

Insert Assignment Title Here

02807 Computational Tools for Big Data

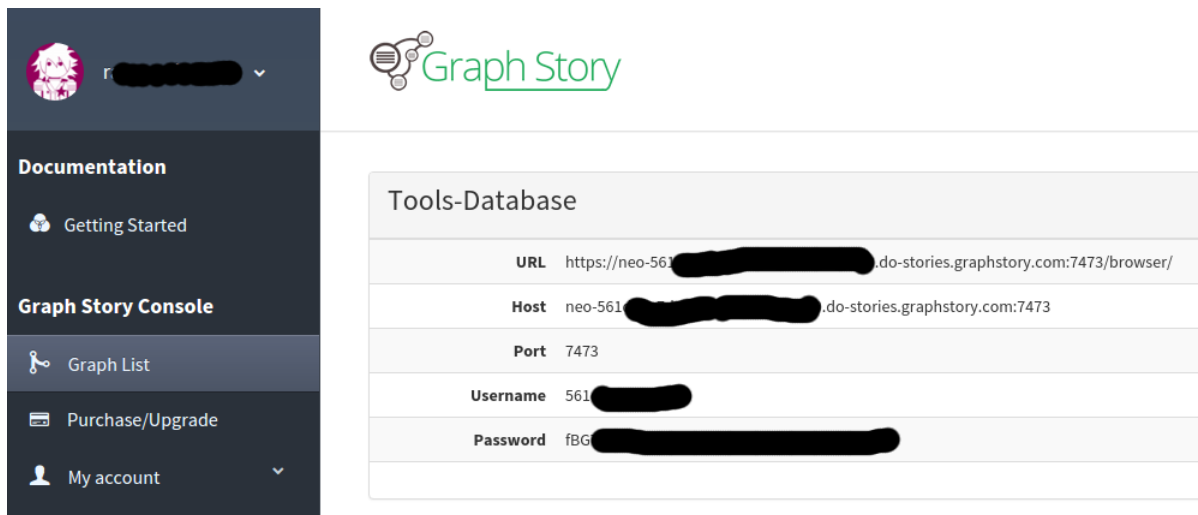
Anonymous authors

Insert hand in date here

1 Exercise 6

1.1 Exercise 6.1

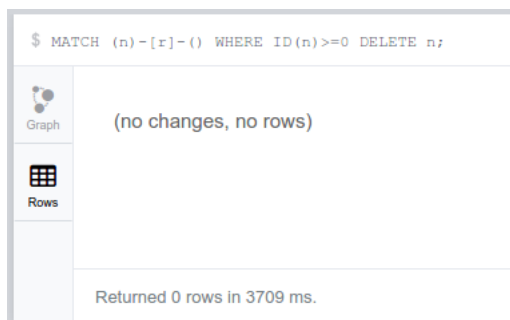
Here you can see we have created a Graph Story account and generated a database:



The screenshot shows the Graph Story web interface. On the left is a dark sidebar with a user profile at the top, followed by 'Documentation' (with a 'Getting Started' link), 'Graph Story Console', and a 'Graph List' link. Below that are 'Purchase/Upgrade' and 'My account' links. The main area on the right is titled 'Tools-Database' and contains a table with the following data:

Tools-Database	
URL	https://neo-561[redacted].do-stories.graphstory.com:7473/browser/
Host	neo-561[redacted].do-stories.graphstory.com:7473
Port	7473
Username	561[redacted]
Password	fBG[redacted]

Then we removed all data from the database:



The screenshot shows a Cypher query execution interface. The query entered is: `$ MATCH (n)-[r]-() WHERE ID(n)>=0 DELETE n;`. The interface has a 'Graph' view (selected) and a 'Rows' view. The result area shows '(no changes, no rows)'. At the bottom, it states 'Returned 0 rows in 3709 ms.'

Lastly, we loaded all the data into the database, using the given code. I've chosen to only include the output of the last line of code, which should be enough proof that we have done them all.

```
$ LOAD CSV WITH HEADERS FROM "http://data.neo4j.com/northwind/order-details.csv" AS ro
```

Graph

Rows

Set 12930 properties, created 2155 relationships, statement executed in 2109 ms.

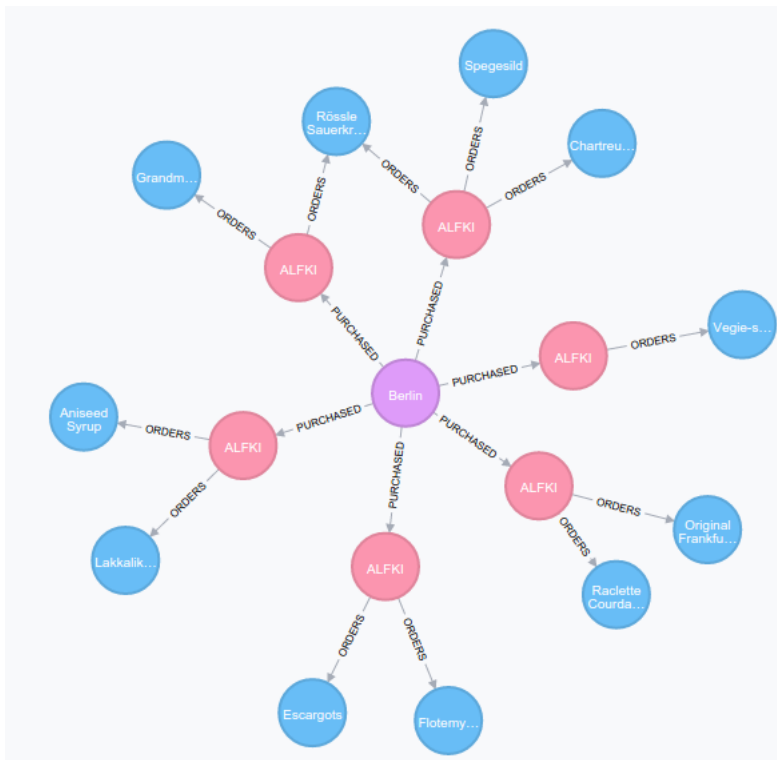
Returned 0 rows.

1.2 Exercise 6.2

Using the following query, we are able to find all orders for the customer with customerID 'ALFKI', including the products in those orders.

```
MATCH (customer {customerID: 'ALFKI'})--(orders) RETURN orders
```

This returns the following graph from the database:



Here, the purple circle is the customer, the red circles are orders, and the blue circles are the products.

1.3 Exercise 6.3

Using the following query, we are able to find all orders for the customer with customerID 'ALFKI', including only the orders with at least 2 products.

```
MATCH (customer {customerID: 'ALFKI'})--(order)
MATCH p=(order)--(X)
WITH order as order, count(p) as paths
WHERE paths > 2
RETURN order
```

This returns the following graph from the database:

