
ЗАДАЧА 7 – ГРАФОВЫЕ БАЗЫ



1. Установить 2 графовые базы из списка

DB-Engines Ranking

1. Предпочтительные - neo4j, nebula, arangodb

2. Предпочтительный язык запросов cypher

2. Создать iрynb ноутбук в котором:

3. Считать данные из источника https://disk.yandex.ru/d/s6wWqd8OI_5lvQ

4. Внести данные из таблицы в графовую БД

5. Построить графовое представление в БД, осуществить несколько запросов на языке запросов к графовой БД

6. Найти взаимосвязи визуально и с помощью алгоритмов (алгоритмы на ваше усмотрение)

7. Написать rest сервис на python к графовой БД в котором на вход поступает ФИО, на выходе graphml или json

8. Результаты представить на гитхаб и в виде кода + небольшой презентации

9. Прислать ссылку на решение и резюме в телеграм @frankshikhaliev

10. Также надо будет заполнить форму

1. Установить 2 графовые базы из списка [DB-Engines Ranking](#)

Выбираем Neo4J ondisk, разработчик языка запросов Cypher, отличная документация <https://neo4j.com/docs/>

MemGraph onmemory поддерживает язык запросов Cypher, отличная документация <https://memgraph.com/docs/>, по заявлению SpaceX в 120 раз быстрее Neo4j, обязательно проверим

Установка Neo4J Windows

- zulu19.32.13-ca-jdk19.0.2-win_x64.msi - JavaSDK
- neo4j-community-5.5.0-windows.zip – распаковать на c:\path\, выполнить cmd: `<NEO4J_HOME>\bin\neo4j windows-service install`, отключим на localhost авторизацию `dbms.security.auth_enabled=false`
- neo4j-graph-data-science-2.3.1.zip [Graph Data Science Library 2.3.1](#) устанавливаем как плагин [разкомментировать строку dbms.security.procedures.allowlist=apoc.coll.*,apoc.load.*,gds.*](#) в конфиге Neo4J

Установка MemGraph WSL2, Docker

- memgraph-2.5.2-docker.tar.gz – docker image, выполнить shell: `docker load -i /path-to/memgraph-<version>-docker.tar.gz`, `docker run -p 7687:7687 -p 7444:7444 -v mg_lib:/var/lib/memgraph`
 - MemgraphLab-2.4.0.exe - полный набор инструментов администрирования/анализа графов
-

2. Создать iрunb ноутбук в котором:

3. Считать данные из источника https://disk.yandex.ru/d/s6wWqd8OI_5lvQ

4. Внести данные из таблицы в графовую БД

- среда разработки Anaconda3-2022.10-Windows-x86_64.exe

- все значения в колонках уникальны, значения ФИО в колонке **full_name_event1** не имеют вхождения в колонку с ФИО **full_name_event2**

- формируем запрос Cypher

```
CREATE (label189:Event {name:189, id_event:189})
```

```
CREATE (label206:Event {name:206, id_event:206})
```

```
CREATE (label445:Event {name:445, id_event:445})
```

```
CREATE (label503:Event {name:503, id_event:503})
```

```
CREATE (label571:Event {name:571, id_event:571})
```

 и тд, записываем в файл *.cypherl

- с использованием python с использованием MemGraph Lab и пакета cypher-shell для Neo4j импортируем в обе БД.

-
5. Построить графовое представление в БД, осуществить несколько запросов на языке запросов к графовой БД
 6. Найти взаимосвязи визуально и с помощью алгоритмов (алгоритмы на ваше усмотрение)
-

7. Написать rest сервис на python к графовой БД в котором на вход поступает ФИО, на выходе graphml или json

- ознакомимся с The Neo4j REST API Documentation v3.5 – Neo4j выполняет запросы Cypher по порту <http://localhost:7474/db/имя> вашей дб/tx в формате json

- request – {

```
"query" : "MATCH (x {name: 'I'})-[r]->(n) RETURN type(r), n.name, n.age",
```

```
"params" : { }
```

```
}
```

- response - {

```
"columns" : [ "type(r)", "n.name", "n.age" ],
```

```
"data" : [ [ "know", "you", null ], [ "know", "him", 25 ] ]
```

```
}
```

- Rest API сервис реализуем на Bottle – компактный Python web-фреймворк(всего один файл bottle.py). Для отладки Rest API используем **HTTPIe 2023.1.2**

Выводы :

- установлены 2 графовые СУБД: Neo4J & MemGraph
- на основе датасета построено графовое представление в Neo4J & MemGraph
- найдены взаимосвязи визуальными средствами и алгоритмически
- разработан Rest API веб-сервис на фреймворке Bottle согласно

The Neo4j REST API Documentation v3.5

- MemGraph работает быстрее Neo4J примерно в 106 раз :)

- Исполнитель Балынин Дмитрий <https://t.me/db2alynin>

СПАСИБО ЗА ВНИМАНИЕ
