

UV Distributed Information Management

Summer semester 2024

Assignment 03

Summary:

Deadline: June 27, 2024, 11:55 pm (aka 23:55) CET.

Extended Deadline: July 04, 2024, 11:55 pm (aka 23:55) CET.

Submission: Submit a compressed archive (e.g., a zip or a tar .gz file) that contains your Python3 code and your answers to the questionnaire via Blackboard.

Grading: 50% Python3 code, 50% answers (incl. meeting; cf. Section 5 for details).

1 Introduction

In the course of this assignment, we will learn emph to interact with **Neo4j**, a NoSQL database system (DBS) that is based on the graph data model. Just like PostgreSQL, Neo4j ¹ is open-source ² and rather easy to install. Since the graph data model is very intuitive, we will also make use of the graphical user interface (GUI) of Neo4j to interact with the database. However, also Neo4j provides (1) a command-line tool named `cypher-shell` (where **Cypher** stands for Neo4j's **Cypher query language** ³), and (2) an application programming interface (API) for the Python3 programming language ⁴ (i.e., using the `neo4j-driver` ⁵ module).

1.1 Grading & Submission

For your convenience, Section 5 contains the grading scheme that will be applied for this assignment.

Please submit your final Python3 code **and** your answers to the questionnaire until June 27, 2024, 11:55 pm (aka 23:55) CET via Blackboard ⁶ (late submission until July 04, 2024, 11:55 pm (aka 23:55) CET). Furthermore, please keep in mind that the exams contribute 40% to your final grade, hence you need to submit at least one assignment (partially) to pass the course.

1.2 Formatting Conventions

Commands for the Linux command-line tool (`terminal`) ⁷ and the command-line tool of Neo4j (`cypher-shell`) are written in TrueType font ⁸. In addition, all commands are in a box that

¹Neo4j: <https://de.wikipedia.org/wiki/Neo4j>

²Neo4j code on GitHub: <https://github.com/neo4j/neo4j>

³The Cypher query language: [https://en.wikipedia.org/wiki/Cypher_\(query_language\)](https://en.wikipedia.org/wiki/Cypher_(query_language))

⁴The Python programming language: [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))

⁵<https://neo4j.com/docs/api/python-driver/current/> and <https://pypi.org/project/neo4j-driver/>

⁶Blackboard: <https://elearn.sbg.ac.at>

⁷Command-line tool: https://en.wikipedia.org/wiki/Command-line_interface

⁸TrueType font: <https://en.wikipedia.org/wiki/TrueType>

specifies the used command-line tool at the beginning of the title (separated by a dash `-`, i.e., **terminal** for Linux and **cypher-shell** for Neo4j). Listing 1 shows an example command executed in the Linux terminal:

The Linux terminal shows a prefix `dbtutorial@database-tutorial:~#` that consists of

- the name of the user that executes the command; `dbtutorial` is the default user of the virtual machine (VM) (this may be different if you do not use the given VM),
- the name of the machine; `database-tutorial` is the name of the given VM, and
- a delimiter that separates the actual command from the “user@machine” string; “`:~#`” is the default delimiter of the given VM (where `~` denotes the current directory).

Moreover, we use a color code for readability:

- A **command** itself (which is to be copied) is depicted in bold, solid **black**.
- **Prefixes**, which are **not** part of a command itself, are shown in `gray`.
- **Outputs**, which are the result of a command that has been executed, are shown in `green`.

```
Listing 1: terminal – Show directories.
1  dbtutorial@database-tutorial:~# ls -l
2  total 32
3  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Mar 31 15:43 Desktop
4  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Documents
5  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Downloads
6  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Music
7  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Pictures
8  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Public
9  drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Templates
10 drwxr-xr-x 2 dbtutorial dbtutorial 4096 Feb 28 2022 Videos
```

Contrarily, Listing 2 exemplifies a command in Neo4j’s command-line tool, i.e., the `cypher-shell`. It shows an example that executes the command `SHOW DATABASES` while connected to a database named `neo4j`:

```
Listing 2: cypher-shell – Show all databases.
1  neo4j@neo4j> SHOW DATABASES;
```

Notably, Neo4j provides a graphical user interface to the `cypher-shell` and you are free to choose either the command-line tool or graphical user interface to solve this assignment. For commands that are to be executed in the `cypher-shell`, the non-bold prefix `neo4j@neo4j>` denotes the user that is connected in combination with the database we are currently connected to (separated by a “`@`”; cf. Section 2.3). Once again, the Cypher query language (CQL) is different from both SQL and MQL, and capitalization of keywords (like `SHOW`) is optional. For readability, we capitalize Neo4j keywords. More information can be found in the supplementary material given in Section 4.

Python³⁹ code is simply wrapped in a box without specifying any command-line tool, and we provide a `.py` file that contains the Python3 code (in order to make using the template easier).

1.3 Support

Remark: Please notify the instructor as soon as possible if this assignment description is (partially) unclear or if there is a problem with the submission.

If you have trouble understanding this assignment, please use one of the following communication channels to get help (in this order):

⁹The Python Programming language: [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))

1. **Lecture:** wednesdays 01:00 - 02:30 pm CET (exception: lecture-free periods).
2. **Slack:** <https://dbteaching.slack.com/archives/C06MFP830GH> (I will check regularly and do my best to reply fast, but please do not expect me to be available 24/7).
3. **Email:** dkocher@cs.sbg.ac.at (as a last resort).

We recommend to start the assignment early. In case of a problem, it is easier for the instructor and other students to provide help in time if you identify problems early.

2 Assignment Description

We **highly recommend** that you read this section (including all the subsections) to the end before you start working (this should be less error-prone).

Similar to Assignment 1 and 2, we split this assignment into five parts and only part 1 (depending on your choice), 4, and 5 contribute to the overall grade of this assignment. We recommend to follow these steps and refer to the corresponding sections for more details.

1. Set up and configure Neo4j, cf. Section 2.2 and Section 2.3.
2. Fill the database with some data, cf. Section 2.4.
3. Familiarize yourself with Neo4j and the Cypher query language, cf. Section 2.1.
4. Write an example application that accesses the database in Python3; cf. Section 2.6.
5. Answer the questionnaire, cf. Section 2.7.

2.1 Introduction to Graph Databases and Neo4j

Neo4j is a NoSQL¹⁰ database system that complies to the graph data model¹¹. Graphs¹² are a rather intuitive concept in computer science that is almost a perfect fit for certain application scenarios. Nowadays, for example, most people are familiar with social networks (Facebook, Twitter, Instagram, and the likes) and navigation systems (e.g., Google or Apple Maps). From a (very) simplified point of view, social networks resemble specific parts of the real world like the social connections (i.e., the network) and the preferences of a person (e.g., through a like-based system). A graph consists of nodes that represent the actual data, and (directed) edges that represent the interrelations in the data (i.e., the nodes). As an example social network, we consider nodes that represent persons and the connections between the persons are the edges. Typically, a graph data model supports different types of edges and nodes (with numerous different attributes), which is referred to as *property graph* model. Figure 1 shows an example property graph that consists of four nodes (three persons; one movie) and four (directed) edges that encode the relationships between the nodes. For example, Mark and Christian both know Scarlett, but Scarlett does not know any person in this network. Additionally, Mark and Christian both like the movie “American Psycho” (resembled as edges of type “like”). For better readability, the node/edge types are color-coded, i.e., nodes of type “Actor” are shown in **grape**, nodes of type “Movie” are encoded in **green**, whereas edges of type “like” and “knows” are visualized in **orange** and **black**, respectively.

The property graph model of Neo4j also supports node labels that represent the “role” of the node (and can also be used to store meta data). Edges are typically directed (i.e., have a direction) but support the navigation in both directions. For more details on the property graph model of Neo4j and how to model your data as graph, we refer to the Neo4j documentation^{13 14}.

¹⁰NoSQL: <https://en.wikipedia.org/wiki/NoSQL>

¹¹Graph database: https://en.wikipedia.org/wiki/Graph_database

¹²Graph in computer science: [https://en.wikipedia.org/wiki/Graph_\(abstract_data_type\)](https://en.wikipedia.org/wiki/Graph_(abstract_data_type))

¹³Graph model: <https://neo4j.com/docs/getting-started/get-started-with-neo4j/graph-database/>

¹⁴Modeling: <https://neo4j.com/docs/getting-started/data-modeling/relational-to-graph-modeling/>

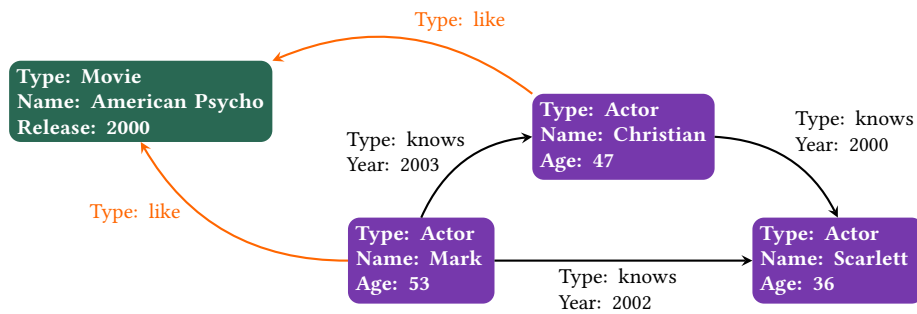


Figure 1: Example graph of a small social network.

We conclude with Figure 2, which depicts the change in popularity per database category¹⁵ (i.e., data model) over time, starting from 2013. Despite the fact that the popularity of graph databases dropped in 2023, Figure 2 shows that graph databases are highly relevant.

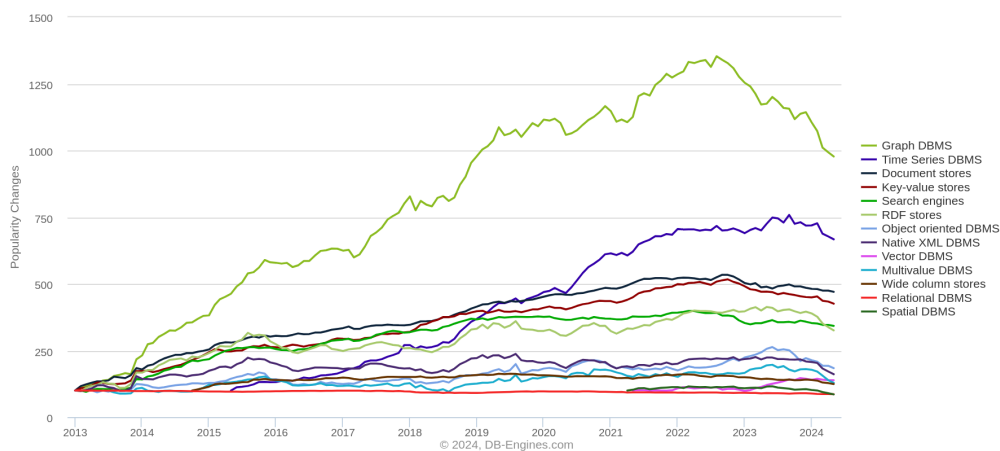


Figure 2: Popularity changes per database category since 2013 (<https://db-engines.com>).

2.2 Neo4j Setup

Similar to Assignment 1 and Assignment 2, you have two possible ways to set up and configure Neo4j in order to solve this assignment:

Neo4j Installation Install Neo4j on your own system (or virtual machine) using an operating system of your choice. This implies that you “pollute” your system with this installation and that the instructor may need additional information on your particular setup to provide help (since we mostly work on Linux systems). Nonetheless, we want to emphasize that it is an **excellent exercise** for students to perform the installation of such a system on their own (at least once in their career). If you choose this option, you can jump to Section 2.2.1.

Neo4j using a VM Configure a pre-installed Neo4j instance in a virtual machine (VM) that runs Debian Linux. This implies that you will not experience the process of installing Neo4j on your own, but it may be easier for the instructor to help since the VM runs Debian Linux (and, ideally, the setup is reproducible). On the one hand, it may be cumbersome if you do not have any experience with Linux systems (and VMs), but on the other hand

¹⁵DB-Engines popularity change: https://db-engines.com/en/ranking_categories

it may also be a nice opportunity to familiarize yourself and play around with a Linux system (and a VM). **Remark:** We want to note that **Neo4j may run slowly in the VM**, in particular when using the graphical user interface. To continue with this option, you can jump to Section 2.2.2.

From our experience, the installation of Neo4j should not cause many troubles. Hence, we encourage everyone to give it a try. You can choose your preferred way and the choice itself will not influence your grade in any way. However, make sure that you can answer questions regarding your choice.

2.2.1 Neo4j Installation

Remarks: (i) If you decide to use the VM image, you can directly jump to Section 2.2.2. Neo4j is already pre-installed. (ii) Please use the command-line tool of your operating system to call Neo4j's command-line tools. On Windows systems, for example, use the `cmd.exe` or the Powershell (and navigate into the directory that contains Neo4j's command-line tools, i.e., the directory that you chose during installation).

As a first step, we are required to install Neo4j (Community Server edition) in version 5.18.0¹⁶ (or higher) locally on your machine.

Remark: Please make sure to use version 5.18.0 (or higher) as some commands may be different for older versions of Neo4j, and the data to import into the database may be incompatible with older Neo4j versions.

Neo4j should be available for all popular operating systems, but the specific steps may diverge. All the details on how to install Neo4j on different operating systems can be found in the Neo4j documentation (cf. links given in Section 4). Like many other systems, Neo4j has a (web-based) graphical user interface (GUI; that can be access using your browser) and a command-line tool to interact with the database. In this assignment, we encourage you to **use both**. Since the web-based GUI provides a way to query the data, everything but the data import and the Python3 code can be done using the web-based GUI. Nonetheless, we encourage you to try out the Neo4j command-line tool. The `cypher-shell` command-line tool (aka **cypher terminal**) is the most basic way to interact with this particular graph database. In order to import the data into Neo4j, the `neo4j-admin` command-line tool can be utilized.

2.2.2 Virtual Machine Setup

Remark: Neo4j may run slowly in the VM, in particular when using the graphical user interface. This is mainly due to the higher resource requirements of the graphical user interface. If possible, you may want to consider giving the VM more memory (and CPUs) in order to speed it up.

We provide a so-called virtual machine (VM) **image**, which consists of a pre-installed Debian Linux as operating system and a pre-installed Neo4j instance. For this option, you may have to familiarize yourself with the concept of a virtual machine (VM) and how to host the corresponding image (cf. Assignment 0 for details). Essentially, a virtual machine¹⁷ is a software that is designed to provide you with a substitute of a real machine, that is, it runs another operating system on top of your regular operating system – it **virtualizes** the hardware of another system. For Assignment 3, it suffices to install a software named VirtualBox¹⁸ that acts as a host for virtual machines. After you successfully installed VirtualBox, the VM image can be used to set

¹⁶Neo4j download center: <https://neo4j.com/download-center/#community>

¹⁷Virtual machine: https://en.wikipedia.org/wiki/Virtual_machine

¹⁸VirtualBox: <https://www.virtualbox.org/manual/ch02.html>

up the virtual machine for this assignment. The image (as ova file) can be downloaded from a Google drive ¹⁹ and runs Debian Linux ²⁰ as operating system with an installation of Neo4j.

After downloading the `debian-vm.ova` file, this file must be imported into VirtualBox. There are many online tutorials ²¹ on how to accomplish this and it primarily consists of two steps (see also Assignment 0): (1) Choose the image to import (navigate to the downloaded file) and (2) check/modify the settings of the virtual machine (typically no modifications are necessary, but sometimes, for example, you may have to disable the USB port). Once you click on the **Import** button, VirtualBox imports the image and sets up the virtual machine that can be used for this assignment. VirtualBox then shows a new VM on the left-hand side, which can be started/booted with a double-click.

For this VM, there exist two users: (1) `dbtutorial` and (2) `root`. If you are not familiar with the concept of a root user (or superuser), please check out the corresponding article ²² on Wikipedia. In essence, the root user has **all** privileges whereas the `dbtutorial` user has not. By default, we will work with the `dbtutorial` user, but we will temporarily switch to the root user if we need additional privileges. The password for both users is the same: `dbpwd1`

Once you logged into the VM and before you start working on the actual assignment, try to familiarize yourself a bit with Debian Linux (unless you already have experience using Linux systems and/or solved Assignment 0). For example, open a terminal (aka Linux **shell** or **command-line tool**) and try some basic commands (some links can be found in the supplementary material, cf. Section 4). In particular, we will use the `cypher-shell` command-line tool (aka Neo4j **terminal**), which provides the most basic way to use Neo4j.

Remark: Note that you may not be able to use the typical CTRL+C and CTRL+V sequence to copy and paste between your regular system and the VM as the clipboard is not shared by default ²³.

2.3 Neo4j Configuration

After the installation, the first step is to validate that the Neo4j daemon (or service) is running on your system. A **daemon** is a process that runs in the background. As for MongoDB, the Neo4j daemon manages the data and the requests ²⁴. This is most likely **not** the case and it is part of this assignment to (re-)start the Neo4j daemon named `neo4j` as shown in Listing 3.

Remark: The `dbtutorial@database-tutorial:~$` (as well as `root@database-tutorial:~#`) at the beginning should already be displayed in your terminal and is not part of the command itself (which is `su -`). Do not get confused by the fact that nothing is shown when you type in the password (not even asterisks `“*”`, which are often used for passwords). This is the default behavior of Linux and you will be asked again if the password is incorrect.

In Listing 3, we first confirm that the Neo4j daemon is **not** running (line 1) using the `systemctl` command. This is only a passive command, hence we can execute this command with the `dbtutorial` user. We observe that the state of our Neo4j daemon is **inactive (dead)**, i.e., it exists but is not running. Next, we need to execute an active command that **modifies** the state of our operating system by starting the Neo4j daemon. To this end, we must first switch to the root user (line 6) as the `dbtutorial` user does not have enough privileges to perform this

¹⁹VM image for this assignment: <https://drive.google.com/file/d/1s6bp8TyzFfHL08EzAI1P4Qm6nnTG7mFA/view?usp=sharing>

²⁰Debian Linux: <https://en.wikipedia.org/wiki/Debian>

²¹Virtual machine import: https://docs.oracle.com/cd/E26217_01/E26796/html/qs-import-vm.html

²²The root user: <https://en.wikipedia.org/wiki/Superuser>

²³Shared clipboards in VirtualBox: <https://www.youtube.com/watch?v=fqrJ7qlhJu0>

²⁴Neo4j daemon: <https://neo4j.com/docs/operations-manual/current/installation/linux/systemd/>

action. Then, we use the `systemctl` command to (re-)start the Neo4j daemon (line 8). Line 9 then switches back to the `dbtutorial` user (as we do not want to stay in root mode).

Afterwards, we confirm that the Neo4j daemon is running (cf. Listing 4).

On Windows systems, one way to see whether the Neo4j daemon is running is to look for a process called `neo4j` in the Task Manager ²⁵.

Listing 3: **terminal** – (Re-)Start the Neo4j daemon.

```
1 dbtutorial@database-tutorial:~$ systemctl status neo4j
2 ● neo4j.service - Neo4j Graph Database
3   Loaded: loaded (/lib/systemd/system/neo4j.service; disabled; vendor ...)
4   Active: inactive (dead)
5 dbtutorial@database-tutorial:~$ su -
6 Password:
7 root@database-tutorial:~# systemctl restart neo4j
8 root@database-tutorial:~# exit
```

Listing 4: **terminal** – Confirm that the Neo4j daemon is running.

```
1 dbtutorial@database-tutorial:~$ systemctl status neo4j
2 ● neo4j.service - Neo4j Graph Database
3   Loaded: loaded (/lib/systemd/system/neo4j.service; disabled; vendor ...)
4   Active: active (running) since Tue 2023-05-02 13:12:12 CEST; 5s ago
5   Main PID: 57756 (java)
6   Memory: 305.7M
7   CPU: 5.330s
8   CGroup: /system.slice/neo4j.service
9           └─Ä57756 /usr/bin/java -cp /var/lib/neo4j/plugins: ...
```

In this case, the `systemctl` command shows a lot of status information about the Neo4j daemon, e.g., the starting time, a process ID (PID), used memory, and the likes. This information may be different on your system, and the most important information is that the Neo4j daemon is **active (running)**. Once the Neo4j daemon is running, we can connect to our local database using the `cypher-shell` as shown in Listing 5.

Listing 5: **terminal** – Connect to the local Neo4j server.

```
1 dbtutorial@database-tutorial:~$ cypher-shell
```

On the first connection, the Neo4j server will ask you for credentials to log into the system. The default credentials are user “`neo4j`” with password “`neo4j`”. After the first login, you will be asked to set a new password, which is to be used for subsequent logins.

The connection can also be established using the web-based graphical user interface (GUI) by opening a browser (e.g., Firefox in the VM) and connecting to address **localhost:7474** (without `www`).

For both methods, we can start to interact with the database once the connection is established. As highlighted in red in Figure 3, the web-based GUI provides a terminal on top where queries and commands can be executed.

Similar to PostgreSQL and MongoDB, Neo4j has a default database named `neo4j`. The name that is displayed on the left-hand side of the `cypher-shell` (command-line) tool as well as in the web-based graphical user interface is the name of the database we are currently operating on (e.g., `neo4j@neo4j>` in the `cypher-shell` and `neo4j$` in the graphical user interface both indicate that we operate on the `neo4j` database). To list all databases that are available on our Neo4j server, we can use the command shown in Listing 6

²⁵Windows’ Task Manager: [https://en.wikipedia.org/wiki/Task_Manager_\(Windows\)](https://en.wikipedia.org/wiki/Task_Manager_(Windows))

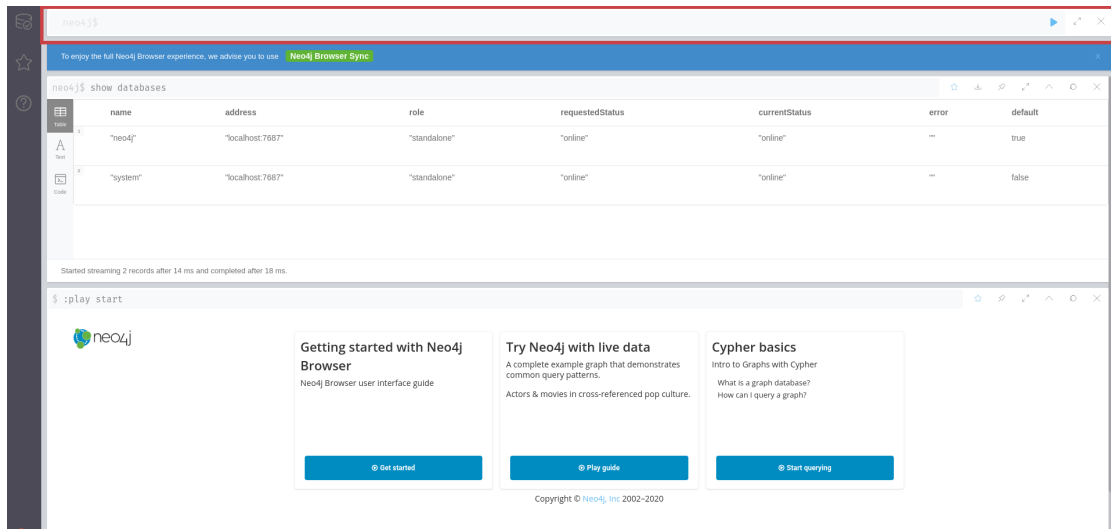


Figure 3: Terminal in the web-based graphical user interface of Neo4j (red).

Listing 6: **cypher-shell** – Show all databases on our server (mind the trailing semi-colon).

```
1 neo4j@neo4j> SHOW DATABASES;
```

In order to switch to a specific database, e.g., the neo4j database, we can use the command in Listing 7.

Listing 7: **cypher-shell** – Switch to database neo4j (mind the colon at the beginning).

```
1 neo4j@neo4j> :USE neo4j
```

In Neo4j, it is required to create a database before we can fill it with data (contrarily to MongoDB, which creates the databases on the fly). Unfortunately, the Community Server edition does not support the creation a database ²⁶. **Therefore, please use the default database neo4j as database for this assignment.**

However, for the sake of completeness, we still discuss how to create a new database (in case you work with the Professional edition eventually). We can use the command shown in Listing 8 to create a new database named assignment3 and switch onto it.

Listing 8: **cypher-shell** – Create and switch onto database assignment3.

```
1 neo4j@neo4j> CREATE DATABASE assignment3;
2 neo4j@neo4j> :USE assignment3
```

In order to create a new database, you need to have administrator/root privileges, otherwise you may see the error message depicted in Figure 4 (for example, if you try to create a new database using the web-based graphical user interface). In this case, please make sure that the Neo4j user has the privileges that are required to create a new database (i.e., write permissions).

²⁶<https://stackoverflow.com/a/60431349>

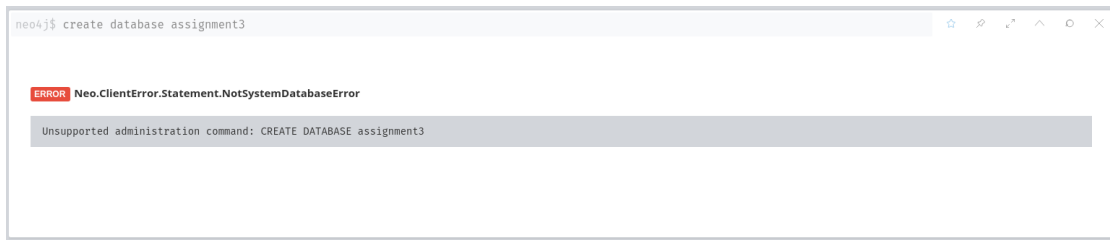


Figure 4: Error message in the graphical user interface if you do not have sufficient privileges.

2.4 Data Initialization

A fresh Neo4j installation only contains an empty neo4j database. We can verify this by executing the command shown in Listing 9, which shows ten nodes and the corresponding edges in our graph database.

Remark: We intentionally limit the number of nodes to ten because otherwise the query may take quite some time to finish in the VM. Consequently, you may want to consider limiting the number of nodes for most of your queries.

Listing 9: **cypher-shell** – Show ten nodes and the corresponding edges in our database.

```
1 neo4j@neo4j> MATCH (n)-[r]->(m) RETURN n,r,m LIMIT 10;
```

The next step is to populate the database with some data. In the course of this assignment, we will use data of a publicly available movies dataset, which is provided as database dump (for Neo4j version 5.18.0 or higher). Although there are multiple ways to import data into Neo4j (e.g., via CSV file ²⁷ or from a relational model ²⁸), we provide you with a so-called **database dump** ²⁹ of a graph that represents popular movies and actors. A dump (of a database) is a snapshot of the data and can be directly imported into the corresponding database system (in our case Neo4j). Database dumps are commonly used to create a backup of a database.

First, we must download the data from our Nextcloud ³⁰. Since the data is a dump of a database, we can simply import it directly into our neo4j database. For this (and other administrative) purpose(s), Neo4j provides a command-line tool named neo4j-admin ³¹, which can be used as shown in Listing 11 (note that neo4j-admin is called from the system command-line tool). First, however, we must stop the database daemon before we can import the data as depicted in Listing 10. To this end, we switch to the root user as our dbtutorial user does not have enough privileges (line 1). Then, we proceed by stopping the Neo4j daemon (line 2) and checking whether the Neo4j daemon has been stopped indeed (line 3).

Remark: The commands in Listing 11 are to be executed directly after the commands in Listing 10. The two listings are separated to highlight the two separate steps, i.e., (i) stopping the database and (ii) importing the data.

The import using the neo4j-admin tool must be executed by a user that has enough privileges to import data into the database. In case of Neo4j, a dedicated user named neo4j has been created during installation that owes these privileges, and we can switch to this user as shown in line 1 of Listing 11. Line 2 shows the actual import of the database dump: We tell the

²⁷CSV import: <https://neo4j.com/docs/getting-started/data-import/csv-import/>

²⁸Rel. import: <https://neo4j.com/docs/getting-started/data-import/relational-to-graph-import/>

²⁹Database dump: https://en.wikipedia.org/wiki/Database_dump

³⁰Data for this assignment: <https://kitten.cosy.sbg.ac.at/index.php/s/DesSFRwZ83FAZdP/download>

³¹The neo4j-admin tool: <https://neo4j.com/docs/operations-manual/current/tools/neo4j-admin/>

neo4j-admin tool to load a database and to overwrite any existing database with the same name (`--overwrite-destination true`). Moreover, we tell the neo4j-admin tool to read the database dump from standard input (`--from-stdin`), which means that we must provide the file at the end of the command using the `<` pipeline. This pipeline operator of the Linux terminal directly redirects the content of the right-hand file to the command on the left-hand side of the operator, i.e., `neo4j-admin ... < movies-50.dump` tells the neo4j-admin tool to directly read to content of the file `movies-50.dump`. Right before this operator, we finalize the arguments to the neo4j-admin tool by providing the name of the database the given file should be imported to. Subsequently, we log back into our root user by entering the `exit` command (line 3), restart the Neo4j daemon once again (line 4), and verify that the Neo4j daemon is running again (line 5). Finally, we log back into our `dbtutorial` user by entering the `exit` command a second time (line 14).

Listing 10: **terminal** – Stop the Neo4j daemon.

```
1 dbtutorial@database-tutorial:~# su -
2 root@database-tutorial:~# systemctl stop neo4j
3 root@database-tutorial:~# systemctl status neo4j
4 • neo4j.service - Neo4j Graph Database
5   Loaded: loaded (/lib/systemd/system/neo4j.service; disabled; vendor ...)
6   Active: inactive (dead)
```

Listing 11: **terminal** – Import a database dump into Neo4j.

```
1 root@database-tutorial:~# su - neo4j
2 neo4j@database-tutorial:~# neo4j-admin database load --overwrite-destination true \
3 --from-stdin neo4j < movies-50.dump
4 neo4j@database-tutorial:~# exit
5 root@database-tutorial:~# systemctl restart neo4j
6 root@database-tutorial:~# systemctl status neo4j
7 • neo4j.service - Neo4j Graph Database
8   Loaded: loaded (/lib/systemd/system/neo4j.service; disabled; vendor ...)
9   Active: active (running) since Tue 2023-05-02 13:12:12 CEST; 5s ago
10  Main PID: 57756 (java)
11  Memory: 305.7M
12  CPU: 5.330s
13  CGroup: /system.slice/neo4j.service
14          ÅÅ57756 /usr/bin/java -cp /var/lib/neo4j/plugins: ...
15 root@database-tutorial:~# exit
```

Remark: The `--from-stdin` option of `neo4j-admin` also works with full paths, for example, `neo4j-admin ... --from-stdin neo4j < /home/dbtutorial/movies-50.dump`. Therefore, please ensure that you are using the correct path to the `movies-50.dump` file. **Hint:** If you unzipped the `assignment3-data.zip` as user `dbtutorial`, the `movies-50.dump` file will be located in the home directory of the `dbtutorial` user, i.e., `/home/dbtutorial/movies-50.dump`.

This imports the `movies` database dump into the `neo4j` database, and we can verify this by reconnecting to our database (using the `cypher-shell` or the graphical user interface) and executing the command shown in Listing 9 again. This time, Neo4j should report ten nodes and the corresponding edges in either a table format (cf. Figure 5) or – if you are using the graphical user interface – as a graph visualization (cf. Figure 6).

Word of Caution: Please use the `--overwrite-destination` option with caution in a production environment! As the name suggests, this option **overwrites** an existing database (in our case the `neo4j` database) and all data in this specific database is lost.

Remark: Neo4j may report different nodes and edges since we do not explicitly impose an order in our `MATCH` query.

```
neo4j@neo4j> MATCH (n)-[r]->(m) RETURN n,r,m LIMIT 10;
```

n	r	m
{:Person (name: "Keanu Reeves", born: 1964)}	{:ACTED_IN (roles: ["Neo"])}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Carrie-Anne Moss", born: 1967)}	{:ACTED_IN (roles: ["Trinity"])}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Laurence Fishburne", born: 1961)}	{:ACTED_IN (roles: ["Morpheus"])}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Hugo Weaving", born: 1960)}	{:ACTED_IN (roles: ["Agent Smith"])}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Lilly Wachowski", born: 1967)}	{:DIRECTED}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Lana Wachowski", born: 1965)}	{:DIRECTED}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Joel Silver", born: 1952)}	{:PRODUCED}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Emil Eifrem", born: 1978)}	{:ACTED_IN (roles: ["Emil"])}	{:Movie (tagline: "Welcome to the Real World", title: "The Matrix", released: 1999)}
{:Person (name: "Keanu Reeves", born: 1964)}	{:ACTED_IN (roles: ["Neo"])}	{:Movie (tagline: "Free your mind", title: "The Matrix Reloaded", released: 2003)}
{:Person (name: "Carrie-Anne Moss", born: 1967)}	{:ACTED_IN (roles: ["Trinity"])}	{:Movie (tagline: "Free your mind", title: "The Matrix Reloaded", released: 2003)}

10 rows
ready to start consuming query after 3 ms, results consumed after another 15 ms

Figure 5: Ten nodes and the corresponding edges in table format.

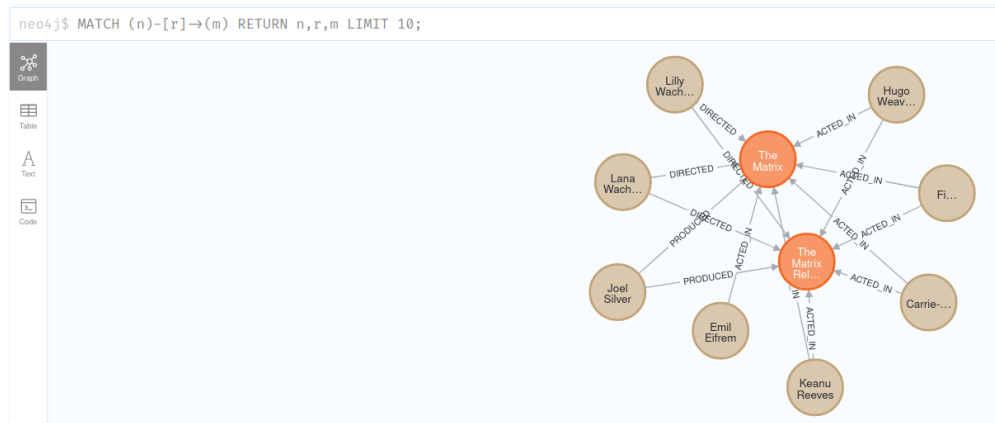


Figure 6: Ten nodes and the corresponding edges as graph visualization.

2.5 Start Using Neo4j

Remark: Questions F1 and F2 in the questionnaire may also help to get used to Neo4j.

After importing the database dump, try to further familiarize yourself with the `cypher-shell` command-line tool (or the corresponding web interface). Neo4j does not support SQL but uses the Cypher query language (CQL), which is a declarative query language that provides a visual and intuitive way to match graph patterns. In the course of this assignment, we will use CQL to find patterns in our movies graph database and the Neo4j documentation provides a detailed explanation of CQL ³². In the following, we briefly cover some basics of the Cypher query language and provide **two (!)** queries, **Q1–Q2**, which can be executed out of the box by entering them into the `cypher-shell` command-line tool (or the corresponding web interface) once the data has been imported.

After connecting to the neo4j database, we can directly use CQL statements to query the data. The query shown in Listing 12, for example, selects **all** pairs of nodes that have some relationship (i.e., edge):

Listing 12: **cypher-shell** – Show all nodes and the corresponding edges in our database.

```
1 neo4j@neo4j> MATCH (n)-[r]->(m) RETURN n,r,m;
```

The `MATCH` keyword is used to search for a node, a relationship, or a pattern in the database. In this specific case, we search for a pattern that consists of two nodes, (`n`) and (`m`), and a relationship (i.e., edge), `-[r]->`, which connects the two nodes. `n` and `m` are variable names that can be used to refer to a particular node (e.g., in the `RETURN` statement). Analogously, `r` is the variable name of the corresponding relationship. The arrow denotes the direction of the

³²Cypher query language: <https://neo4j.com/docs/getting-started/cypher-intro/>

relationship, i.e., from (n) to (m). As a result, the system will report (i.e., RETURN) all nodes and edges in the graph that match this specific pattern.

In Listing 9, we have already seen that we can use the LIMIT keyword to restrict the number of nodes that are reported (with the corresponding edges connecting them).

In this assignment, you are only given two queries in the Cypher query language, namely Q1–Q2. It is part of this assignment to study the Cypher query language and come up with two additional CQL queries on your own.

For query Q3, please study the **Updating the Data** section of the documentation³³ and either create a new relationship between two (or more) nodes or update an existing relationship between two (or more) nodes.

For query Q4, please study the **Getting the correct results** section of the documentation³⁴ and apply a filter on a pattern.

It is up to you what you create/update and which filter you apply. However, the queries are supposed to be **meaningful**, i.e., meaningful in the sense that the query has some effect (reports a result or creates/updates an edge). Just try to explore the data and the Cypher query language. Interested students can also study and use more advanced CQL features (e.g., graph algorithms) to formulate additional queries Q5, Q6, and so on to obtain Bonus points. For reference, however, creation/update (Q3) and filtering (Q4) are the minimum requirements.

Before we continue with Q1 and Q2, we study another CQL query in detail (cf. Listing 13):

Listing 13: `cypher-shell` – Find all movies of person (actor) “Tom Hanks”.

```
1 neo4j@neo4j> MATCH (p:Person {name: "Tom Hanks"})
2               -[r:ACTED_IN]->(m:Movie) RETURN p,r,m;
```

Similar to the previous CQL query, we define the pattern to have two nodes that are connected via some relationship. However, this time we specify the following additional details: (i) The type of the node/edge and (ii) a property for the left-hand node. (i) After the variable name of a node/edge, we can specify the type of the node/edge (separated by a colon). In our example, we search for nodes of type Person with a connection to nodes of type Movie. The relationship between the nodes must be of type ACTED_IN. (ii) { name: "Tom Hanks" } is a property of a node that must be satisfied to match the pattern. Consequently, we only find movies of the person “Tom Hanks”. For more details, we refer to the Neo4j documentation on Querying³⁵.

Now that we have learned about the basics of CQL, we provide the two queries Q1 and Q2 ((cf. Listings 14– 15)) that can be executed out of the box by entering them into the cypher-shell (or the web interface; one after another). Please use SHIFT+ENTER for multi-line queries in the web-based GUI, and the blue Play button on the right-hand side to execute it.

Listing 14: `cypher-shell` – Query Q1.

```
1 neo4j@neo4j> MATCH (p1:Person {name: "Tom Hanks"})
2               -[a1:ACTED_IN]->(m:Movie)<-[a2:ACTED_IN]-
3               (p2:Person) RETURN p1,m,p2;
```

Listing 15: `cypher-shell` – Query Q2.

```
1 neo4j@neo4j> MATCH v=shortestPath(
2               (p1:Person {name: "Christian Bale"})
3               -[*]-(p2:Person {name: "Halle Berry"})
4               ) RETURN v;
```

³³Updating the data: <https://neo4j.com/docs/getting-started/cypher-intro/updating/>

³⁴Getting the correct results: <https://neo4j.com/docs/getting-started/cypher-intro/results/>

³⁵<https://neo4j.com/docs/getting-started/cypher-intro/>

In query **Q1**, we specify a pattern with three nodes and two `ACTED_IN` relationships (in opposite directions). Semantically, this identifies all persons that acted in a movie with “Tom Hanks”.

Query **Q2** executes an operation that is common for graphs, namely a **shortest path** query³⁶. Almost everybody has implicitly used a shortest path in a navigation system to navigate from one physical location to another one. A shortest path algorithm in a navigation system identifies the path that has the smallest costs to reach the other location (with respect to some cost model that includes, for example, the path length and/or the current traffic). However, shortest paths are a general concept in graphs and can also be used to identify the shortest path of relationships between persons in social networks. In our example, query **Q2** finds the degree of separation between the two persons (actors) “Christian Bale” and “Halle Berry”, i.e., the connection between “Christian Bale” and “Halle Berry” in hops through other persons and movies.

This concludes the two given queries **Q1** and **Q2**. Recall that it is part of this assignment to come up with two additional queries **Q3** (create a new edge or update an existing edge)³⁷ and **Q4** (apply a filter in a CQL query)³⁸ on your own by studying the Neo4j documentation and the data in the web-based graphical user interface of Neo4j.

2.6 Access the Data Using Python3

Similar to most database systems, Neo4j is typically accessed by an application. Therefore, the fourth part of this assignment is to write a small Python3 application. We recommend to use the `neo4j-driver` module for Python3 to (a) establish a connection to your local database, (b) execute the queries and retrieve the results, and (c) close the connection to your local database. Your application should execute the queries **Q1–Q2** (given in Section 2.5) and **Q3–Q4** (introduced by your team), one after another, and print their results to the command line.

Remark: If you cannot execute all queries (for whatever reason), please contact the instructor *before* the submission. We will find a reasonable solution together.

Template Code There are many tutorials regarding the installation³⁹ and the usage of the `neo4j-driver`⁴⁰ module to access the Neo4j database. Nonetheless, we provide a minimum template code in Python3 that can be used as a starting point. Unfortunately, PDF viewers tend to represent formatted code differently that cause troubles when copying the code from the PDF into another file. Therefore, we do **not** include the template code in this assignment description but as a separate `.py` file that is available via the course website⁴¹.

Hint: For your Python3 code, you may also want to consider the `execute_write` function⁴², in particular if you modify the database. Please study the corresponding documentation and use the respective functions properly.

2.7 Questionnaire

The questionnaire contains questions about this assignment. These questions are potentially discussed during the after-assignment meetings. The questionnaire can be found in a separate text file called `assignment3-questionnaire.txt`.

³⁶Shortest path problem: https://en.wikipedia.org/wiki/Shortest_path_problem

³⁷Updating the data: <https://neo4j.com/docs/getting-started/cypher-intro/updates/>

³⁸Getting the correct results: <https://neo4j.com/docs/getting-started/cypher-intro/results/>

³⁹<https://neo4j.com/docs/api/python-driver/current/> and <https://pypi.org/project/neo4j-driver/>

⁴⁰neo4j-driver tutorial: <https://neo4j.com/docs/getting-started/languages-guides/neo4j-python/>

⁴¹Template code: <https://dbresearch.uni-salzburg.at/teaching/2024ss/dim/assignment3-template.py>

⁴²The `execute_write` function: https://neo4j.com/docs/api/python-driver/current/api.html#neo4j.Session.execute_write

3 Submission

Please submit a single compressed archive (e.g., .zip or .tar.gz) that contains exactly two files: (a) Your Python3 code and (b) your answers to the questionnaire.

Code

Remark: Please consider removing the database credentials (i.e., username and password) before you submit your code (in case you used them).

Please submit a single Python3 file (.py) that contains the full code for this assignment, i.e., the connection to the database and the execution of the four queries. The code **must** print the results of **all** four queries **Q1–Q4** (one after another) when executed **as submitted** (**Hint:** Set up another virtual machine and try whether it runs). We will **not** debug your code, for example, change some variable to make it work. Therefore, please double-check that your Python3 code works as expected and that **all** four queries are executed (recall that queries **Q3** and **Q4** need to be formulated by yourself as described in Section 2.5).

Questionnaire

Remark: The recommended formats are .txt and .pdf.

You can answer the questions directly in the text file assignment3-questionnaire.txt. If you prefer to use a different application to answer the questions (e.g., Microsoft Word and the likes), you are welcome to do so. In any case, the submitted file must be in one of the following formats: .txt, .pdf, .odt, .doc, or .docx.

4 Supplementary Material

This section provides a list of pointers to material that may be helpful to solve the assignment.

- Neo4j: <https://neo4j.com/>
- The full Neo4j documentation: <https://neo4j.com/docs/>
- Neo4j “Getting Started”: <https://neo4j.com/docs/getting-started/>
- Neo4j resources regarding the installation on
 - Linux systems:
<https://neo4j.com/docs/operations-manual/current/installation/linux/>
 - MacOS systems:
<https://neo4j.com/docs/operations-manual/current/installation/osx/>
 - Windows systems:
<https://neo4j.com/docs/operations-manual/current/installation/windows/>
- Other resources regarding the installation and the usage of Neo4j:
 - Two of the many Neo4j “Getting Started” guides:
 1. Windows: <https://www.youtube.com/watch?v=hWDq0GSi7Tc>
 2. Linux/MacOS: <https://www.youtube.com/watch?v=M2JSvN1Afw8> (without the custom configuration at the end)
 - Comparing SQL with Cypher: <https://neo4j.com/developer/cypher/guide-sql-to-cypher/>
- Neo4j data model concepts: <https://neo4j.com/docs/getting-started/current/graphdb-concepts/>
- Reference for the cypher-shell command-line tool and its web-based counterpart (the graphical user interface):
<https://neo4j.com/docs/getting-started/cypher-intro/> and <https://neo4j.com/docs/cypher-manual/current/introduction/>

- Python Modules: <https://docs.python.org/3/installing/index.html>
- The neo4j-driver module: <https://neo4j.com/docs/api/python-driver/current/>
- Installation of the neo4j-driver module for Python:
 - Official website: <https://neo4j.com/docs/api/python-driver/current/#installation>
 - The Python Package Index: <https://pypi.org/project/neo4j-driver/>
- The neo4j-driver tutorial: <https://neo4j.com/docs/getting-started/languages-guides/neo4j-python/>
- An introduction to all Neo4j drivers (including the neo4j-driver Python module): <https://www.youtube.com/watch?v=JwgxkGY-36Q>
- One of the many introductions to the Linux terminal: <https://www.digitalocean.com/community/tutorials/an-introduction-to-the-linux-terminal>

5 Grading

For the sake of transparency, this section provides more details on the grading of this assignment, i.e., which part contributes how many points to the total number of 20 points.

Code The code contributes at most 10 points and is evaluated based on the following criteria (if the code is executed as submitted; disregarding the credentials):

Max. Points	Criterion
1.5	<i>Q1 is executed and the correct result is printed to the command line.</i>
1.5	<i>Q2 is executed and the correct result is printed to the command line.</i>
1.5	<i>Q3 is executed and the correct result is printed to the command line.</i>
1.5	<i>Q4 is executed and the correct result is printed to the command line.</i>
2+2	<i>Answer two questions on your code in the after-assignment meeting correctly.</i>
10	

Questionnaire The questionnaire contributes at most 10 points and is evaluated based on the following criteria (taking the discussion in the after-assignment into account):

Max. Points	Criterion
1.5	<i>Correctness of answer A1.</i>
1.5	<i>Correctness of answer A2.</i>
1.5	<i>Correctness of answer A3.</i>
1.5	<i>Correctness of answer A4.</i>
2+2	<i>Answer two additional questions in the after-assignment correctly.</i>
10	