                session.close()
        return response
```

In [11]:
```
conn = Neo4jConnection(uri="bolt://localhost:7687", user="neo4j", pwd="654321")
print(conn)
```

Input the username and password as shown above. And run the file to view all the visualizations.