**МИНОБРНАУКИ РОССИИ**
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**
**Кафедра МО ЭВМ**

# ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

**по дисциплине «Разработка программного обеспечения информационных систем»**

**Тема: Построение маршрутов – Neo4j**

Студент гр. 5303 _____ Бычков А.Г.

Преподаватель _____ Заславский М.М.

Санкт-Петербург
2018

**ЗАДАНИЕ**

**НА ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ**

Студенты Бычков А.Г.

Группа 5303

Тема проекта: Построение маршрутов – Neo4j

Исходные данные:

Проект должен быть разработан с использованием базы данных Neo4j

Содержание пояснительной записки:

Содержание, Введение, Качественные требования к решению, Сценарии использования, Модель данных, Разработанное приложение, Заключение, Список использованных источников.

Предполагаемый объем пояснительной записки:

Не менее 15 страниц.

Дата выдачи индивидуального домашнего задания: 13.09.2018

Дата сдачи индивидуального домашнего задания: 17.01.2019

Дата защиты индивидуального домашнего задания: 17.01.2019

Студент гр.5303 _____ Бычков А.Г.

Преподаватель _____ Заславский М.М.

**АННОТАЦИЯ**

В ходе выполнения данного индивидуального домашнего задания был реализован веб-сервис на основе СУБД Neo4j, с помощью которого выполняются следующие функции:

- построение маршрута из одной точки на карте в другую;
- просмотр координат выбранной точки на карте.

**SUMMARY**

In the course of this individual homework, a Neo4j-based web service was implemented, with the help of which the following functions are performed:

- building a route from one point on the map to another;
- view the coordinates of the selected point on the map.

# СОДЕРЖАНИЕ

## ВВЕДЕНИЕ

Начало теории графов как математической дисциплины было положено Эйлером в его знаменитом рассуждение о Кенигсбергских мостах. Однако эта статья Эйлера 1736 года была единственной в течение почти ста лет. Интерес к проблемам теории графов возродился около середины прошлого столетия и был сосредоточен главным образом в Англии. Имелось много причин для такого оживления изучения графов. Естественные науки оказали свое влияние на это благодаря исследованиям электрических цепей, моделей кристаллов и структур молекул. Развитие формальной логики привело к изучению бинарных отношений в форме графов.

Целью проекта является разработка приложения, с помощью которого можно построить маршрут из одной точки на карте в другую.

В проекте разработано веб-приложение на основе СУБД Neo4j.

# 1. КАЧЕСТВЕННЫЕ ТРЕБОВАНИЯ К РЕШЕНИЮ

Необходимо разработать веб-приложение, позволяющее построить маршрут из одной точки на карте в другую.

Основные функции:

- Координаты выбранной точки;
- Построение маршрутов.

## 2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

### 2.1. Сценарии использования для задачи импорта, представления, анализа и экспорта данных

1. «Построение маршрутов»
    1) Пользователь располагает маркер на карте и вводит радиус поиска точек дорог;
    2) Система отмечает на карте найденные точки дорог (автомобильные, пешие, водные) место, а также показывает координаты маркера.
    3) Пользователь кликом мыши выбирает стартовую точку и в поле над картой системой заполняется id начальной точки;
    4) Пользователь перемещает маркер в зону назначения на карте и вводит радиус поиска точек дорог;
    5) Система отмечает на карте найденные точки дорог (автомобильные, пешие, водные) место, а также показывает координаты маркера.
    6) Пользователь кликом мыши выбирает конечную точку и в поле над картой системой заполняется id конечной точки;
    7) Система автоматически вычисляет маршрут, в зависимости от выбранного типа маршрута (простейший/кратчайший по алгоритму Дейкстры), строит маршрут и выводит его на карту, отображая информацию о длине пути.
2. «Информация по координате»
    1) Пользователь располагает маркер на карте;
    2) Система отмечает на карте найденные точки дорог (автомобильные, пешие, водные) место, а также показывает координаты маркера.

### 2.2. Вывод

При использовании конечным пользователем преобладают операции чтения, так как основной функцией является поиск точек на карте и отображение маршрутов.

# 3. МОДЕЛЬ ДАННЫХ

Модель данных для нереляционной базы данных.

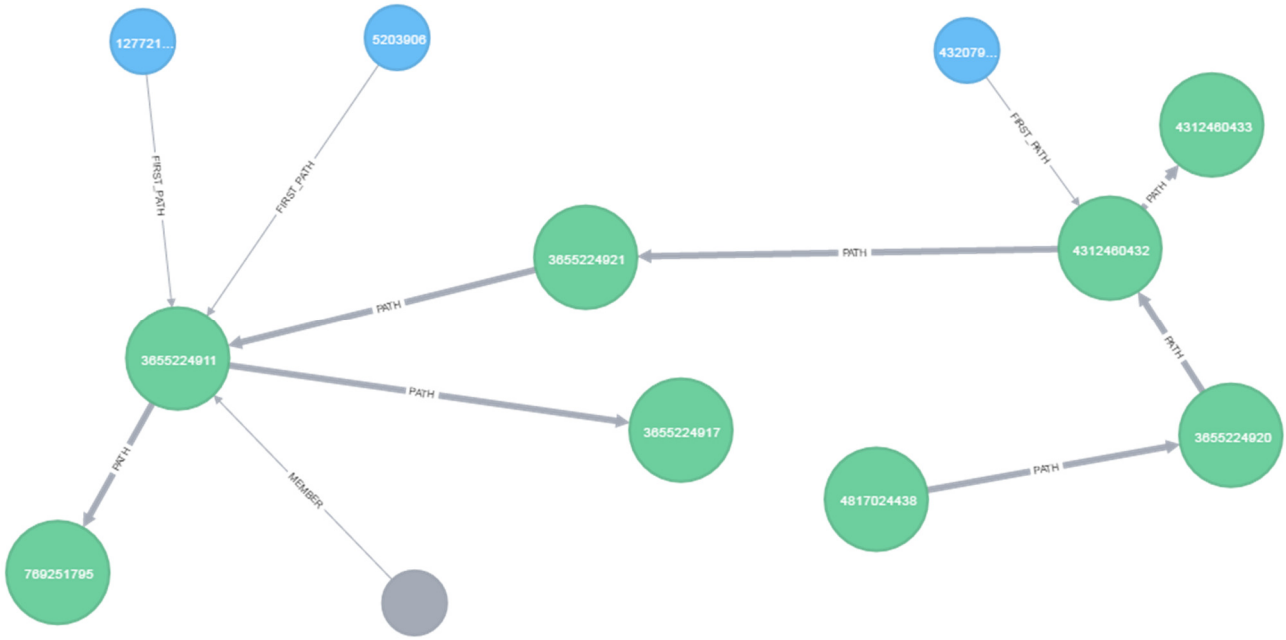


Рисунок 1 – Модель данных нереляционной БД

## 3.1. Описание структуры

**Узел Node (зелёный цвет)**

Описывает точку дороги. Содержит поля:

- id – встроенный идентификатор узла (int64);
- node_osm_id – идентификатор точки OpenStreetMap (int64);
- lon – координата (долгота) точки (float);
- lat – координата (широта) точки (float).

**Узел Way (синий цвет)**

Описывает дорогу(транспортную/пешую/водную). Содержит поля:

- id – идентификатор маршрута (int64);
- way_osm_id – идентификатор дороги OpenStreetMap (int64);
- oneway – описание направления дороги (string);
- highway/waterway – описание типа дороги (string).

**Связь FIRST_PATH**

Связывает узел Way с первой точкой Node в неё. Содержит поля:

- id – идентификатор (int64);
- from_id – идентификатор точки источника (int64);
- to_id – идентификатор точки стока (int64).

**Связь PATH**

Связывает узлы Node между собой. Содержит поля:

- id – идентификатор (int64);
- from_id – идентификатор точки источника (int64);
- to_id – идентификатор точки стока (int64);
- length – расстояние между точками в метрах (float).

### 3.2. Нереляционная модель данных

Расчет памяти для Neo4j:

Средний размер Node: 24 байта;

Средний размер Way: 40 байт;

Средний размер First_Path: 24 байт;

Средний размер Path: 28 байт.

Размер данных:

узлы: 0.98 * |V| * size(NODE) + 0.02 * |V| * size(WAY)

связи между узлами: 1.07 * 0.98 * |V| * size(PATH) + 0.02 * |V| * size(FIRST_PATH)

Коэффициенты 0.98 и 0.02 выбраны из расчёта, что в среднем 1 дорога содержит 49 узлов, а 1.07 из расчёта что из 1 из 20 узлов соединён более чем с двумя другими узлами.

Итоговый размер данных: 0,98 * |V| * (size(NODE) + 1.07 * size(PATH)) + 0,02 * |V| * (size(FIRST_PATH) + size(WAY)) = 54,16 * |V|

где |V| - количество узлов в графе.

### 3.3. Аналог модели данных для SQL СУБД

На рисунке 1 представлена модель данных для реляционной базы данных.

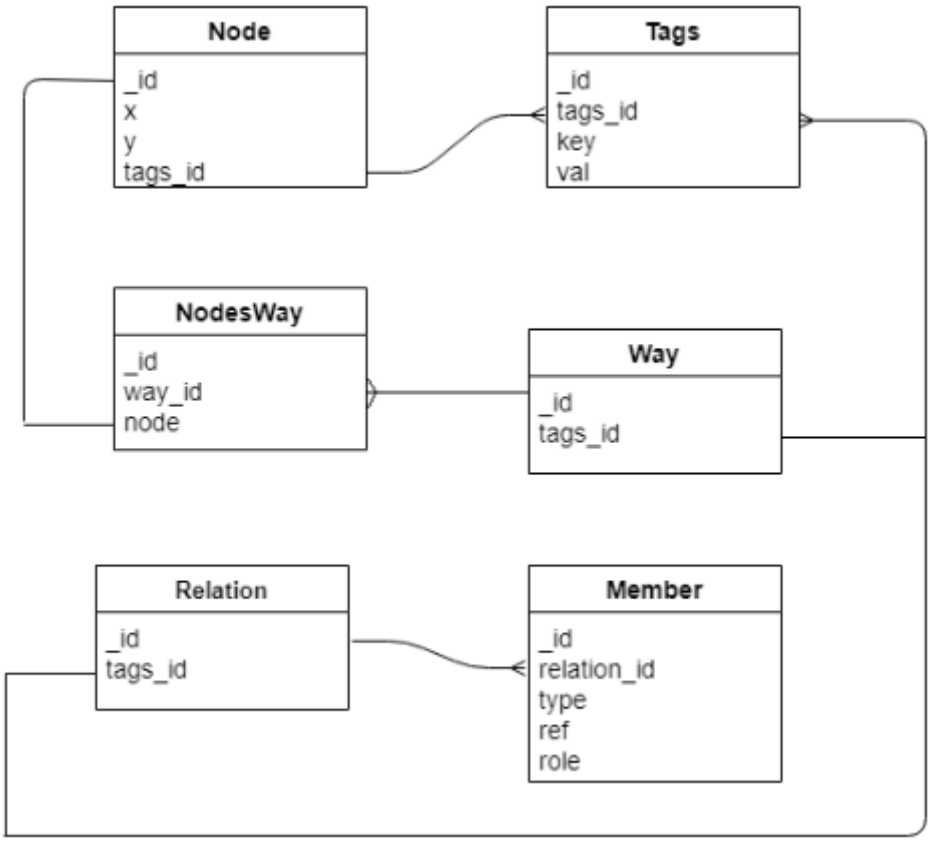


Рисунок 2 – Модель данных реляционной БД

**Описание назначений коллекций, типов данных и сущностей**

6 коллекций:

1-ая Way - хранит данные о путях.

- id () - идентификатор пути;
- tags id - id тега пути

2-ая Node - хранит точки

- id - идентификатор точки;
- x - координата x
- y - координата y
- tags id - id тега пути

3-я Relation - хранит отношения

- id - идентификатор точки;
- tags id - id тега пути

4-ая Member - хранит данные об участниках

- id - идентификатор участника;
- relation id - идентификатор отношения
- type - хранит тип
- ref - хранит ссылку на id участника отношения
- role - роль участника отношения

5-ая NodesWay

- id - идентификатор
- way id - идентификатор пути;
- node - точки;

6-ая Tags

- id - идентификатор
- tags id - идентификатор тега
- key - ключ тега
- val - значение тега

**Оценка удельного объема информации, хранимой в модели**

Среднее количество тегов = 100;

Среднее количество nodes = 100

Среднее количество relation = 1000000

Среднее количество member = 10000000

Size (nodes) = 152

Size (way) = 88

Size (NodesWay) = 36

Size (Way) = 24

Size (Relarion) = 24

Size (Member) = 100

Size = (10088 + 12 + 100(88 + 88100) + 100100) * n

### 3.4. Запросы

Примеры запросов представлены в таблице 1.

Таблица 1 – Примеры запросов

| Формулировка запроса | Neo4j | SQL |
|---|---|---|
| Поиск дорог | MATCH (a:WAY {osm_way_id: $id}) RETURN a | SELECT * FROM Way w WHERE w.type == "restriction"; |
| Информация по координатам | MATCH (a:NODE {osm_node_id: $id}) RETURN a.lat as latitude, a.lon as longitude | SELECT * FROM tip t WHERE t.wayId == $id; |
| Ближайшую к координате | MATCH ()-[:PATH]->(a) WHERE distance(point({latitude: a.lat, longitude: a.lon}), point({latitude: $lat, longitude: $lon})) < ($radius * 1000) RETURN a as node, distance(point({latitude: a.lat, longitude: a.lon}), point({latitude: $lat, longitude: $lon})) AS dist | SELECT * FROM Node n WHERE n.x > 55.7 AND n.x < 55.8 AND n.y > 55.7 AND n.y < 55.8; |

| | | |
|---|---|---|
| | ORDER BY dist<br>LIMIT 1 | |

### 3.5. Выводы

Для рассматриваемой задачи наиболее подходящей базой данных будет нереляционная БД, т.к. сложность запросов будет меньше. Но у нереляционной базы данных тоже есть свои недостатки – объем затрачиваемой памяти больше.

# 4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

## 4.1. Краткое описание

Разработанное приложение осуществляет построение маршрутов. Приложение состоит из главной станицы, на которой отображаются следующие формы:

1. Форма с координатами маркера, id стартовой и конечной точек, длины маршрута;
2. Карта, на которой отображаются маршруты и найденные точки.

## 4.2. Использованные технологии

При написании приложения использовались следующие технологии:

- JavaScript – основной язык программирования, использованный для вывода маршрутов и обращения к СУБД;
- Neo4j – графовая база данных, взятая за основу и для изучения в данной работе;
- npm – менеджер пакетов, входящий в состав Node.js;
- MapBox GL JS – библиотека с открытым исходным кодом, написанная на JavaScript, предназначенная для отображения карт на веб-сайтах;
- HTML – стандартизированный язык разметки документов;
- CSS – формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

## 4.3. Ссылки на Приложение

Исходный код приложения и инструкция по установке находятся по ссылке:

https://github.com/moevm/nosql2018-neo4j_routing

## ЗАКЛЮЧЕНИЕ

В ходе выполнения данного индивидуального домашнего задания было разработано приложение с использованием графовой базы данных Neo4j, с помощью которого выполняются следующие функции:

- построение маршрута из одной точки на карте в другую;
- просмотр координат выбранной точки.

Приложение работает корректно. При создании приложения были изучены особенности работы нереляционных баз данных.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Dyl T., Przeorski K. Mastering Full Stack React Web Development. – 2017.
2. Gackenheimer C. Introduction to React. – Apress, 2015.
3. Добро пожаловать на OpenStreetMap! URL: https://www.openstreetmap.org/#map=16/54.3561/18.6303 (дата обращения: 19.12.2018).

# ПРИЛОЖЕНИЕ А. ДОКУМЕНТАЦИЯ ПО СБОРКЕ И РАЗВЕРТЫВАНИЮ ПРИЛОЖЕНИЯ

Инструкция по сборке и запуску:

1. Скачать проект из репозитория.
2. Запустить сервер Neo4j.
3. Импортировать данные OSM.
4. В папке с проектом в консоли выполнить команду “npm install”.
5. В папке с проектом в консоли выполнить команду “npm start”.
6. Перейти в браузере по адресу: http://localhost:3000.

## ПРИЛОЖЕНИЕ В. ИНСТРУКЦИЯ ДЛЯ ПОЛЬЗОВАТЕЛЯ

При входе на сайт отображается географическая карта.

В левом верхнем углу располагается поиск по карте. Пользователь может выбрать тип построения маршрута: простой/кратчайший по алгоритму Дейкстры.

Просмотреть координаты места можно расположив маркер на нём. В строке над картой будут показаны координаты данного места.

Построить маршрут пользователь может, выбрав пункт отправления и пункт назначения.

# ПРИЛОЖЕНИЕ С. СНИМКИ ЭКРАНА ПРИЛОЖЕНИЯ

На рисунках 3-4 изображены снимки экрана приложения.

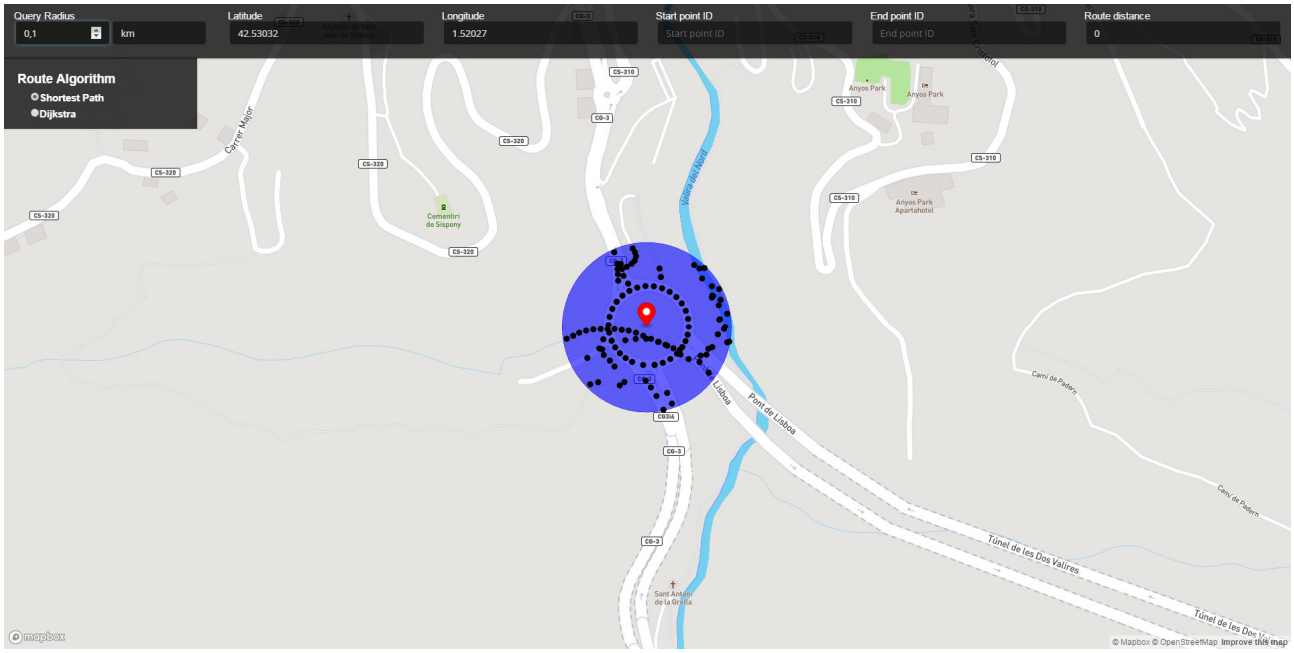


Рисунок 3 – Вид главной страница при запуске

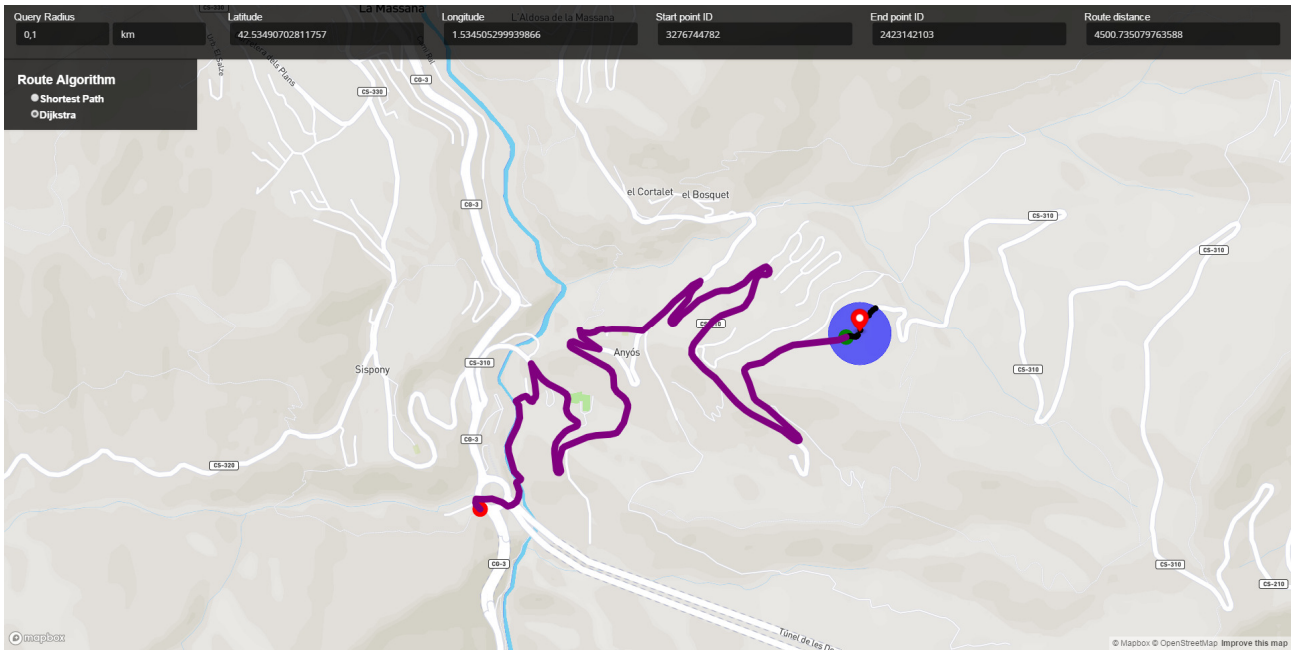


Рисунок 4 – Построение маршрута