**Aufgabe 1**

**Installation java**

su –

echo "deb http://ppa.launchpad.net/webupd8team/java/ubuntu xenial main" | tee /etc/apt/sources.list.d/webupd8team-java.list

echo "deb-src http://ppa.launchpad.net/webupd8team/java/ubuntu xenial main" | tee -a /etc/apt/sources.list.d/webupd8team-java.list

apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys EEA14886

apt-get update

apt-get install oracle-java8-installer

**Installation neo4j**

wget -O – <https://debian.neo4j.org/neotechnology.gpg.key> | sudo apt-key add-

echo ‚deb [http://debian.neo4j.org/repo stable/](http://debian.neo4j.org/repo%20stable/)‘ >/tmp/neo4j.list

sudo mv /tmp/neo4j.list /etc/apt/sources.list.d

sudo apt-get update

sudo apt-get install neo4j

sudo service neo4j restart

**Connect to neo4j**

Username: neo4j

Pw: crdnvr9f

**Remote-Verbindung**

sudo gedit /etc/neo4j/neo4j.conf

Uncomment:

dbms.connectors.default\_listen\_address=0.0.0.0

dbms.connector.http.listen\_address=:7474

dbms.conncetor.https.listen\_address=:7473

ifconfig -> vm ip finden -> browser im rechner mit dieser ip+port verbinden

**Aufgabe 2**

**Input Data – Ohne Vorverarbeitung bzw. Neo4j übernimmt Vorverarbeitung**

using periodic commit 500

load csv from "file:///wahlverwandschaften\_clean.txt" as row fieldterminator "\*"

with row

unwind row as text

with text

where text is not null

with reduce(t=tolower(text), delim in [",",".","!","?",'"',":",";","'","-","\_","/","%","&","+","\*","~","<",">","|","`","´"] | replace(t,delim,"")) as normalized

with [w in split(normalized," ") | trim(w)] as wahle

unwind range(0,size(wahle)-2) as idx

MERGE (w1:Wahlverwandschaften {name:wahle[idx]})

MERGE (w2:Wahlverwandschaften {name:wahle[idx+1]})

MERGE (w1)-[r:WahlKanten]->(w2)

ON CREATE SET r.count = 1 ON MATCH SET r.count = r.count +1

oder

**Input Data – Mit Vorverarbeitung durch beigefügtes Javaprogramm**

using periodic commit 500

load csv from "file:///wahlverwandschaften\_clean.txt" as row fieldterminator "\*"

with row

unwind row as text

with text

where text is not null

with split(tolower(text)," ") as wahle

unwind range(0,size(wahle)-2) as idx

MERGE (w1:Wahlverwandschaften {name:wahle[idx]})

MERGE (w2:Wahlverwandschaften {name:wahle[idx+1]})

MERGE (w1)-[r:WahlKanten]->(w2)

ON CREATE SET r.count = 1 ON MATCH SET r.count = r.count +1

**Nützliche und/oder Testbefehle**

**Find Edge**

MATCH (w1:Wahlverwandschaften)-[r:WahlKanten]->(w2:Wahlverwandschaften)

WHERE w1.name="ottilie" and w2.name="von"

RETURN r.count

**Delete all**

MATCH (n)

DETACH DELETE n

**Aufgabe 3**

**3.a - Max. Knoten & Kanten**

MATCH(w:Wahlverwandschaften)

RETURN count(w)

* Knoten: 10864

MATCH ()-[r:WahlKanten]->()

RETURN count(r)

* Kanten: 52944

**3.b - Häufigstes Diagramm**

MATCH (w1:Wahlverwandschaften)-[r:WahlKanten]->(w2:Wahlverwandschaften)

RETURN r.count, w1, w2

ORDER BY r.count desc

LIMIT 1

* Diagramm: **in der** -> 152

**3.c – Zusammenhängikeit des Graphen – Wahlverwandschaften**

MATCH (w:Wahlverwandschaften) WHERE w.name="in"

CALL apoc.path.subgraphNodes(w, {}) YIELD node

RETURN node

* 10864/10864

**3.d - Schnellster Pfad von ‚ottilie‘ zu ‚goethe‘**

MATCH (w1:Wahlverwandschaften { name: 'ottilie' }),(w2:Wahlverwandschaften { name: 'goethe' }), p = shortestPath((w1)-[\*]-(w2))

RETURN p

* ottilie – von - goethe

**3.e – Min. 5 Wörter von ‚ottilie‘ zu ‚goethe‘**

MATCH (w1:Wahlverwandschaften { name: 'ottilie' }),(w2:Wahlverwandschaften { name: 'goethe' }), p = shortestPath((w1)-[\*]-(w2))

WHERE length(p)> 5

RETURN p

* ottilie - ohne - jedoch - spätlinge - deren - von - goethe