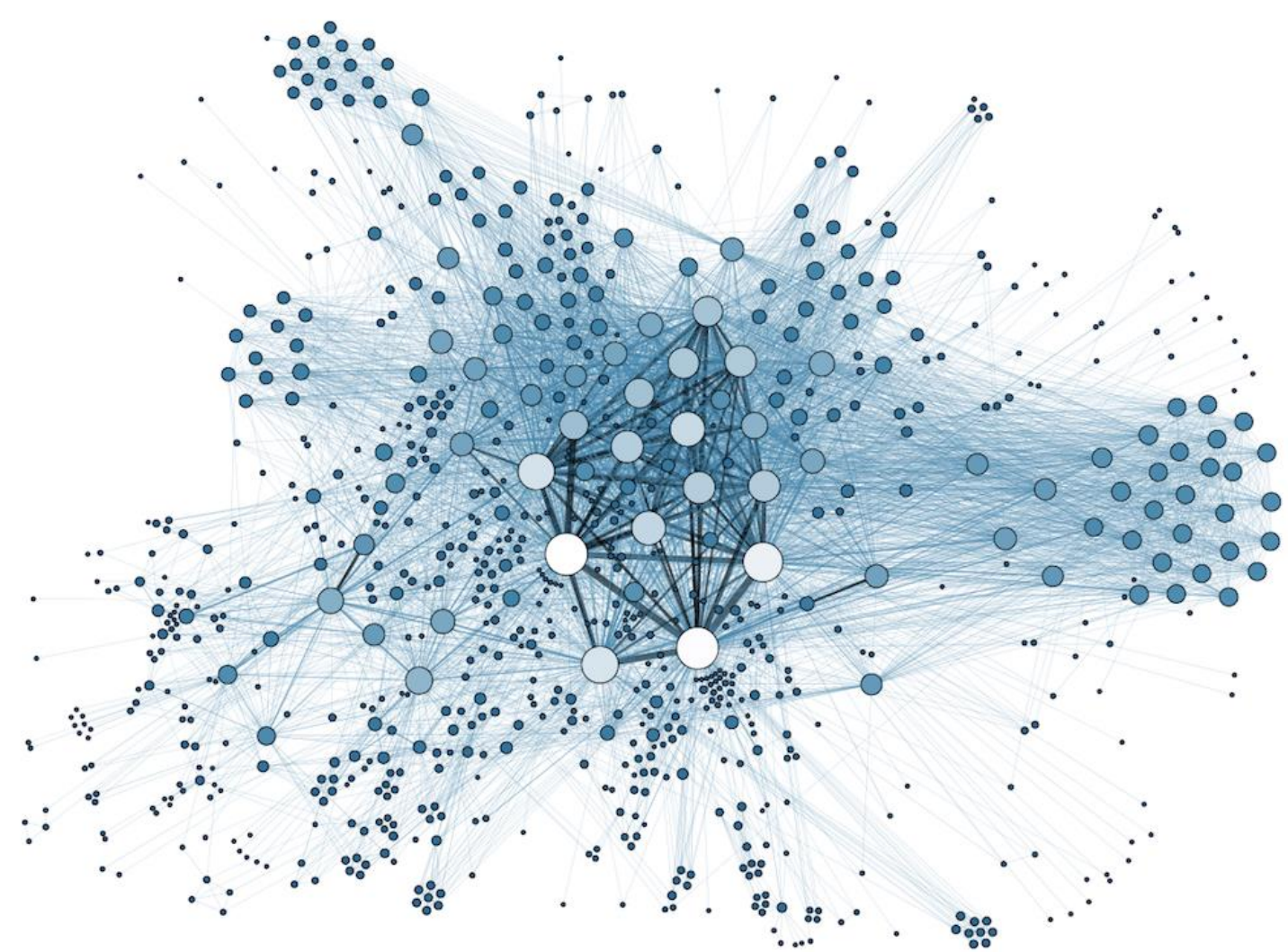
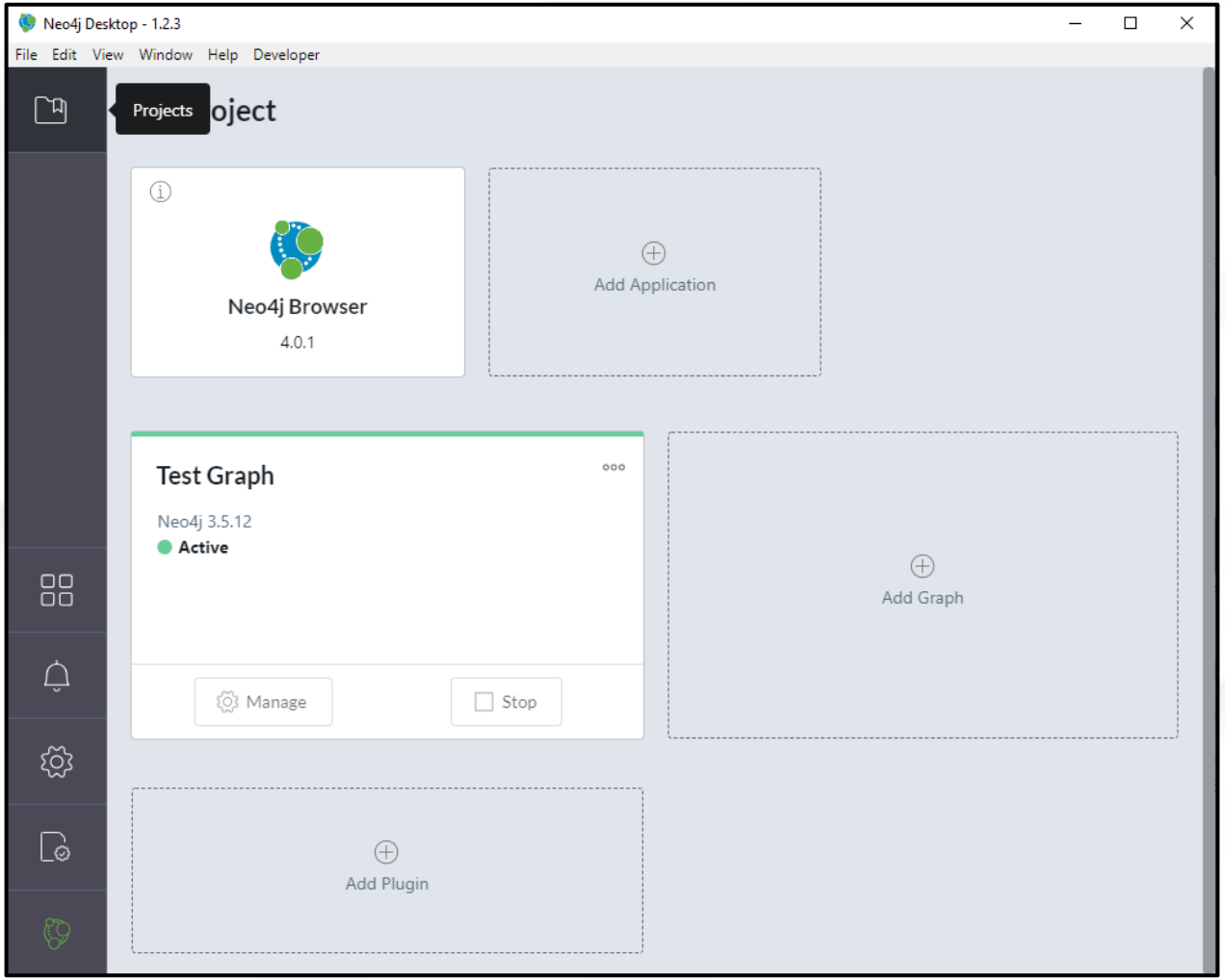


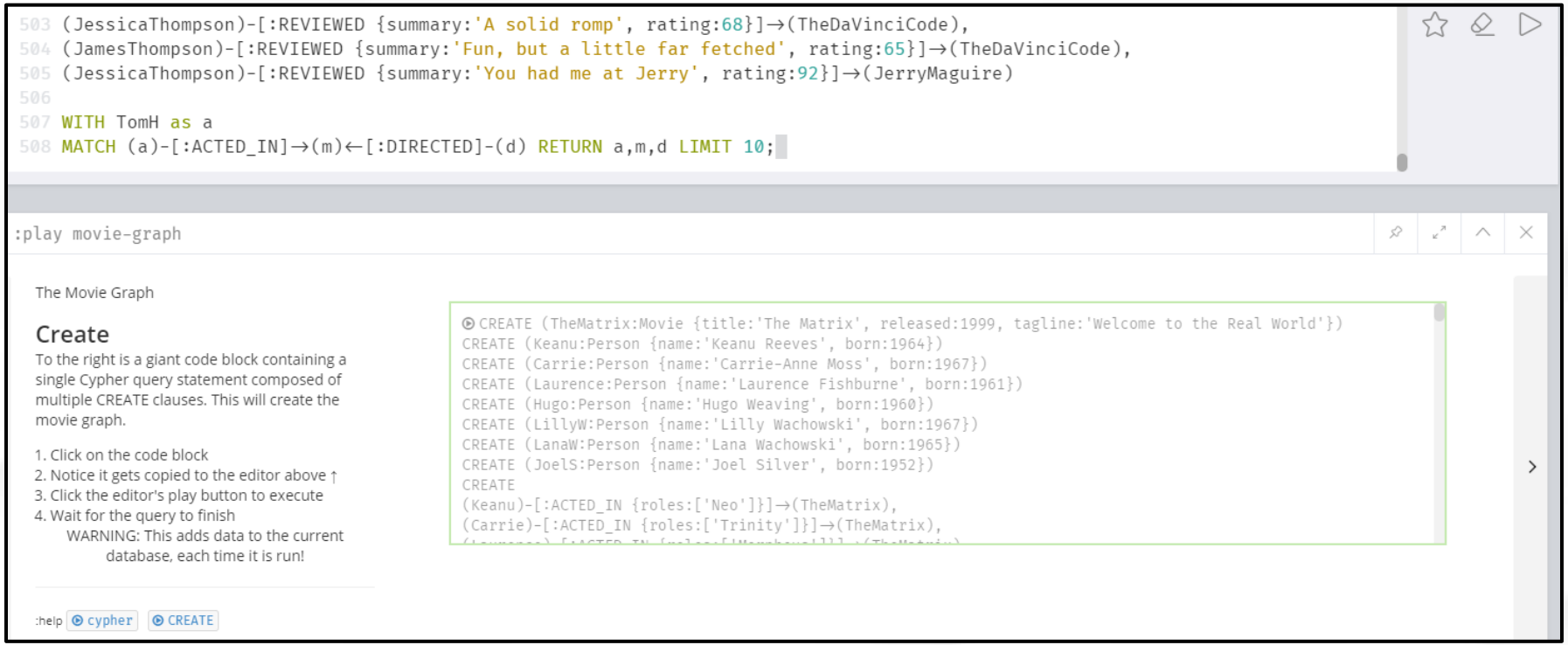
# Assignment #3
# Neo4j Graphs


CSC 131
Quinn Roemer
December 12, 2019

I started my foray into Neo4j by downloading their desktop version of the graph based database.

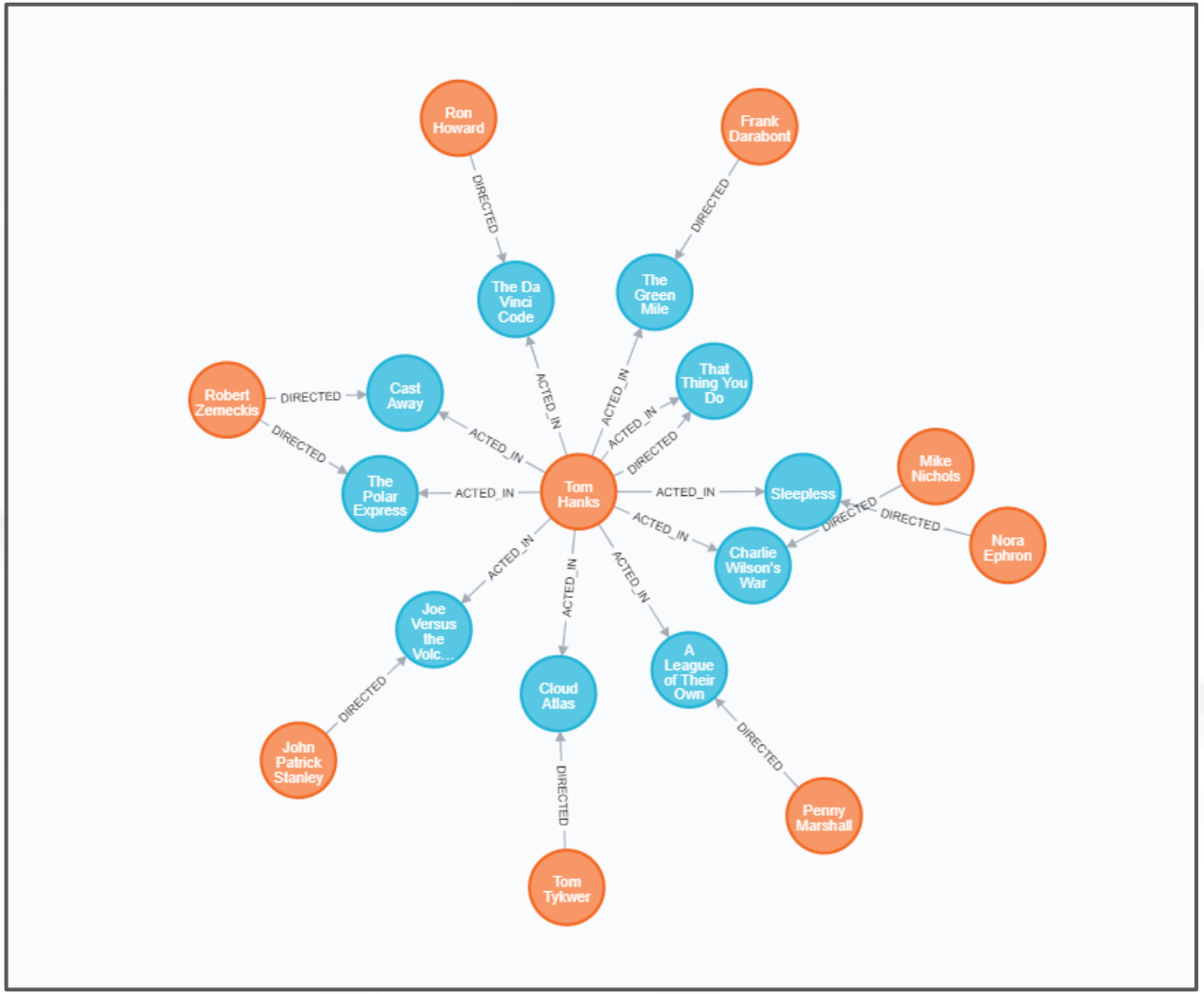
I created a new graph and then activated it to run on my local machine.






I used their built-in learning system to import an existing graph database sample that I could play around with!

To the right, you can see the graph that was created.



- I then began to learn some Cypher queries such as: `MATCH (tom {name: "Tom Hanks"}) RETURN tom`
- I started playing around with relationships between nodes in Cypher. Learning that the language uses a form of ASCII art to represent relationships in queries.
- For example, this query looks for movies that “Tom Hanks” acted in:

```
MATCH (tom:Person {name: "Tom Hanks"})-[:ACTED_IN]→(tomHanksMovies) RETURN tom,tomHanksMovies
```

- I also experimented with some queries that had relationships in both directions.
- This query finds all the co-actors who acted in movies where Tom Hanks also acted in:

```
MATCH (tom:Person {name:"Tom Hanks"})-[:ACTED_IN]→(m)←[:ACTED_IN]-(coActors) RETURN coActors.name
```

**Below, you can see the data that each of the queries returned in the previous slide.**


Query 1


Query 2

| "coActors.name" |
| --- |
| "Ed Harris" |
| "Gary Sinise" |
| "Kevin Bacon" |
| "Bill Paxton" |
| "Parker Posey" |
| "Greg Kinnear" |
| "Meg Ryan" |
| "Steve Zahn" |
| "Dave Chappelle" |
| "Madonna" |

Query 3

**After this, I connected the database to Intellij**


I used Neo4j's plugin for Intellij. This allowed me to connect my database to the IDE. As shown in the picture to the left.

# Here is the following code I wrote to query the database

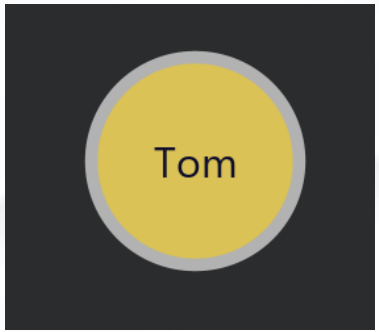
```
import java.io.File;
import org.neo4j.graphdb.GraphDatabaseService;
import org.neo4j.graphdb.factory.GraphDatabaseFactory;

public class neoFun
{
    private static final File dbDir = new File("C:\\Users\\compu\\.Neo4jDesktop\\neo4jDatabases\\database-3eed76ee-5496-430f-80c8-37850a210d44\\installation-3.5.12");

    private static String queryOne = "MATCH (tom {name: 'Tom Hanks'}) RETURN tom";
    private static String queryTwo = "MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]->(tomHanksMovies) RETURN tom,tomHanksMovies";
    private static String queryThree = "MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]->(m)<-[:ACTED_IN]-(coActors) RETURN coActors.name LIMIT 10";


    public static void main (String[] args)
    {
        GraphDatabaseService graphdb;
        graphdb = new GraphDatabaseFactory().newEmbeddedDatabase(dbDir);

        System.out.println("Running query: " + queryOne);
        graphdb.execute(queryOne);
        System.out.println("Running query: " + queryTwo);
        graphdb.execute(queryTwo);
        System.out.println("Running query: " + queryThree);
        graphdb.execute(queryThree);

        System.out.println("Queries complete...");

    }
}
```

**I used the built-in Database Console to view if the queries ran successfully**

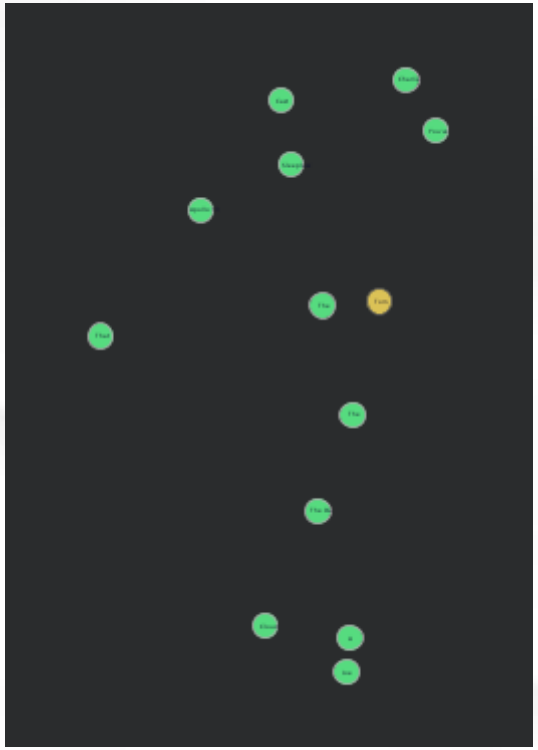
```
Executing query:
MATCH (tom {name: 'Tom Hanks'}) RETURN tom
Query executed in 3ms. Query type: READ_ONLY.
Got 1 rows. View results: as Graph, as Table

Executing query:
MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]->(tomHanksMovies) RETURN tom,tomHanksMovies
Query executed in 3ms. Query type: READ_ONLY.
Got 12 rows. View results: as Graph, as Table

Executing query:
MATCH (tom:Person {name: 'Tom Hanks'})-[:ACTED_IN]->(m)<-[:ACTED_IN]-(coActors) RETURN coActors.name
Query executed in 6ms. Query type: READ_ONLY.
Got 39 rows. View results: as Table
```

As you can see, all the queries were successful!

**Here is the output from each query in Intellij**




Query 1



Query 2

| coActors.name |
| --- |
| "Ed Harris" |
| "Gary Sinise" |
| "Kevin Bacon" |
| "Bill Paxton" |
| "Parker Posey" |
| "Greg Kinnear" |
| "Meg Ryan" |
| "Steve Zahn" |
| "Dave Chappelle" |
| "Madonna" |

Query 3

It was great fun experimenting with Neo4j and learning how to query it from Java!

Below you will find a link to download my code

https://drive.google.com/open?id=1dKKkaCG1pSq-vJbQ6BNQDMISpTf6TGTN

Or, you can just view the “neoFun.java” file that was uploaded with this presentation.

Please note, the code will not run properly because the database only exists on my computer.