**2. Практическая часть**

**2.1 Обзор графовых баз данных.**

Sones GraphDB.

Данная графовая база данных была разработана компанией Sones в 2009 г. и была поддерживаема вплоть до 2011 г., когда Sones прекратила свое существование из-за банкротства. Последняя доступная версия 2.1,которая доступна в двух вариантах: Community, которая распространяется по лицензии AGPL v3, и Enterprise, для коммерческих проектов. Главным отличием платной версии является возможность хранения данных на жестком диске, в отличии от бесплатной, в которой доступно только хранение данных в оперативной памяти. Поскольку компания Sones больше не существует, доступна только бесплатная версия, которая накладывает серьезные ограничения на применение данной БД в реальных проектах и может, в основном, применена только для тестов, при условии частой очистки данных либо применения небольшого количества хранимых данных. Данная БД написана на языке C#.

Neo4J GraphDB.

Разработанная фирмой Neo Technology данная графовая база данных является самой популярной из существующих графовых БД. Свое существование Neo4J начало в 2003 г. На данный момент доступны 3 вида лицензии: Community, Advanced и Enterprise. Бесплатная версия распространяется по лицензии AGPL v3. Для коммерческих проектов предусмотрена лицензия Advanced, добавляющая дополнительные возможности в виде мониторинга состояния базы. В Enterprise версии добавляется функциональность в виде резервного копирования и масштабируемости. В Neo4J реализованы 2 вида кэширования: кэш файлов и кэш объектов. В первом хранятся данные о графе: вершины, ребра, которые служат для улучшения производительности алгоритмов обхода графов. Файловый кэш служит для улучшения скорости чтения/записи с жесткого диска, путем сохранения в специальном оптимизированном формате. Для импорта больших объемов данных предусмотрен режим BatchInserter, который отключает транзакций в БД, следствием чего является резкое увеличение скорости импорта. В Neo4J реализована поддержка алгоритмов работы с графами, такими как поиск в ширину, нахождение кратчайшего пути в графе. В качестве языка манипулирования данными служит специально разработанный язык Cypher.

Sparksee.

В предыдущем известная как DEX GraphDB, разрабатываемая компанией Sparsity Technologies. Доступна в двух видах лицензии: Personal и Commercial license. Бесплатная версия содержит ограничения в виде максимального количества вершин не более одного миллиона, но не ограничивает количества ребер графа. В Commercial лицензии нет каких-либо ограничений. Sparksee полноценно хранит свои данные на жестком диске. Одним из преимуществ Sparksee является то, что БД написана на языке C++, что дает прирост производительности, но может негативно сказаться на поддержке различными операционными системами. Кэширование данных реализовано в виде объектного хранилища, в котором хранятся часто запрашиваемы объекты [11]. Также как и Neo4J в данной БД есть реализации алгоритмов работы с графами: поиск в ширину, поиск в глубину, алгоритм Дейкстры, поиск связанных компонентов.

В дальнейшам для анализа будет использоваться графовая БД Neo4J.

**2.2 Анализ использования графовой БД Neo4J в геолокационных приложениях.**

Язык запросов Cypher.

Для работы с базой данной neo4j используется специальный язык Cypher. Cypher является декларативным языком запросов для графов, который позволяет выразительно и эффективно выполненить запросы и обновленить базу графов. Cypher является относительно простым, но также мощным языком. Сложные запросы к базе данных легко могут быть выражены через Cypher.

Cypher разработан как человекочитаемый язык запросов, подходящий как для разработчиков, так и специалистов аналитиков. Задачей языка состоит в том, чтобы сделать простые вещи легко, и сложные вещи возможными. Его конструкции основаны на английской языке и иконографии, которые помогают сделать запросы более понятными. Язык старается быть удобным как для чтения, так и для записи.

Cypher заимствует свою структуру из SQL - запросы строятся с использованием различных операций.

Операции могут быть соединены друг с другом, и они распределяют промежуточные результаты между собой. Например, совпадающие переменные из одного пункта MATCH имеют один контекст между операциями.

Язык запросов состоит из нескольких различных операций:

* MATCH: Шаблон графа для поиска. Наиболее распространенный способ получения данных из графа.
* WHERE: Используется как часть часть MATCH, OPTIONAL MATCH и WITH. Добавляет ограничения на шаблон, или фильтрет промежуточные результаты проходящий через WITH.
* RETURN: Что будет возвращено.

Использование neo4j в приложении.

Для текущего проекта будет использоваться следующая модель данных, представленная на рисунке 3.1

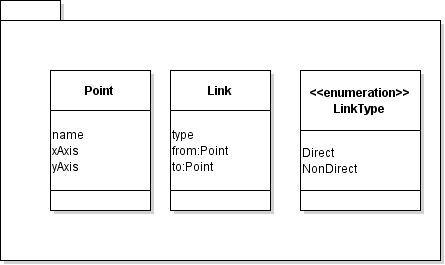


Рис.3.1 Модель данных приложения

Для создания в базе данных вершин графа используется следующий запрос:

CREATE (p:Point {name:'First', xAsis:15.15, yAxis:15.15})

CREATE (p:Point {name:'Second', xAsis:20.20, yAxis:20.20})

Для создания ребер графа использется следующий запрос:

MATCH (f:Point {name:'First'}) MATCH (s:Point{name:'Second'}) CREATE (f)-[d:DIRECT]->(s)

Для анализа работы с алгоритмами будет использоваться следующий граф.

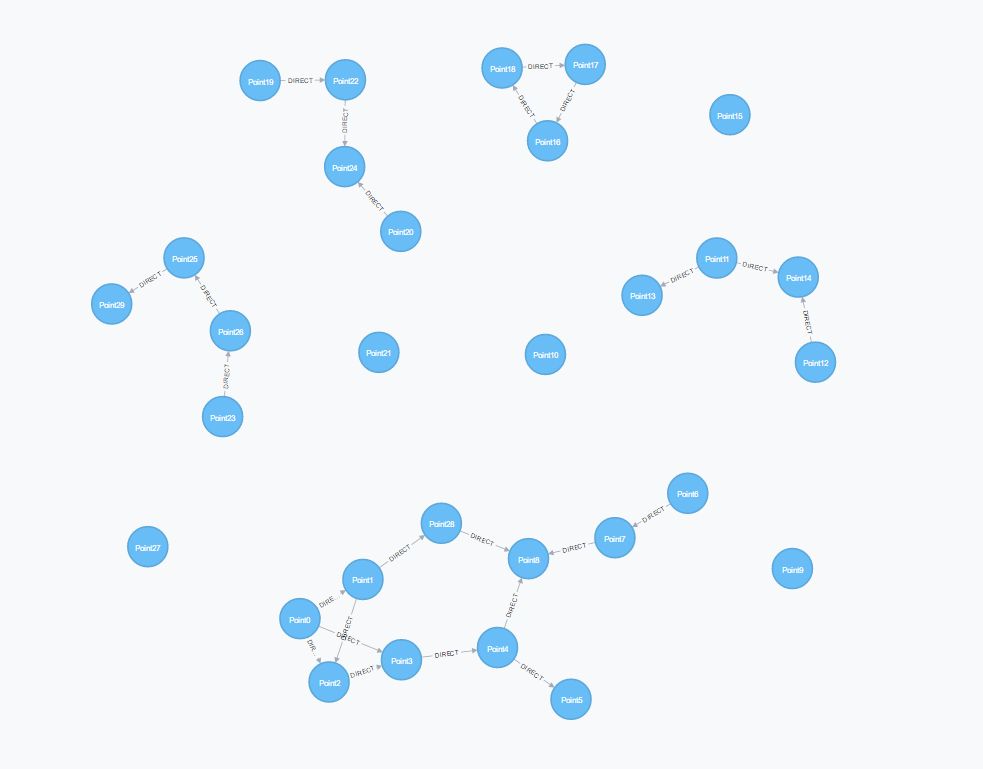


Рис. 3.2 – Тестовый граф

В базе данных neo4j также присутствуют реализации алгоритмов работы с графами, таких как поиск связанности вершин.

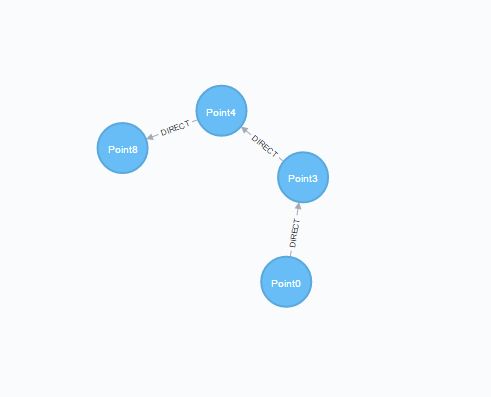
MATCH (f:Point{name:'Point0'})-[\*]->(s:Point{name:'Point8'}) RETURN count(\*)>0

Данный запрос вернет булево значение в зависимости связаны ли 2 вершины.

Другим встроенным алгоритмом является нахождение кратчайшего пути

MATCH (f:Point { name:"Point0" }),(s:Point { name:"Point8" }), p = shortestPath((f)-[\*]-(s)) RETURN p

Результатом выполнения будет являться подграф являющимся кратчайшим путем для данного графа.



3.3 – Результат нахождения кратчайшего пути для тетсового графа

Для примения пространственного анализа Neo4J в java-приложения имеется библиотека Neo4j Spatial. Данная библиотека позволяет выполнять следующие функции:

- Импорт данных из файлов ESRI Shapefile и Open Street Map

- Поддержка разных геометрических фигур

- Реализации структуры данных R-дерево для поиска пространственных данных

- Поддержка топологических операций поиска (содержит, в пределах, пересекается, охватывает, не пересекаются, и т.д.)

- Возможность применения пространственных операций на любом графе

Для выполнения поиска точек в пределах определенной геометрической фигуры может примениться следующий запрос, выполненный через библиотеку Neo4j Spatial:

GraphDatabaseService database = new GraphDatabaseFactory().newEmbeddedDatabase(storeDir);

try {

SpatialDatabaseService spatialService = new SpatialDatabaseService(database);

Layer layer = spatialService.getLayer("layer\_roads");

SpatialIndexReader spatialIndex = layer.getIndex();

Search searchQuery = new SearchIntersectWindow(new Envelope(xmin, xmax, ymin, ymax));

spatialIndex.executeSearch(searchQuery);

List<SpatialDatabaseRecord> results = searchQuery.getResults();

} finally {

database.shutdown();

}

Результатом выполнения будут вершины графа, координаты которого будут располагаться внутри параметров xmin, xmax, ymin, ymax.