Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

ОТЧЕТ ПО ЭКСПЛУАТАЦИОННОЙ ПРАКТИКЕ

Студент Журавлев Николай Вадимович
фамилия, имя, отчество

Группа ИУ5-24М

Тип практики ЭКСПЛУАТАЦИОННАЯ

Название предприятия

Студент _____ **Журавлев Н.В.**
подпись, дата *фамилия, и.о.*

Руководитель практики _____ **Балдин А.В.**
подпись, дата *фамилия, и.о.*

Оценка _____

2024 г.

Оглавление

Описание системы сбора информации о работе NoSQL СУБД

Тема ВКР: Система автоматического сбора информации о работе NoSQL баз данных.

Основная идея ВКР в создании приложения, которое на основе полученных баз данных (БД) строит локальную модель и метаинформацию по всем БД, входящим в неё.

СУБД, которые будут использованы это: MongoDB, Cassandra и Neo4j. В них есть хранилище состоящие из разных БД, которые в последствии будут использоваться в локальной модели.

По каждой БД необходимо выводить метаинформацию, которая её описывает, которая делится:

- По логической структуре. В неё входят таблицы, связи и т.п.
- По физической. В неё входят настройки БД, индексы, какая-либо статистика и т.п.

На основе полученных данных строиться локальная модель.

Данное приложение необходимо, например, если имеется несколько БД у какой-либо компании и необходимо хотим получить данные для оптимизации или принять решения об администрировании системы в целом. Так же, чтобы можно получать информации о нагрузке системы и соединять разные удалённые БД.

Обзор выбранных NoSQL СУБД

NoSQL - это подход к проектированию баз данных, который фокусируется на предоставлении механизма для хранения и извлечения данных, моделированных иным образом, чем в табличных отношениях, используемых в реляционных базах данных. В отличие от типичной табличной структуры реляционной базы данных, базы данных NoSQL хранят данные в одной структуре данных. Этот подход не требует схемы и обеспечивает быстрое масштабирование для управления большими и обычно неструктурированными наборами данных.

Плюсы NoSQL:

- Гибкость схемы данных: NoSQL базы данных предназначены для конкретных моделей данных и хранят данные в гибких схемах, которые легко масштабируются для современных приложений.
- Распределенное хранение: NoSQL базы данных обеспечивают распределенное хранение данных, что улучшает производительность и обеспечивает отказоустойчивость.
- Большие объемы данных: NoSQL базы данных хорошо подходят для работы с большими объемами данных, такими как Big Data.

Минусы NoSQL:

- Отсутствие стандартизированного языка запросов: В отличие от SQL, NoSQL базы данных не имеют стандартизированного языка запросов, что может усложнить разработку и поддержку.
- Сложность в обеспечении целостности данных: В NoSQL базах данных может быть сложнее обеспечить целостность данных, особенно в сравнении с реляционными базами данных, где это обеспечивается через отношения (relations).

Не всегда подходят для традиционных приложений: NoSQL базы данных могут не всегда подходить для традиционных приложений, которые обычно используют SQL базы данных, такие как MySQL для CMS, форумов и т.д.

Многомодельная база данных — это база данных, предназначенная для поддержки нескольких моделей данных в одной системе хранения данных. Это означает, что такая система может хранить, индексировать и запрашивать данные в нескольких моделях. Этот тип базы данных обеспечивает единый интерфейс для обеспечения согласованности, безопасности и доступа к данным, а также устраняет необходимость сложных преобразований и миграций между различными базами данных [1].

Далее представлены базы данных, которые используются в ВКР:

1. MongoDB - это СУБД, ориентированная на хранение данных в виде документов похожие на JSON, с опциональными схемами.
2. Apache Cassandra - это бесплатная и открытая система управления базами данных NoSQL, распределенное хранилище широких колонок, разработанное для обработки больших объемов данных на множестве серверов.
3. Neo4j - это графовая база данных, которая хранит данные в виде узлов, связей между ними и атрибутов узлов и связей. Она описывается её разработчиками как ACID-совместимая транзакционная база данных с нативным графовым хранилищем и обработкой.

Эти базы данных имеют следующие модели данных: колоночные, документальные и графовые.

Графовая модель базы данных — это модель, в которой структуры данных для схемы и/или экземпляров моделируются как направленный, возможно, помеченный граф или обобщение структуры данных графа, где манипулирование данными выражается с помощью графо-ориентированных

операций и конструкторов типов, а соответствующие ограничения целостности могут быть определены в структуре графа [2].

Колоночная база данных - это база данных NoSQL, которая хранит данные с использованием колоночного подхода, в отличие от реляционных, которые упорядочивают данные по строкам. Данные, хранящиеся в базе данных семейства столбцов, выбираются вертикально, что делает частичное чтение более эффективным, поскольку загружается только набор атрибутов строки [3].

Вместо хранения данных в фиксированных строках и столбцах базы данных документов используют гибкие документы. Документ – это запись в базе данных документов. Документ обычно хранит информацию об одном объекте и любых связанных с ним метаданных. Документы хранят данные в парах поле-значение. Значения могут быть различных типов и структур, включая строки, числа, даты, массивы или объекты [4].

Описание выбранных СУБД

MongoDB

MongoDB – документно-ориентированная система управления базами данных. Документно-ориентированная СУБД заменяет концепцию «строки» более гибкой моделью, «документом». Позволяя использовать вложенные документы и массивы, документно-ориентированный подход дает возможность представлять сложные иерархические отношения с помощью одной записи. Также нет предопределенных схем: ключи и значения документа не имеют фиксированных типов или размеров. Когда нет фиксированной схемы, добавлять или удалять поля по мере необходимости становится проще. MongoDB – СУБД общего назначения, поэтому помимо создания, чтения, обновления и удаления данных она предоставляет большинство тех функций, которые можно ожидать от системы управления базами данных. Специальные типы коллекций и индексов MongoDB поддерживает коллекции данных TTL (time-to-live), срок действия которых должен истечь в определенное время, такие как сеансы и коллекции фиксированного размера, для хранения недавно полученных данных, например, журналов. MongoDB также поддерживает частичные индексы, ограниченные только теми документами, которые соответствуют фильтру критериев, чтобы повысить эффективность и уменьшить необходимый объем дискового пространства.

Коллекция представляет собой группу документов MongoDB. Коллекции не используют схему. Документы в коллекции могут иметь разные поля. Как правило, все документы в коллекции имеют аналогичное или связанное назначение. Документ – состоит из полей, это набор пар ключ-значение. Документы имеют динамическую схему. Динамическая схема означает, что документы в одной коллекции не обязательно должны иметь одинаковый набор полей или структуру, а общие поля в документах коллекции

могут содержать данные разных типов. Документы являются иерархичными структурами: значения полей документа могут быть в свою очередь вложенными документами или массивами [5].

Каждый запрос к БД начинается с “db”, после чего идёт символ точки, а затем название коллекции, к которой необходимо обратиться. Так же возможно использоваться collection('name_ collection').

Фрагментация является частным случаем горизонтального масштабирования. Суть ее в разделении (партиционировании) базы данных на отдельные части по определенному правилу так, чтобы каждую из них можно было вынести на отдельный сервер. Однако реплицирование данных является обязательным требованием для шардирования в Mongo. Сервер конфигурации и каждый шард представляют из себя так называемый Реплика-сет.

Это группа экземпляров mongod (демон mongoDB), хранящих одинаковые наборы данных. В наборе копии один узел – ключевой, который получает все операции записи. Все остальные узлы – вторичны, принимают операции из первого, таким образом, храня такие же записи, как и первичный узел. Набор копий может иметь только один первичный узел. В случае его отказа другой узел может занять его место либо случайным образом (схема без арбитража), либо по решению сервера-арбитра.

Все операции записи, удаления, обновления, попадают в мастер (primary), а затем записываются в специальную коллекцию oplog, откуда асинхронно попадают на реплики — repl.1 и repl.2 (secondary). Таким образом происходит дублирование данных.

Непосредственно фрагментация происходит следующим образом — данные попадают в фрагменты документа одинакового размера (далее порции) по определенному диапазону заданного поля — shard key. Сначала создается всего одна порция и диапазон значений, которые она принимает, лежит в пределах (-∞, +∞). Когда размер этой порции достигает chunksize, mongos

оценивает значение всех ключей внутри порции и делит порцию таким образом, чтобы данные были разделены примерно поровну.

Предположим размер chunksize, уже достигнут. В данном случае Mongos разделит диапазон примерно так (-беск,45]; (45, +беск).

При появлении новых документов, они будут записываться в порцию, которая соответствует диапазону shardKey. По достижению chunksize (предельное количество записей в одну порцию) разделение произойдет снова и диапазон будет еще уже и так далее. Все порции хранятся в фрагментах.

Важно заметить, что при достижении какой-либо порции неделимого диапазона, например, (44, 45], деление происходить не будет и порция будет расти свыше chunksize. Поэтому следует внимательно выбирать shard key, чтобы это была как можно наиболее случайная величина [6].

Apache Cassandra

Apache Cassandra - это бесплатная и открытая распределенная система управления базами данных NoSQL широкого столбца, предназначенная для обработки больших объемов данных на множестве серверов. Она обеспечивает высокую доступность без единой точки отказа. Cassandra поддерживает кластеры, охватывающие несколько центров обработки данных, с асинхронной репликацией без мастера, обеспечивающей низкую задержку операций для всех клиентов.

В SELECT запросах Cassandra Query Language (CQL) отсутствуют привычные SQL операции JOIN, GROUP BY. А операция WHERE сильно урезана. В SQL вы можете фильтровать по любой колонке, тогда как в CQL только по распределительным ключам (partition key), кластерным ключам (clustering columns) и вторичным индексам.

В Cassandra можно создавать вторичные индексы у любой колонки наподобие SQL индексов. Фактически же, вторичные индексы Cassandra —

это скрытая дополнительная таблица, поэтому производительность WHERE запросов по ним хуже запросов по ключевым колонкам.

СУБД не дает возможности по умолчанию искать по не ключевым колонкам, для которых не создан индекс.

Когда Cassandra получает запрос на запись, данные сначала записываются в журнал закрепления (commit log), а затем в структуру, находящуюся в оперативной памяти и называемую таблицей в памяти (memtable). Операция записи считается успешной, если она записана и в журнал закрепления, и в таблицу в памяти. Записи группируются в памяти и периодически записываются в структуры, которые называются SSTable. После очистки структуры SSTable она не перезаписывается; измененные данные записываются в новую структуру SSTable. Неиспользуемые структуры SSTables удаляются в процессе уплотнения.

База данных Cassandra задумана как очень доступная, потому что в ней нет ведущего узла и все узлы в кластере имеют одинаковые права. Доступность кластера можно увеличить, уменьшив уровень согласованности запросов.

В Cassandra все операции записи данных – это всегда операции перезаписи, то есть, если в колоночную семью приходит колонка с таким же ключом и именем, которые уже существуют, и метка времени больше, чем та которая сохранена, то значение перезаписывается. Записанные значения никогда не меняются, просто приходят более новые колонки с новыми значениями [7].

Neo4j

Neo4j - это система управления графовыми базами данных, которая хранит элементы данных в виде узлов, связей между ними и атрибутов узлов и связей. Она описывается своими разработчиками как транзакционная база

данных, совместимая с ACID, с нативным графовым хранилищем и обработкой данных. Он реализован на Java и доступен из программного обеспечения, написанного на других языках, с использованием языка запросов Cypher.

Основными структурными элементами в базе данных Neo4j являются: Узлы, отношения, метки и свойства.

Узлы – используются для представления сущностей, но в зависимости от отношений в графе могут быть использованы для представления связи. Самый простой граф представляет собой одну вершину. Вершина может не иметь значить, либо иметь одно или более именованных значений, которые указываются в виде свойств.

Отношения – это ассоциативные связи между узлами. В модели данных графа свойств отношения должны быть направленными.

Метки – предоставляют собой графы, которые были сгруппированы в наборы. Все узлы, помеченные одной меткой, принадлежит к одному набору. Упрощают написание запросов к базе. Вершина может быть помечена любым 20 количеством меток. Метки используются для задания ограничений и добавления индексов для свойств.

В рамках отдельного сервера данные всегда являются согласованными, особенно в базе Neo4J, которая полностью поддерживает транзакции ACID. Если база Neo4J работает на кластере, запись на ведущий узел синхронизируется с ведомыми узлами, которые всегда доступны для чтения. Операции записи на ведомые узлы всегда доступны и немедленно синхронизируются с ведущим узлом; остальные ведомые узлы не синхронизируются немедленно – они ждут, пока данные не будут распределены с ведущего узла [7].

Предварительные предложения по реализации

Для реализации приложения в целом можно выбрать язык Python, так как для его функционирования, а именно передача метаинформации не необходимо использовать большие вычислительные мощности, что отбрасывает необходимость в использовании языков, направленных на это, например, C++ и т.п. Так же существенным плюсом в выборе данного языка является тот факт, что он хорошо адаптирован для межсетевого взаимодействия.

Интерфейс библиотека PyQt5, так как она поддерживает работу на различных операционных системах, включая Windows, macOS и Linux. К тому же библиотека предоставляет широкий спектр инструментов для создания сложных пользовательских интерфейсов, включая возможности работы с виджетами, стилями, событиями и т.п.

Задачи, которые необходимо решить

Работу по ВКР, можно разбить на следующие подзадачи:

1. Изучение строения выбранных СУБД
    a. Изучение особенности строения
    b. Определение значимой для этой СУБД метаинформации
2. Проектирование интерфейса
    a. В зависимости от получившегося по итогам предыдущего пункта данным спроектировать интерфейс для приложения
3. Реализация получения метаинформации из каждой из выбранных СУБД
4. Реализация интерфейса и внутренней логики работы приложения
5. Определить и добавить в приложение дополнительную метаинформацию, которую необходимо отображать

Ожидаемые результаты

В результате выполнения ВКР ожидается получение работающего приложения, которое на основе полученных баз данных (БД) строит локальную модель по 3 СУБД (MongoDB, Cassandra и Neo4j) и метаинформацию по всем БД, которая показывает следующие:

- Входящие таблицы и связи между ними
- Таблицы индексов
- Нагрузка БД
- Настройки БД (в зависимости от СУБД)
- Нагрузка системы
- И прочие данные, которые определятся позднее

Список используемой литературы

1. [Электронный ресурс] – 2023 г. – Режим доступа: https://en.wikipedia.org/wiki/Multi-model_database, свободный.
2. Survey of graph database models / Renzo Angles, Claudio Gutierrez; — ACM Computing Surveys, vol. 40, 2008. — 1–39 с.
3. An Approach for Implementing Online Analytical Processing Systems under ColumnFamily Databases / Abdelhak Khalil and Mustapha Belaissaoui; — IAENG International Journal of Applied Mathematics, vol. 53, 2023. — 31–39 с.
4. [Электронный ресурс] – 2023 г. – Режим доступа: https://www.mongodb.com/document-databases, свободный.
5. MongoDB: The Definitive Guide / Shannon Bradshaw, Eoin Brazil and Kristina Chodorow. — Boston: O'Reilly Media, Inc., 2019. — 511 с.
6. MongoDB Sharded Cluster. – Текст. Изображение : электронные // Национальная библиотека им. Н.Э. Баумана: [сайт]. – URL: https://ru.bmstu.wiki/MongoDB_Sharded_Cluster
7. NoSQL: новая методология разработки нереляционных баз данных / Фаулер, Мартин, Садаладж, Прамодкумар Дж.; Пер. с англ. - М.: ООО "И.Д. Вильямс", 2013г.