УДК 004.6:528.9
DOI 10.33764/2411-1759-2022-27-3-95-106

Возможности NOSQL СУБД для обработки пространственных данных

А. А. Колесников[1] *

[1] Сибирский государственный университет геосистем и технологий, г. Новосибирск, Российская Федерация
* e-mail: alexeykw@mail.ru

Аннотация. Все большее число систем управления базами данных расширяют свою функциональность для работы с различными типами пространственных данных. Это справедливо как для СУБД, основанных на реляционных, так и на NoSQL моделях данных. В статье приводятся основные особенности тех моделей данных, для которых реализованы функции хранения и обработки пространственных данных. Рассмотрены методы искусственного интеллекта, которые реализованы на основе той или иной СУБД. Выполнен сравнительный анализ производительности типовых пространственных запросов для систем управления базами данных, базирующихся на различных моделях данных, в том числе и мультимодельных. Набор данных, на котором выполняется сравнение, представлен в виде трех блоков векторных данных OpenStreetMap на территорию Новосибирской области. По результатам исследования приводятся рекомендации по использованию тех или иных моделей данных в зависимости от имеющихся данных и решаемых задач.

Ключевые слова: СУБД, пространственные данные, методы обработки, пространственный анализ, искусственный интеллект, машинное обучение

Введение

На сегодняшний день большинство геоинформационных систем используют в качестве основной модели хранения данных реляционную модель. Причем она может быть реализована как в файлах формата самой геоинформационной системы, так и с использованием какой-либо сторонней реляционной системы управления базами данных (РСУБД). Реляционные базы данных обладают рядом неоспоримых преимуществ [1, 2], но значительное развитие получили альтернативные модели хранения (NoSQL), которые также имеют свои плюсы [3, 4]. Эти положительные стороны рассмотрены достаточно подробно [5–7], но далеко не всегда затрагивают сторону хранения и обработки пространственных данных. При этом многие из разработчиков NoSQL СУБД сейчас добавляют и расширяют функции по работе с пространственными данными [8]. Также сейчас не менее важным, по сравнению с выполнением типовых операций пространственного анализа, становится реализация методов искусственного интеллекта на основе внутренних механизмов СУБД. Основной идеей описанного в статье исследования являлось сравнение производительности СУБД, основанных на различных моделях хранения данных, при выполнении базовых пространственных запросов, а также анализ доступных функций обработки данных с помощью методов искусственного интеллекта и машинного обучения.

Методы и материалы

Исследование состояло из следующих блоков: формирование набора векторных пространственных данных, подбор NoSQL СУБД, позволяющих делать запросы к пространственным данным, составление списка идентичных по смыслу пространственных запросов, их выполнение, анализ возможностей и формулировка выводов [9–12].

Набор векторных данных для экспериментов представляет собой объекты (точечные, линейные и площадные, слои с точками сервиса, дорожной сети и землепользования соответственно) из OpenStreetMap на территорию Новосибирской области. Для оценки влияния количества объектов на итоговую производительность был сформированы поднаборы на территорию города Новосибирска и отдельно Ленинского района (рис. 1).

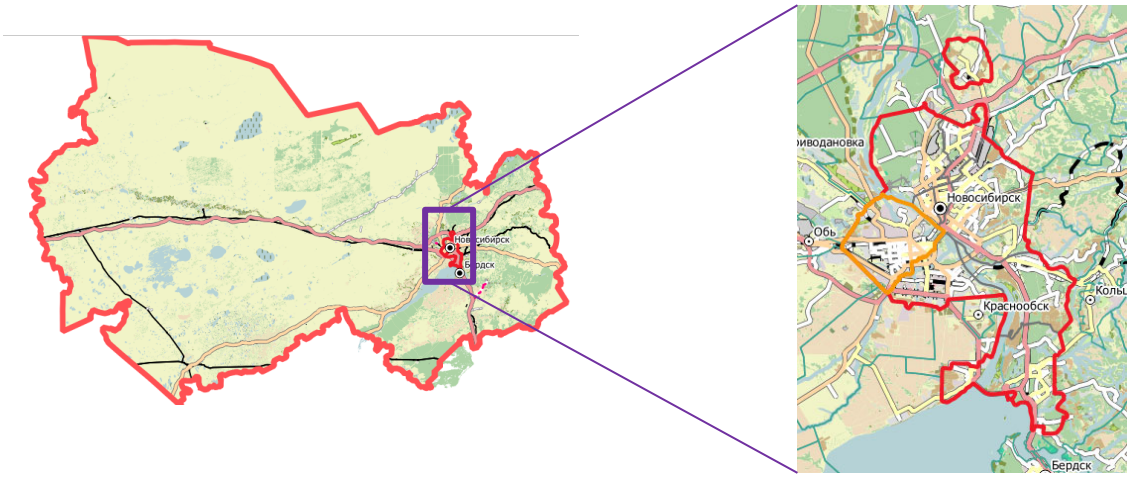

Рис. 1. Используемые данные OpenStreetMap

Количество объектов каждого типа по сформированным поднаборам представлено в табл. 1.

В качестве базовой СУБД, с которой в исследовании выполнялось сравнение NoSQL решений, была взята реляционная СУБД PostgreSQL с модулями PostGIS и pgRouting. Данный выбор был сделан исходя из популярности этого программного обеспечения при реализации хранилищ данных для различных геоинформационных систем.

Таблица 1

Количество объектов в использованных наборах данных

Обьекты	Область	Город	Район города
Точечные (POI)	12 102	8 631	986
Линейные (дорожная сеть)	104 615	34 886	2 129
Площадные (объекты землепользования)	23 070	4 843	513

Основные модели хранения данных (в том числе и пространственных) NoSQL относятся к следующим категориям:

– документо-ориентированные (англ. document based);

– графовые (англ. graph);

– колоночные (англ. wide column);

– ключ-значение (англ. key-value);

– поисковые системы (англ. search engine) [13, 14].

В качестве основных преимуществ NoSQL решений (справедливым и в случае хранения пространственных данных) выделяют:

– динамически модифицируемую схему данных;

– увеличение скорости обработки по сравнению с реляционной моделью, в случаях создания схемы хранения наиболее, оптимально подходящей под решаемую задачу;

– более гибкую горизонтальную масштабируемость, также по сравнению с реляционной моделью [15].

При этом недостатки NoSQL СУБД при работе с пространственными данными, также присутствуют. Среди наиболее значимых нужно отметить:

– меньшее число типов пространственных данных, доступных для хранения (оценка по ISO/IEC 13249: 2016) по сравнению с наиболее популярными реляционными СУБД;

– в большинстве случаев присутствуют только базовые функции по работе с пространственными данными (поиск пересечений, объектов в области, на расстоянии и т. п.), а более

сложные задачи подразумевается решать путем использования внешних программных библиотек;

– меньшее количество типов пространственных индексов, по сравнению с реляционными СУБД, что не всегда позволяет реализовать потенциально большую скорость обработки данных;

– сложности интеграции с распространенными ГИС в качестве основного хранилища данных в отличие от реляционных СУБД, которые могут полностью заменить файловое хранилище;

– сложности с хранением и обработкой трехмерных данных, заключающиеся в недостатке функций, учитывающих высотные данные при выполнении пространственных запросов и геометрических вычислениях;

– сложности с использованием пользовательских систем координат в NoSQL СУБД, заключающиеся в отсутствии, либо недостаточно полной реализации наиболее распространенных форматов описаний систем координат (PROJ6, WKT, ProjJSON).

Но, как уже было указано ранее, перечисленные недостатки NoSQL СУБД (в том числе и мультимодельных) компенсируются более широкими возможностями распределенного хранения и обработки и тем самым потенциального увеличения скорости обработки, а также дополнительными функциями в случае отдельных задач, что и планировалось подтвердить описанным исследованием.

Рассмотрим далее особенности основных моделей хранения NoSQL и их реализаций, поддерживающих работу с пространственными данными.

Документо-ориентированные СУБД в качестве основы хранения используют понятие документа, который представляет собой совокупность наборов атрибутов (состоящих из ключа и соответствующего ему значения). Значения атрибутов могут быть, в свою очередь, вложенными документами или массивами. Документы одного типа (внутри одной коллекции) могут иметь общие атрибуты, а могут и не иметь их, поскольку отсутствует жестко заданная схема. Такие СУБД во многих случаях позволяют разработчику отражать сущности предметной области на сущности БД без введения дополнительных сущностей, которые приходится вводить в реляционных СУБД, то есть отношения один-к-одному и один-ко-многим отображаются без дополнительных таблиц и полей для связи. К особенностям документо-ориентированных СУБД также относятся:

– преимущественное использование формата GeoJSON для хранения пространственных данных;

– собственный язык запросов (хотя существуют трансляторы с языка SQL);

– базовый набор геометрических типов пространственных объектов;

– хранение сложных структур при более простой схеме данных по сравнению с реляционными;

– потенциально более быстрое выполнение базовых пространственных запросов (при условии больших объемов данных, корректной настройке индексов и схемы распределенного хранения);

– изначальная реализаций функций распределенного хранения и обработки, в том числе и пространственных данных [16].

Графовые СУБД изначально ориентированы на связи между объектами, и эти связи могут иметь разные характеристики. Основным преимуществом графовых баз данных является универсальность в виде хранения реляционных, документных и семантических данных. К их особенностям относятся:

– собственный язык запросов (позволяющий задействовать сложные семантические связи базы данных, но требующий освоения конструкций запросов и плохо транслируемый с SQL);

– необходимость преобразования пространственных объектов с разными типами локализации в графовую структуру;

– более быстрое выполнение запросов для анализа сетевых структур (при условии корректно заданной архитектуры и настройки индексов) [17, 18].

Колоночные базы хранят данные не в строках (в случае реляционных СУБД), а в колонках. Они представляют собой как бы отдельную таблицу из одной колонки, которая хранит только свои значения. Кроме этого, все данные в колоночной базе данных обычно хра-

нятся в отсортированном виде. К особенностям колоночных СУБД относятся:

– собственный язык запросов (как правило, очень близкий по структуре к SQL);

– необходимость проработки оптимальной структуры базы данных, с точки зрения физического размещения отдельных колонок по имеющимся серверным хранилищам;

– базовый набор геометрических типов пространственных объектов;

– высокая скорость записи группированных (то есть поступающих блоками из нескольких записей) данных;

– более быстрое выполнение базовых пространственных запросов (при условии корректной настройки группировки данных внутри колонок);

– реализаций функций распределенного хранения и обработки, в том числе и для пространственных данных [19].

База данных на основе пар хранит данные как совокупность пар «ключ – значение», в которых ключ является уникальным идентификатором элемента. Значения как ключей, так и значений могут представлять собой как обычный набор символов, так и сложный составной объект. Базы данных с использованием пар «ключ – значение» обеспечивают высокую степень параллельной обработки и горизонтальное масштабирование, часто не реализуемую при использовании других моделей баз данных. К особенностям СУБД «ключ – значение» относятся:

– собственный язык запросов;

– необходимость проработки оптимальной структуры базы данных;

– базовый набор геометрических типов пространственных объектов;

– более быстрое выполнение базовых пространственных запросов (при условии корректной настройки группировки данных);

– быстрое выполнение базовых операций за счет in-memory механизмов (при условии достаточности аппаратных ресурсов).

Поисковые системы изначально были ориентированы на работу с текстовой информацией, но сейчас их функциональность расширилась и позволяет работать и с пространственными объектами. К особенностям поисковых систем относятся:

– собственный язык запросов, ориентированный в большей степени на обработку текстов;

– базовый набор геометрических типов пространственных объектов;

– более быстрое выполнение базовых пространственных запросов в сочетании с атрибутивными, а также метаданных (при условии корректной настройки индексов).

Проанализировав популярность, функциональность при работе с пространственными данными, интегрируемость с геоинформационными системами для проведения эксперимента, были выбраны следующие СУБД в каждой категории:

– документо-ориентированные – MongoDB;

– графовые – Neo4j;

– колоночные – Cassandra;

– ключ-значение – Redis;

– поисковые системы – ElasticSearch [20, 21].

После выбора программного обеспечения необходимо было импортировать исходный набор данных в каждую из СУБД. По завершении этой операции, из-за особенностей представления данных в конкретной модели часть общего набора не была использована. Кроме этого, в Neo4j и Cassandra данные были загружены, но в стандартных функциях не было подходящих операторов пространственного анализа, и также часть объектов не учитывалась. Сводная информация о используемых типах объектов по локализации в конкретной СУБД приведена в табл. 2.

Для каждой категории СУБД были выбраны те запросы, которые также позволяет реализовать СУБД PostgreSQL. Содержание самих запросов рассмотрено в разделе результатов данной статьи. За единицу принималось время выполнения запроса в СУБД PostgreSQL. В таблицах результатов приведено значение, показывающее, во сколько раз быстрее или медленнее выполнялся аналогичный запрос в указанной NoSQL СУБД. То есть цифра меньше единицы обозначает, что запрос выполнялся медленнее, чем в реляционной СУБД, больше единицы – соответственно наоборот. Все эксперименты проводились на одном и том же аппаратном обес-

печении. В результатах заголовки таблиц «Область», «Город», «Район» обозначают наборы объектов, относящиеся к Новосибирской области, городу Новосибирску и Ленинскому району города Новосибирска соответственно.

Таблица 2

Типы данных в СУБД относительно слоев исходного набора данных

Слои	PostGIS	MongoDB	Neo4j	Cassandra	Redis	ElasticSearch
Слой с объектами сервиса (POI)	Точка	Точка	Точка	Точка	Точка	Точка
Слой дорожной сети	Линия	Линия	Линия	Линия (загружен но не использовался в расчетах)	Не применимо	Линия
Слой объектов землепользования	Полигон	Полигон	Полигон (загружен, но не использовался в расчетах)	Не применимо	Не применимо	Полигон

Результаты

Далее рассмотрим результаты выполнения определенных запросов. Первым выполнялся запрос на поиск объектов определенного типа локализации внутри одного указанного площадного объекта. Результаты приведены на рис. 2.

СУБД	Область	Город	Район
PostgreSQL	1	1	1
MongoDB	1,5	1,4	1
ElasticSearch	1,4	1,4	1,3

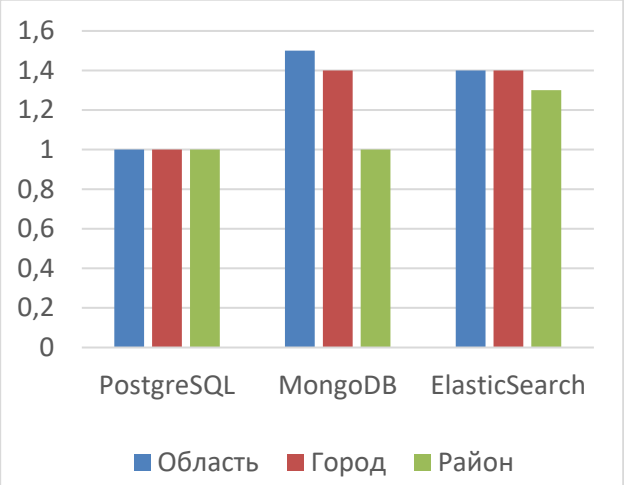

Рис. 2. Поиск объектов, отфильтрованных по семантике, внутри площадного объекта

Вторым запросом выполнялся поиск пересечений (с формированием точечных объектов в общих точках) между объектами двух заданных типов локализации и с фильтром по значениям семантических характеристик (рис. 3).

СУБД	Область	Город	Район
PostgreSQL	1	1	1
MongoDB	3,1	2,6	1,2
Cassandra	0,9	0,9	0,8
OrientDB	2,7	2,5	2
CosmosDB	2,1	1,8	1,8

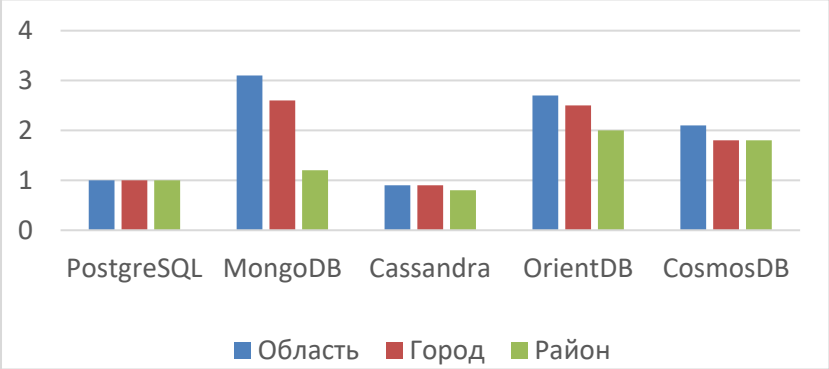

Рис. 3. Формирование точечных объектов
на пересечении двух наборов объектов

Далее выполнялся запрос для поиска объектов определенного типа локализации с заданными значениям семантической характеристики на указанном расстоянии от заданной точки (рис. 4).

СУБД	Область	Город	Район
PostgreSQL	1	1	1
MongoDB	4,4	3,6	1,8
Cassandra	1,4	1,1	1,1
ElasticSearch	1,7	1,1	1,1

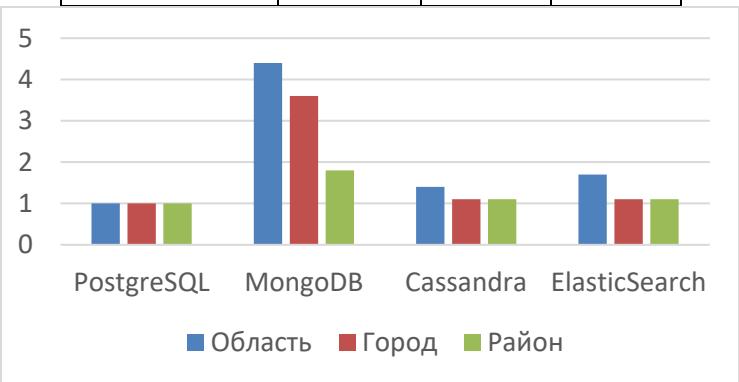

Рис. 4. Поиск объектов, отфильтрованных по семантике,
на указанном расстоянии от заданной точки

Следующим запросом выполнялся поиск одного ближайшего точечного объекта с заданными критериями отбора по значениям семантических характеристик (рис. 5).

СУБД	Область	Город	Район
PostgreSQL	1	1	1
Neo4j	1,3	1,2	1,1

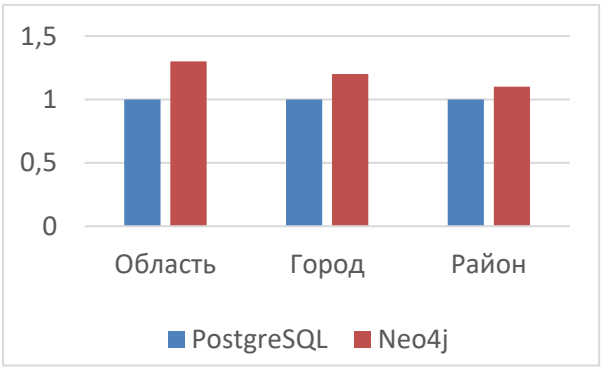

Рис. 5. Поиск ближайшего точечного объекта

Пятым запросом выполнялся поиск кратчайшего пути (набор линейных объектов) между двумя указанными точечными объектами (рис. 6).

СУБД	Область	Город	Район
PostgreSQL	1	1	1
Neo4j	1,7	1,3	1
OrientDB	2,9	1,6	1,4
CosmosDB	2,1	1,4	1,4

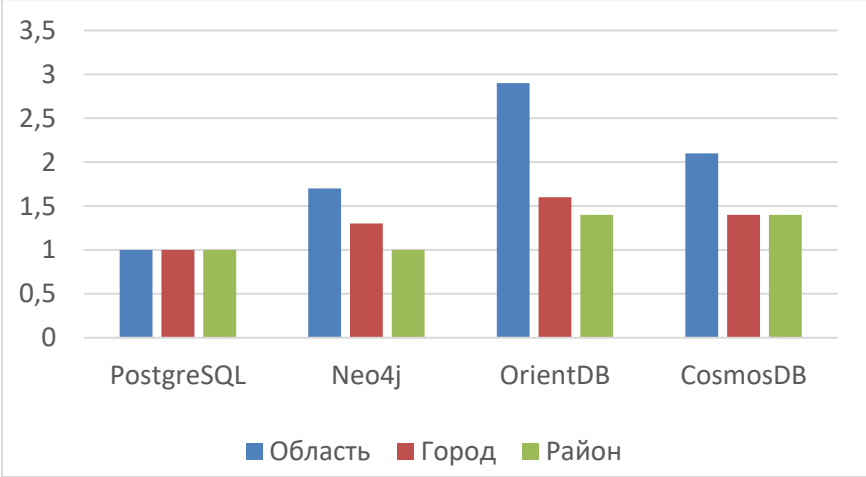

Рис. 6. Поиск кратчайшего пути между двумя точками

Шестым запросом выполнялось вычисление расстояния между двумя заданными точками (рис. 7).

После проведенных экспериментов были сформулированы представленные ниже выводы и замечания по используемым СУБД.

MongoDB показала повышенную производительность при обработке самого большого набора данных по Новосибирской области за исключением поиска объектов внутри полигональной области со сложной геометрией. В случае этой СУБД нужно отметить большие затраты времени на корректную (без потери или искажения семантики объектов) загрузку данных в базу данных.

Специфика графовой базы данных позволила Neo4j показать более высокую производительность при поиске кратчайшего пути в случае наиболее объемного набора данных.

Более медленной выполнение запросов СУБД Cassandra обусловлено необходимостью проработки оптимальной структуры базы данных для выбранных наборов и настройки группировки данных в колонках.

СУБД	Область	Город	Район
PostgreSQL	1	1	1
Redis	4,3	3,8	3,3
OrientDB	2,1	1,6	1,5
CosmosDB	2,4	2,1	2,1

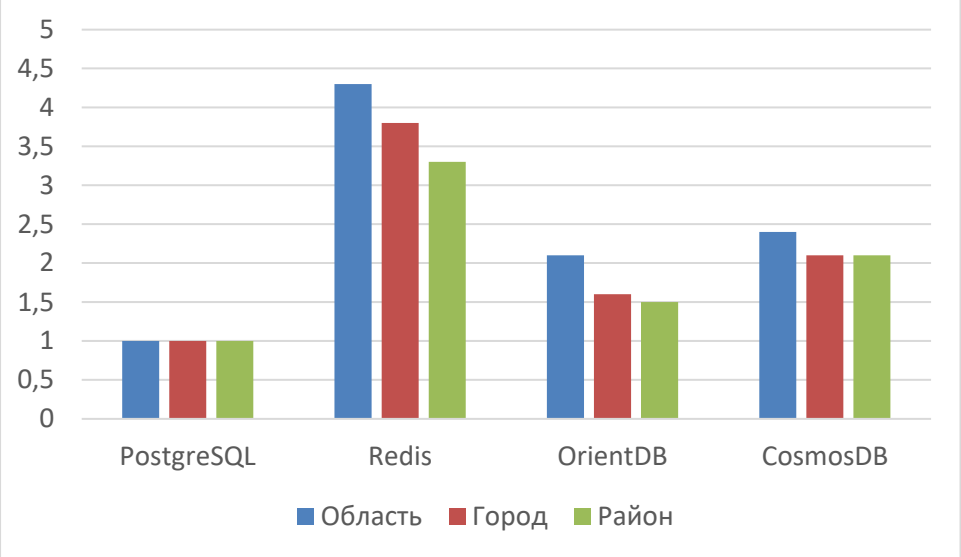

Рис. 7. Вычисление расстояния между заданными точками

Высокая скорость СУБД Redis объясняется использованием механизмов хранения и обработки данных в оперативной памяти, но это ограничивает эффективный объем данных (для крупных геоинформационных систем) и виды возможных пространственных запросов.

ElasticSearch также использует механизмы использования оперативной памяти для хранения и обработки данных, что объясняет, в целом, более высокую скорость при выполнении запросов, но сами возможные запросы весьма ограничены по реализуемым функциям.

В качестве дополнительной альтернативы скорость выполнения запросов были проанализированы в мультимодельных СУБД OrientDB и CosmosDB. Производительность по результатам эксперимента оказалась близка к PostgreSQL, но возможность разделять данные по разным моделям хранения позволяет расширить возможный функционал хранилища данных.

Кроме сравнения производительно сформирована сводная таблица возможностей использованных СУБД с точки зрения реализации возможностей дальнейшей обработки рассмотренных данных с помощью технологий искусственного интеллекта и машинного обучения (табл. 3).

Таблица 3

Использование элементов технологий искусственного интеллекта в СУБД

Технологии	PostgreSQL (используя модуль MADLib)	MongoDB (в варианте MindsDB)	Neo4j	Cassandra (используя библиотеку MLLib)	Redis (используя модуль RedisAI)	ElasticSearch
Условные случайные поля (CRF)	+	–	–	–	+	–
K-ближайших соседей	+	+	–	+	+	–
Нейронные сети	+	–	+	–	+	+

Окончание табл. 3

Технологии	PostgreSQL (используя модуль MADLib)	MongoDB (в варианте MindsDB)	Neo4j	Cassandra (используя библиотеку MLLib)	Redis (используя модуль RedisAI)	ElasticSearch
Регрессионные модели	+	+	+	–	+	–
Машины опорных векторов	+	–	–	–	+	–
Деревья решений (случайный лес)	+	+	–	+	+	+
Байесовские методы				+	+	–

Заключение

1. Реляционные СУБД в текущем варианте однозначно лучше подходят для работы с пространственными данными в небольших проектах.

2. У каждой модели хранения данных есть преимущества для конкретного сценария использования.

3. Большинство NoSQL решений ориентированы на работу с web-интерфейсами и есть трудности по их интеграции в настольные ГИС.

4. NoSQL СУБД изначально лучше приспособлены для распределенного хранения и обработки, но часто требуют больше времени на установку и настройку по сравнению с реляционными СУБД.

5. Производительность сильно зависит от настройки СУБД, индексов (и их типа), структуры базы данных.

6. Геометрические вычисления на основе пространственных данных в СУБД NoSQL могут выполняться быстрее за счет точности получаемых координат (данный режим доступен в ряде СУБД).

В дальнейших исследованиях планируется сформулировать рекомендации по использованию конкретных моделей данных в зависимости от решаемой задачи, определенных типов индексов в зависимости от используемых пространственных данных и наиболее частых запросов, отдельных алгоритмов и методов искусственного интеллекта для обработки пространственных данных, а также способы повышения производительности в многозадачности.

Статья подготовлена в рамках выполнения гранта, предоставленного в форме субсидии на проведение крупных научных проектов по направлениям научно-технологического развития в рамках подпрограммы «Фундаментальные научные исследования для долгосрочного развития и обеспечения конкурентоспособности общества и государства» государственной программы Российской Федерации «Научно-технологическое развитие Российской Федерации», проект «Социально-экономическое развитие Азиатской России на основе синергии транспортной доступности, системных знаний о природно-ресурсном потенциале, расширяющегося пространства межрегиональных взаимодействий», номер соглашения с Министерством науки и высшего образования Российской Федерации № 075-15-2020-804 (внутренний номер гранта № 13.1902.21.0016).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Тимофеева Н. Е., Дмитриева К. А. Сравнительный анализ реляционной и нереляционной модели хранения служебной информации централизованной распределенной базы данных // Вестник Российского нового университета. Серия: сложные системы, модели, анализ и управление. – 2019. – Т. 1. – С. 66–74.

2. Ali W., Shafique M. U., Majeed M. A., Raza A. Comparison between SQL and NoSQL Databases and Their Relationship with Big Data Analytics // Asian Journal of Research in Computer Science. –2019. – Vol. 4(2). – P. 1–10. doi: 10.9734/ajrcos/2019/v4i230108.

3. Reniers V., Rafique A., Van Landuyt D. Object-NoSQL Database Mappers: a benchmark study on the performance overhead // Journal of Internet Services and Applications – 2017. – Vol. 8. – 1. doi: 10.1186/s13174-016-0052-x.

4. Győrödi C. A., Dumşe-Burescu D. V., Zmaranda D. R., Győrödi R. Ş., Gabor G. A., Pecherle G. D. Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application's Data Storage // Applied Sciences. – 2020. – Vol. 10 (23). – P. 8524. doi: 10.3390/app10238524.

5. Shmueli G. Research Dilemmas with Behavioral Big Data // Big Data. –2017. – Vol. 5(2). – P. 98–119. doi: 10.1089/big.2016.0043.

6. Королева Ю. А., Маслова В. О., Козлова В. К. Разработка концепции миграции данных между реляционными и нереляционными системами БД // Программные продукты и системы. – 2019. – Т. 1. – С. 63–67.

7. Hasan M. Performances analysis of NoSQL and relational databases for analyzing GeoJSON spatial data // Перспективы науки. –2019. – Т. 7. – С. 40–42.

8. Mabele B. C. P. Fundamentals of the geographic information database of the specially protected natural areas of the Republic of Congo // Изв. вузов. Геодезия и аэрофотосъемка. – 2020. – Т. 64 (5). – С. 596–607.

9. Preuveneers D., Joosen W. Automated Configuration of NoSQL Performance and Scalability Tactics for Data-Intensive Applications // Informatics. – 2020. – Vol. 7(3). – P. 29. doi: 10.3390/informatics7030029.

10. Kabakus A. T., Kara R. A performance evaluation of in-memory databases // Journal of King Saud University – Computer and Information Sciences. – 2017. – Vol. 29 (4). – P. 520–525. doi: 10.1016/j.jksuci.2016.06.007.

11. Laksmita N., Apriliyanto E., Pandu I., Rini K. Comparison of NoSQL Database Performance with SQL Server Database on Online Airplane Ticket Booking // Indonesian Journal of Applied Informatics. – 2020. – Vol. 4(2). – P. 64–75. doi: 10.20961/ijai.v4i2.38956.

12. Wisal K., Ejaz A., Waseem S. Predictive Performance Comparison Analysis of Relational & NoSQL Graph Databases // International Journal of Advanced Computer Science and Applications(IJACSA). – 2017. – Vol. 8(5). doi: 10.14569/IJACSA.2017.080564.

13. Guo D., Onstein E. State-of-the-Art Geospatial Information Processing in NoSQL Databases // ISPRS International Journal of Geo-Information. – 2020. – Vol. 9(5). – P. 331. doi: 10.3390/ijgi9050331.

14. Бёрнс Б. Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable services / Б. Бёрнс. – O'Reilly Media, 2018. – 166 с.

15. Holubová I., Scherzinger S. Unlocking the potential of NextGen multi-model databases for semantic big data projects // In Proceedings of the International Workshop on Semantic Big Data (SBD '19). – 2019. – Vol. 6. – P. 1–6. doi: 10.1145/3323878.3325807.

16. Divya C., Bansal K. L. Using the Advantages of NOSQL: A Case Study on MongoDB // International Journal on Recent and Innovation Trends in Computing and Communication. – 2017. – 5.2. – P. 90–93.

17. Webber J. A programmatic introduction to neo4j // Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity. – 2012. – С. 217–218.

18. Dominguez-Sal D., Urbon-Bayes P., Gimenez-Vano A., Gomez-Villamor S., Martınez-Bazan N., Larriba-Pey J.L. Survey of graph database performance on the HPC scalable graph analysis benchmark // Proceedings of the 2010 International Conference on Web-age Information Management (WAIM'10). Berlin, Heidelberg, Springer-Verlag, 2010. – P. 37–48.

19. Yamaguchi S., Morimitsu Y. Improving Dynamic Scaling Performance of Cassandra // IEICE Transactions on Information and Systems. – 2017. – Vol.E100.D(4). – P. 682–692. doi: 10.1587/transinf.2016DAP0009.

20. Беладинович С. Новый подход к проектированию гибридных баз данных SQL. NoSQL на основе данных. Структурированность // Информационные системы предприятия. – 2018. – С. 1–19.

21. Demidova L., Nikulchev E., Sokolova Yu. Big data classification using the SVM classifiers with the modified particle swarm optimization and the SVM ensembles // International journal of advanced computer science and applications. – 2016. – Vol. 7(5). – P. 294–312.

Об авторах

Алексей Александрович Колесников – доцент, кандидат технических наук, доцент кафедры картографии и геоинформатики.

Получено 17.03.2022

Functions of NOSQL DBMS for processing spatial data

A. A. Kolesnikov[1] *

[1] Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
* e-mail: alexeykw@mail.ru

Abstract. An increasing number of database management systems are expanding their functionality to work with various types of spatial data. This is true for both relational and NoSQL data models. The article presents the main features of those data models for which the functions of storing and processing spatial data are implemented. The methods of artificial intelligence, which are implemented on the basis of a particular data model, are considered. A comparative analysis of the performance of typical spatial queries for database management systems based on various data models, including multi-model ones, is performed. The data set on which the comparison is performed is presented as three blocks of OpenStreetMap vector data for the territory of the Novosibirsk region. Based on the results of the study, recommendations are given on the use of certain data models, depending on the available data and the tasks being solved.

Keywords: DBMS, spatial data, processing methods, spatial analysis, artificial intelligence, machine learning

REFERENCES

1. Timofeeva, N. E., & Dmitrieva, K. A. (2019). Comparative analysis of relational and non-relational models of storage of service information of a centralized distributed database. *Vestnik rossiyskogo novogo universiteta. Seriya: slozhnyye sistemy, modeli, analiz i upravleniye ["Vestnik RosNOU", "Complex Systems: Models, Analysis, Management" Series]*, 1, 66–74 [in Russian].

2. Ali, W., Shafique, M. U., Majeed, M. A., & Raza, A. (2019). Comparison between SQL and NoSQL Databases and Their Relationship with Big Data Analytics. *Asian Journal of Research in Computer Science*, *4*(2), 1–10. doi: 10.9734/ajrcos/2019/v4i230108.

3. Reniers. V., Rafique. A., & Van Landuyt. D. (2017) Object-NoSQL Database Mappers: a benchmark study on the performance overhead. *Journal of Internet Services and Applications*, *8*(1). doi: 10.1186/s13174-016-0052-x

4. Győrödi, C. A., Dumşe-Burescu, D. V., Zmaranda, D. R., Győrödi, R. Ş., Gabor, G. A., & Pecherle, G. D. (2020). Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application's Data Storage. *Applied Sciences*, *10*(23), P. 8524. doi: 10.3390/app10238524

5. Shmueli, G. (2017). Research Dilemmas with Behavioral Big Data. *Big Data*, *5*(2), 98–119. doi: 10.1089/big.2016.0043.

6. Koroleva, Yu. A., Maslova, V. O., & Kozlova, V. K. (2019) Development of the concept of data migration between relational and non-relational database systems. *Programmnyye produkty i sistemy [Software Products and Systems]*, *1*, 63–67 [in Russian].

7. Hasan, M. (2019). Performances analysis of NoSQL and relational databases for analyzing GeoJSON spatial data. *Perspektivy nauki [Perspectives of Science]*, *7*, 40–42.

8. Mabele, B. C. P. (2020). Fundamentals of the geographic information database of the specially protected natural areas of the Republic of Congo. *Izvestia vuzov. Geodeziya I aerofotos"emka [IzvestiyaVuzov. Geodesy and Aerophotosurveying]*, *64*(5), 596–607.

9. Preuveneers, D, & Joosen, W. (2020). Automated Configuration of NoSQL Performance and Scalability Tactics for Data-Intensive Applications. *Informatics*, *7*(3), P. 29. doi: 10.3390/informatics7030029

10. Kabakus, A. T., & Kara, R. (2017). A performance evaluation of in-memory databases. *Journal of King Saud University – Computer and Information Sciences*, *29*(4), 520–525. doi: 10.1016/j.jksuci.2016.06.007.

11. Laksmita, N., Apriliyanto, E., Pandu, I., & Rini, K. (2020) Comparison of NoSQL Database Performance with SQL Server Database on Online Airplane Ticket Booking. *Indonesian Journal of Applied Informatics*, *4*(2), 64–75. doi: 10.20961/ijai.v4i2.38956.

12. Wisal, K., Ejaz, A., & Waseem, S. (2017). Predictive Performance Comparison Analysis of Relational & NoSQL Graph Databases. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *8*(5). doi: 10.14569/IJACSA.2017.080564.

13. Guo, D, & Onstein, E. (2020). State-of-the-Art Geospatial Information Processing in NoSQL Databases. *ISPRS International Journal of Geo-Information*, *9*(5), P. 331. doi: 10.3390/ijgi9050331.

14. Burns, B. (2018). Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable services. O'Reilly Media, 166 p.

15. Holubová, I., & Scherzinger, S. (2019). Unlocking the potential of NextGen multi-model databases for semantic big data projects. *In Proceedings of the International Workshop on Semantic Big Data (SBD '19): Vol. 6* (pp. 1–6). doi: 10.1145/3323878.3325807.

16. Divya C., & Bansal K. L. (2017). Using the Advantages of NOSQL: A Case Study on MongoDB. *International Journal on Recent and Innovation Trends in Computing and Communication*, *5*(2), 90–93.

17. Webber, J. (2012). A programmatic introduction to Neo4j. *Proceedings of the 3rd Annual Conference on Systems, Programming, and Applications: software for humanity* (pp. 217–218).

18. Dominguez-Sal, D., Urbon-Bayes, P., Gimenez-Vano, A., Gomez-Villamor, S., Martınez-Bazan, N., & Larriba-Pey, J. L. (2010). Survey of graph database performance on the HPC scalable graph analysis benchmark. *Proceedings of the 2010 International Conference on Web-age Information Management (WAIM'10)* (pp. 37–48). Berlin, Heidelberg, Springer-Verlag.

19. Yamaguchi, S., & Morimitsu, Y. (2017). Improving Dynamic Scaling Performance of Cassandra. *IEICE Transactions on Information and Systems*, *E100.D*(4), 682–692. doi: 10.1587/transinf.2016DAP0009.

20. Beladinovich, S. (2018). A new approach to designing hybrid SQL databases. Data driven NoSQL. Structured. *Informatsionnyye sistemy predpriyatiya [Enterprise Information Systems]*, pp. 1–19 [in Russian].

21. Demidova, L., Nikulchev, E., & Sokolova, Yu. (2016). Big data classification using the SVM classifiers with the modified particle swarm optimization and the SVM ensembles. *International Journal of Advanced Computer Science and Applications*, *7*(5), 294–312 [in Russian].

Author details

Aleksey A. Kolesnikov – Ph. D., Associate Professor, Department of Cartography and Geoinformatics.

Received 17.03.2022