УДК 006.72 **DOI** 10.15622/sp.2020.19.4.5

Ю.А. Шичкина, В.М. Ха

МЕТОД СОЗДАНИЯ КОЛЛЕКЦИЙ СО ВЛОЖЕННЫМИ ДОКУМЕНТАМИ ДЛЯ БАЗ ДАННЫХ ТИПА КЛЮЧ-ДОКУМЕНТ С УЧЕТОМ ВЫПОЛНЯЕМЫХ ЗАПРОСОВ

Шичкина Ю.А, Ха В.М. **Метод создания коллекций со вложенными документами для баз данных типа ключ-документ с учетом выполняемых запросов.**

Аннотация. В последние десятилетия все большую популярность набирают NoSQL базы данных, и все чаще разработчикам и администраторам таких баз по той или иной причине приходится решать задачу миграции баз данных из реляционной модели в модель NoSQL, например документно-ориентированную базу данных MongoDB. Описывается подход к такой миграции данных на основе теории множеств. Предлагаются правила для определения совокупности коллекций со вложенными документами NoSQL базы данных типа ключ-документ, оптимальной по времени выполнения поисковых запросов. Оптимизация числа коллекций и их структуры проводится с учетом атрибутов объектов базы данных, участвующих в поисковых запросах. Исходными данными являются свойства объектов (атрибуты, связи между атрибутами), информация о которых хранится в базе данных, и свойства запросов, которые наиболее часто выполняются или скорость их выполнения максимальна. В правилах учитываются основные типы связей (1-1, 1-М, М-М), свойственные реляционной модели. Рассматриваемая совокупность правил является дополнением к методу создания коллекций без вложенных документов. Также приводится методика для определения, в каких случаях какие методы надо использовать, чтобы сделать работу с базами данных более эффективной. В заключении приведены результаты тестирования предлагаемого метода на базах данных с различными начальными схемами. Результаты экспериментов показывают, что предлагаемый метод помимо сокращения времени выполнения запросов позволяет также значительно сократить объем памяти, необходимый для хранения данных в новой базе данных.

Ключевые слова: NoSQL, запрос, коллекция, ключ-документ, трансляция данных, формат данных, создание структуры базы данных, встроенные документы.

1. Введение. Быстрый доступ к точной информации является серьезной проблемой, с которой сталкиваются организации. Актуальность решения проблемы скорости доступа к данным продиктована повсеместной цифровизацией и внедрением систем искусственного интеллекта. Все больше объектов вокруг человека оснащается датчиками, генерирующими данные для интеллектуальной обработки с огромной частотой, и поэтому скорость обработки этих данных должна быть сравнимой со скоростью получения на их основе знаний и умных решений в соответствии с растущими требованиями современных «умных» систем.

Реляционные базы данных (РБД) являются самыми распространенными формами хранения данных, но кроме них существуют базы данных NoSQL и NewSQL, которые также набирают все большую популярность, что подтверждают различные рейтинги (рис. 1). И эти но-

вые формы баз данных сегодня использует все больше компаний, среди которых такие гиганты, как «Google» или «Amazon» [1].

359 systems in ranking, August 2020

Rank Aug 2020	Rank Jul 2020	Rank Aug 2019	DBMS	Database Model
1.	1.	1.	Oracle	Relational, Multi-model
2.	2.	2.	MySQL	Relational, Multi-model
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model
4.	4.	4.	PostgreSQL	Relational, Multi-model
5.	5.	5.	MongoDB	Document, Multi-model
6.	6.	6.	IBM Db2	Relational, Multi-model
7.	↑ 8.	↑ 8.	Redis	Key-value, Multi-model
8.	↓ 7.	↓ 7.	Elasticsearch	Search engine, Multi-model
9.	9.	↑ 11.	SQLite	Relational
10.	↑ 11.	↓ 9.	Microsoft Access	Relational

Рис. 1. Рейтинг систем управления базами данных [2]

Обо всех этих системах написано немало учебников, статей и им посвящено много различных ознакомительных и профессиональных сайтов и форумов в сети Интернет [3-5].

В [6] описаны пятнадцать категорий баз данных NoSQL, некоторые принципы и примеры для выбора подходящей базы для разных отраслей. В [7] анализируется и сравнивается реализация модели согласованности в пяти популярных базах данных NoSQL: Redis, Cassandra, MongoDB, Neo4j и OrientDB.

Принято считать, что реляционные системы управления базами данных (СУБД) позволяют создавать довольно сложные системы данных, а NoSQL базы данных и СУБД не подразумевают внутренних связей. NoSQL не основываются на одной модели, а каждая база данных в зависимости от целей использует различные модели. Однако проведенный обзор прикладных систем, которые построены на основе NoSQL баз данных, позволяет сделать вывод, что многие разработчики для построения сложных систем данных по тем или иным причинам сегодня выбирают не только реляционные базы данных. Стоит также отметить, что если теория проектирования реляционных баз данных, основанная на функциональных зависимостях и теоремах о нормализации реляционных схем, существует с 70-х годов [8], то для NoSQL баз данных такой теории пока нет, но есть много исследований в этой области.

Описывается новый формализованный метод построения NoSQL базы данных типа ключ-документ с учетом структуры запросов, планируемых для выполнения запросов к базе данных. Исследования показали, что этот метод является основополагающим для преобразования баз данных любых форматов, однако в данной статье демонстрируется его действие только на базах данных типа ключ-документ. Эти базы востребованы и хорошо описаны в большом числе публикаций, например [9-12].

2. Обзор связанных работ. Надо отметить, что вопросам преобразования схемы реляционной базы данных посвящалось много исследований на протяжении последних тридцати лет [13, 14]. Но как бы ни была усовершенствована схема реляционной базы данных, она имеет свои ограничения, которые и способствовали развитию иных форм баз данных, таких как ключ-значение и их разновидностей [9-12] или графовых баз данных [15].

В [16] предложен способ переноса реляционных баз данных в MongoDB путем преобразования таблиц в файлы CSV и импорта преобразованных файлов с помощью встроенной команды MongoDB. Однако данный способ только непосредственно переводит таблицы в коллекции и не учитывает связи между ними. Проведенное тестирование подтверждает, что MongoDB значительно хуже работает при выполнении запросов, затрагивающих несколько коллекций. В данном исследовании мы решаем эту проблему.

В [17] также разработали подход к переносу данных из реляционных баз данных в MongoDB, состоящий из трех этапов: извлечение данных из исходной базы данных, преобразование данных и перенос преобразованных данных в целевую базу данных. К сожалению, этот подход не затрагивает вопросов зависимости производительности от схем баз данных и связей между объектами. В нашем подходе мы учитываем связи и показываем, что от этого производительность запросов выигрывает.

Авторы [18] предложили модель преобразования реляционной схемы в схему базы данных NoSQL на основе структуры данных и запросов к данным. Эта модель имеет трехфазную структуру: описание структуры реляционной базы данных и требований к запросам данных; запросно-ориентированное моделирование данных для базы данных NoSQL; запросно-ориентированное описание схемы базы данных NoSQL. Но в этой модели отсутствует учет зависимостей между объектами базы данных при переходе от реляционной базы данных к NoSQL базе данных. Новая структура NoSQL базы данных основывается только на метаданных об объектах и запросах, а это неизбежно

приведет к потере универсальности запросов. В нашем подходе мы показываем, как связи между объектами влияют на структуру документа со встроенными документами.

В [19] предложили подход к миграции данных из реляционной базы данных в NoSQL базу данных на основе методов реляционной алгебры. В этом исследовании авторы предлагают адаптировать теорию и методы реляционной алгебры к разработке схемы базы данных MongoDB. Для этих же целей мы применяем теорию множеств, которая позволяет построить формализованные правила перевода реляционной БД в БД MongoDB.

В [20] авторы предлагают шесть правил для перевода реляционной базы данных в базу данных NoSQL трех типов (Column-Based, Document-Based и Graph-Based). Каждое правило связано с типом связи между таблицами (1-1, 1-M, M-M) или со специальной операцией, проводимой в одной из таблиц (например, агрегирования). При этом во всех трех типах рассматриваются подвиды NoSQL базы данных. Это работа, которая демонстрирует, как легко можно перевести реляционную базу данных в NoSQL, если не учитывать структуру запросов. Но проблема в том, что NoSQL базы данных являются ориентированными на определенные запросы. И те запросы, которые обращаются к нескольким таблицам в реляционной базе данных будут очень медленно работать в NoSQL базе данных при непосредственном переводе реляционной базы данных в NoSQL. Авторы решают эту проблему путем создания единой коллекции в БД NoSQL. Однако это не очень хорошее решение – соединять все таблицы в одну коллекцию, так как возрастает объем данных и возникают проблемы с памятью. В конце данной статьи мы на основе результатов тестирования показываем, что предлагаемый подход, который основан на построении определенного числа коллекций, лучше, чем подход, который основан на создании единой коллекции.

Главная проблема – отсутствие в NoSQL базах данных операций соединения множеств данных (коллекций, таблиц, семейств и т.п.). Эту проблему решают в [21] с помощью предлагаемого ими подхода к выполнению операций соединения на уровне приложений. Они реализуют свой подход в MongoDB. Их подход хорош тем, что позволяет сохранять модели данных. Однако этот подход не будет эффективным в случае консолидации данных, когда в разных моделях так или иначе окажутся одинаковые данные и, как следствие, при консолидации все равно будет необходима модификация структуры.

В другом случае [22, 23] описывается программная надстройка NoSQLayer, состоящая из двух модулей: модуль переноса данных, ко-

торый идентифицирует и анализирует схему NoSQL базы данных и преобразует ее в схему реляционной базы данных; и модуль отображения данных, который позволяет пользователям создавать запросы на SQL, при этом выбирая данные из базы данных NoSQL. Эта программа не предназначена для миграции данных и многие запросы работают медленнее, чем при непосредственном обращении к базам данных NoSQL на встроенном языке запросов [24]. В предлагаемом подходе мы оптимизируем процесс обработки запроса, используя внутренний язык запросов.

В [25] предоставлена технология, которая позволяет переводить системы реляционных баз данных на документно-ориентированные базы данных. Эта технология состоит из двух этапов: описание данных существующей реляционной базы данных и перенос в базу данных NoSQL. Эта технология не предусматривает оптимизацию схемы NoSQL к выполняемым запросам.

Согласно [26] отношения в документно-ориентированной модели базы данных могут быть представлены в форме встроенных документов и связей между этими документами. Однако, во-первых, встроенные документы могут использоваться для ограниченного объема данных, а во-вторых, остается проблемой определение формы встраивания. Исследования, описанные в [26], это подтверждают.

В [27] авторы применили традиционные правила теории нормализации реляционной схемы базы данных (теоремы о нормальных формах: вторая и третья нормальные формы (2NF, 3NF) и нормальная форма Бойса – Кодда (BCNF)) для разработки схемы MongoDB. Например, если существует функциональная зависимость между двумя атрибутам, то оба атрибута данных преобразуются в один элемент данных в MongoDB. Такой же принцип применяются для частичной и транзитивной зависимостей. Тестирование показало, что в результате применения такого подхода реально повысить производительность запросов для связанных объектов базы данных. Однако тестирование проводилось на небольшой схеме. Подход не учитывает наличие связности отношений типа многие-ко-многим, первичные ключи и внешние ключи. В данной статье показывается, как можно учитывать связность отношений такого типа.

Анализ русскоязычных публикаций показал, что большая часть научных работ, связанных с нереляционными базами данных, лежит в прикладной плоскости. Всего по запросу с ключевыми словами «коллекция, MongoDB» в научной электронной библиотеке «eLIBRARY.RU» были найдены 261 публикация из 34844888, в поисковой системе по полным текстам научных публикаций «Академия

Google» было найдено 370 русскоязычных публикаций по данным ключевым словам. Среди найденных работ следует выделить интересные работы: [28], в которой описано приложение, позволяющее преобразовывать SQL-запросы по чётким и нечётким данным в запросы на языке MongoQL; и [29], в которой описана общая концепция преобразования реляционной базы данных в формат MongoDB без учета запросов к базе данных.

Таким образом, анализ работ в области оптимизации структуры баз данных NoSQL, в частности документных баз данных, показал, что есть много различных подходов к преобразованию баз данных, но нет комплексного подхода, который бы одновременно учитывал типы связей между объектами и структуру запросов к объектам базы данных и позволял бы строить такое количество коллекций и такой внутренней структуры, чтобы запросы на выбор данных из документной базы данных выполнялись максимально быстро.

В данной статье метод и все примеры будут рассматриваться применительно к классическим реляционным базам данным и MongoDB.

MongoDB является одной из самых популярных баз данных. В литературе описано много хороших методологий по ее применению для различных практических задач. Например, в [30] приведена технология шардинга базы данных и модель управления крупномасштабными данными; в [31] описан метод построения слепых запросов к хранилищам документов JSON на примере MongoDB; метод обработки запросов в пространственно-временном диапазоне на основе индекса на примере базы данных большого объема MongoDB описан в [32]. В работах [33, 34] рассмотрены преимущества MongoDB перед известными реляционными базами данных, представлены результаты сравнительного анализа реляционных и нереляционных баз данных. Много работ посвящено описанию применения системы MongoDB в различных предметных областях: для создания прототипа трехмерной кадастровой системы и трехмерной визуализации [35]; прогнозирования потребляемой тепловой мощности зданий [36], для контроля потока грунтовых вод и переноса загрязняющих веществ [37], в персонализированной системе мониторинга здравоохранения [38] и, конечно, приложениях IoT [39].

3.Постановка задачи. Трансляция данных между различными источниками данных – необходимый шаг для многих задач интеллектуального анализа данных. Однако различия между методами хранения данных в этих двух формах SQL и NoSQL поднимают много проблем в области трансляции, трансформации и

консолидации данных. Одна из них – соответствие коллекций NoSQL таблицам реляционных баз данных. Для трансляции, трансформации и консолидации баз данных различных типов необходимо также учитывать отсутствие структуры и особенности языка запросов. Если для реляционных баз данных существуют методы формализованного построения отношений на основе заданных свойств объекта и функциональных зависимостей между ними, то для баз данных NoSQL и NewSQL такого формализованного аппарата не существует. И, например, при трансляции реляционной базы данных к формату СУБД MongoDB всегда встает вопрос о том, как этот перевод осуществлять.

При этом возможны следующие варианты изменения формата (перечень не полный):

1) Каждой таблице в реляционной базе данных поставить в соответствие отдельную коллекцию документов в MongoDB.

2) Из всех таблиц реляционной базы данных сделать одну коллекцию документов в MongoDB.

3) Создать такой набор коллекций документов в MongoDB, чтобы они наиболее полно подходили под выполняемые запросы.

Каждый из трех вариантов может быть эффективным при одной схеме БД и неэффективным при другой схеме БД. В [40] показано, как с помощью метода, основанного на теории множеств, можно определить число коллекций для конкретной БД. В данной статье объясняется, как можно усовершенствовать внутреннюю структуру коллекций, основываясь на информации о свойствах объектов в БД, запросов к БД и связях между объектами. Метод, представленный в данной статье, является продолжением метода создания коллекций без встроенных документов, описанного в [40].

4. Подход к применению методов определения коллекций в документной БД. В [40] мы оговаривали, что новая база данных MongoDB создается на основе существующей реляционной базы данных. Мы показали, что применение формализованного подхода, основанного на теории множеств, помогает найти оптимальный набор коллекций в MongoDB на основе информации о запросах, выполняемых к базе данных.

Но что делать, если новая база данных MongoDB создается не на основе реляционной базы данных? Ответ прост: надо использовать информацию об атрибутах объектов, которые планируется хранить в базе данных, применить к этим атрибутам методы реляционной алгебры и создать схему базы данных, удовлетворяющую третьей нормальной форме или нормальной форме Бойса – Кодда.

Принцип применения предлагаемых методов из [40] и данной статьи для разного рода преобразований структуры базы данных приведен на рисунке 2.

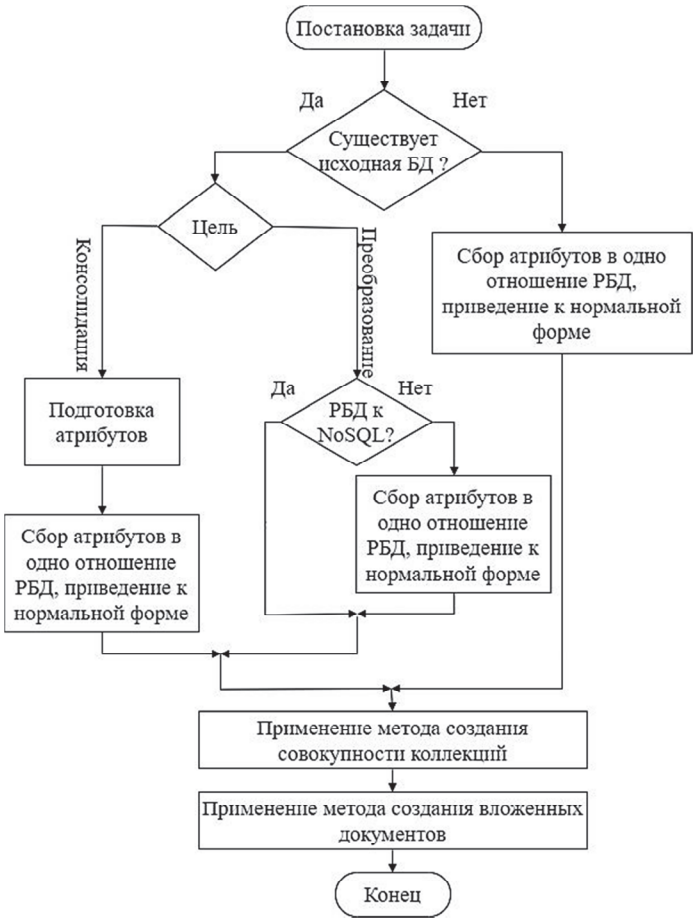


Рис. 2. Применение метода определения коллекций в документных БД

Суть этой методики в том, что независимо от того, происходит ли трансляция реляционной базы данных в документную БД, или консолидация реляционной БД с документной БД, или создание новой документной БД, если конечным результатом должна быть документная БД, то для оптимального по быстродействию выполнения запросов к ней необходимо к набору атрибутов объектов БД последовательно применить два наших метода:

1) Определение оптимального числа коллекций [40].

2) Определение внутренней структуры документов в коллекции: число и порядок встроенных документов. Этот метод описывается ниже.

Для определения необходимости создания вложенных документов лучше, если схема будет приведена изначально к третьей нормальной форме или нормальной форме Бойса – Кодда с помощью реляционной алгебры. Это позволит найти компромисс между быстродействием запросов и минимальным объемом памяти для хранения данных.

Если БД изначально не была реляционной, то ее структуру всегда можно привести к реляционной модели [40]. Нормальная форма реляционной модели – это всего лишь промежуточная форма. Она необходима для установления связей между отношениями, которые впоследствии будут учтены при создании вложенных документов. Таким образом, если исходная БД изначально не была реляционной, то преобразование ее схемы должно пройти по принципу: NoSQL-RDB-NoSQL.

5. Метод определения структуры вложенных документов в документных базах данных. Метод определения числа и порядка встроенных документов основан на типе связей между таблицами реляционной модели. В зависимости от типа связи необходимо применить определенное правило для составления документа в коллекции.

Ниже рассмотрены основные типы связей (1-1, 1-M, M-M) для двух и трех таблиц. Правила для трех таблиц построены на основе правил для двух таблиц. Правила для модели из более чем трех таблиц будут основаны на правилах для двух и трех таблиц, поэтому их рассматривать смысла нет.

Показано, что при применении метода определения числа коллекций из [40], встроенные документы могут быть только в том случае, если между таблицами, атрибуты которых вошли в коллекцию, была связь 1-M. Поэтому для двух таблиц выведено только одно правило. Для трех таблиц имеет значение последовательность связей между таблицами и число главных и второстепенных (связанных с ними), поэтому правил для этого случая больше.

5.1 Определение вложенных документов для двух отношений. Входные данные:

1) Даны два отношения реляционной модели, на основе которых строится схема нереляционной БД типа ключ-документ: T_1 и T_2 :

$$T_1\{T_{11}, T_{12}, ..., T_{1k}\};$$
$$T_2\{T_{21}, T_{22}, ..., T_{2n}\},$$

где T_{ij} – это j -е поле i-го отношения; k – число полей в отношении T_1; n – число полей в отношении T_2.

2) По методу определения коллекций в БД типа ключ-документ получена некоторая коллекция:

$$Q\{T_{11},...,T_{1m},T_{21},...,T_{2r}\}, m \le k, r \le n.$$

3) Отношения T_1 и T_2 имеют связи типа $1-\text{M}$ $\boxed{T_1}_{1-\infty}\boxed{T_2}$.

4) В отношениях определены ключи, принадлежащие данной коллекции:

$$T_1 : T_{11}, T_{12},...,T_{1s1}, \text{ где } 1 \le s1 \le m,$$
$$T_2 : T_{21}, T_{22},...,T_{2s2}, \text{ где } 1 \le s2 \le r.$$

5) Есть совокупность запросов S_2 к коллекции Q:

$$S_{21}\{T_{11},...,T_{1s1},...,T_{1i1},T_{21},...,T_{2s2},...,T_{2j1}\};$$
$$\dots$$
$$S_{2t}\{T_{11},...,T_{1st},...,T_{1it},T_{21},...,T_{2s2},...,T_{2jt}\},$$

и совокупность запросов S_1 к атрибутам отношения T_2 из коллекции Q:

$$S_{11}\{...,T_{2i1},...\},$$
$$\dots$$
$$S_{1r}\{...,T_{2ir},...\}.$$

Запросы во множествах S_1 и S_2 – это запросы только на выбор данных, то есть в терминах SQL запросы вида «*select ... from ... where ...*». В этих запросах могут быть многоуровневые подзапросы или операции агрегирования, группирования, сортировки. В них в терминах SQL могут быть также операции объединения, пересечения и вычитания таблиц, так как предварительно был применен метод [40]. Под запросами не рассматриваются запросы на удаление, модификацию и вставку данных.

6) $T'_2(S_2)$ – это множество всех атрибутов, участвующих в запросах:

$$T'_2(S_2) = \{T_{21},...,T_{2s2},...,T_{2j1}\} \cup ... \cup ,$$

$$\{T_{21},...,T_{2s2},...T_{2jt}\} = \{T_{21},...,T_{2s2},...,T_{2j1},...,T_{2jt}\}.$$

7) Всего запросов S_1 к базе данных равно V_s.

Правило 1. Встроенный документ должен состоять из атрибутов множеств:

$$T'_2 = T'_2(S_2) \cup \bigcup_{i=1}^{V_S} (S_1 - T'_2(S_2))_{S_{1i} - T'_2(S_2) \neq \varnothing \,\&\, S_{1i} - T'_2(S_2) \neq S_{1i}},$$

а новая структура коллекции Q имеет вид:

$$\begin{aligned} Q' \{ & T_{11}, ... T_{1s1}, ..., T_{1m}, T'_2 : \{ T'_2(S_2) \cup \\ & \cup \bigcup_{i=1}^{V_S} (S_{1i} - T'_2(S_2))_{S_{1i} - T'_2(S_2) \neq \varnothing \,\&\, S_{1i} - T'_2(S_2) \neq S_{1i}} \}, \\ & T''_2 : \left\{ \bigcup_{i=1}^{V_S} (S_{1i})_{S_{1i} \cap T'_2(S_2) = \varnothing} \right\} \} \end{aligned} \qquad (1)$$

где T'_2 и T''_2 – это имена новых ключей для встроенных документов.

В данном правиле рассмотрена только связь типа $1-M$. Если схема РБД была изначально приведена к нормальной форме BCNF, 3NF или 4NF, то других связей между отношениями T_1 и T_2 быть не может. Но если по каким-либо причинам, например проведенной денормализации схемы РБД, эти связи существуют, то:

– для связи типа $1-1$ вложенных документов быть не может ввиду однозначности соответствия каждому кортежу из отношения T_1 единственного кортежа из отношения T_2;

– для связи типа $M-M$ вложенные документы строятся по тому же принципу, что и для связи $1-M$.

5.2. Определение вложенных документов для трех отношений с одним главным отношением и двумя подчиненными. Пусть:

1) Даны три отношения реляционной модели, на основе которых строится схема нереляционной БД типа ключ-документ: T_1, T_2 и T_3:

$$T_1\{T_{11}, T_{12}, ..., T_{1k}\};$$

$$T_2\{T_{21}, T_{22}, ..., T_{2n}\};$$

$$T_3\{T_{31}, T_{32}, ..., T_{3m}\}.$$

где T_{ij} – это j-е поле i-го отношения; k – число полей в отношении T_1; n – число полей в отношении T_2; m – число полей в отношении T_3.

2) По методу определения коллекций в БД типа ключ-документ получена некоторая коллекция:

$$Q\{T_{11},...,T_{1k'},T_{21},...,T_{2n'},T_{31},...,T_{3m'}\},\ \ k' \le k, n' \le n, m' \le m.$$

3) Отношения T_1, T_2 и T_3 имеют связи типа $1-\infty$: $\boxed{T_3}_{\infty-1}\boxed{T_1}_{1-\infty}\boxed{T_2}$.

4) В отношениях определены ключи, принадлежащие данной коллекции:

$$T_1 : T_{11}, T_{12}, ..., T_{1s1},\ \text{где}\ 1 \le s1 \le k';$$

$$T_2 : T_{21}, T_{22}, ..., T_{2s2},\ \text{где}\ 1 \le s2 \le n';$$

$$T_3 : T_{31}, T_{32}, ..., T_{3s3}\ \text{где}\ 1 \le s3 \le m'.$$

Правило 2. Если $S_1\{T_{11},...,T_{1s1},...,T_{1i}\}$ – запрос, который обращается только к атрибутам одного отношения, например T_1, то из пункта 5.1. следует, что вложенных документов в данной коллекции быть не может.

Правило 3. Если $S_2\{T_{11},....,T_{1s1},...,T_{1i},T_{21},...,T_{2s2},...,T_{2j}\}$ – запрос, который обращается к атрибутам двух отношений T_1, T_2 или T_1, T_3, то в пункте 5.1. показано, что новая структура коллекции Q примет вид (1).

Правило 4. Если $S_3\{T_{11},...,T_{1s1},....,T_{1i},T_{21},...,T_{2s2},.....,T_{2j},T_{31},.....,T_{3s3},...,T_{3r}\}$ – запрос, который обращается к атрибутам всех трех отношений T_1, T_2 и T_3, то новая структура коллекции Q примет вид:

$$\begin{aligned}
&Q\{T_{11},....,T_{1m},T'_2\{T'_2(S_2)\cup\cup_{i=1}^{VS}(S_{1i}- \\
&-T'_2(S_2))_{S_{1i}-T'_2(S_2)\neq\varnothing \& S_{1i}-T'_2(S_2)\neq S_{1i}}\}, \\
&T"2:\{\cup_{i=1}^{VS}(S_{1i})_{S_{1i}\cap T'_2(S_2)=\varnothing}\}, T'_3:\{T'_3(S_3)\cup \\
&\cup\cup_{i=1}^{VS}(S_{1i}-T'_3(S_3))_{S_{1i}-T'_3(S_3)\neq\varnothing \& S_{1i}-T'_3(S_3)\neq S_{1i}}\}, \\
&T"_3:\{\cup_{i=1}^{VS}(S_{1i})_{S_{1i}\cap T'_3(S_3)=\varnothing}\}\}
\end{aligned} \qquad (2)$$

где T'_2, T'_3 – это имена новых ключей для вложенных документов; атрибуты $T'_2(S_3)$ – это все атрибуты отношения T_2, входящие в за-

просы типа S_3 к коллекции Q; атрибуты $T'_3(S_3)$ – это все атрибуты отношения T_3, входящие в запросы типа S_3 к коллекции Q; V_S – количество запросов типа S_1; T''_2 – множество атрибутов, которые не вошли в T'_2 из всех атрибутов отношения T_2, входящих в коллекцию Q; T''_3 – множество атрибутов, которые не вошли в T'_3 из всех атрибутов отношения T_3, входящих в коллекцию Q.

Надо заметить, что этот случай является обобщением Правила 3 с двух отношений со связью $1-\infty$ на две пары отношений со связью $1-\infty$.

5.3. Определение вложенных документов для трех отношений с двумя главными отношениями и двумя подчиненными. Пусть:

1) Даны три отношения реляционной модели, на основе которых строится схема нереляционной БД типа ключ-документ: T_1, T_2 и T_3:

$$T_1\{T_{11}, T_{12}, ..., T_{1k}\};$$

$$T_2\{T_{21}, T_{22}, ..., T_{2n}\};$$

$$T_3\{T_{31}, T_{32}, ..., T_{3m}\},$$

где T_{ij} – это j-е поле i-го отношения; k – число полей в отношении T_1; n – число полей в отношении T_2; m – число полей в отношении T_3.

2) По методу определения коллекций в БД типа ключ-документ получена некоторая коллекция:

$$Q\{T_{11}, ..., T_{1k'}, T_{21}, ..., T_{2n'}, T_{31}, ..., T_{3m'}\}, k' \le k, n' \le n, m' \le m.$$

3) Отношения T_1, T_2 и T_3 имеют связи типа $1-M$: $\boxed{T_1}_{1-\infty}\boxed{T_2}_{1-\infty}\boxed{T_3}$.

4) В отношениях определены ключи, принадлежащие данной коллекции:

$$T_1 : T_{11}, T_{12}, ..., T_{1s1}, \text{ где } 1 \le s1 \le k'.$$

$$T_2 : T_{21}, T_{22}, ..., T_{2s2}, \text{ где } 1 \le s2 \le n'.$$

$$T_3 : T_{31}, T_{32}, ..., T_{3s3}, \text{ где } 1 \le s3 \le m'.$$

Рассмотрим возможные варианты запросов к коллекции Q:

a) $S_1\{T_{11},...,T_{1s1},...,T_{1i}\}$ – запрос, который обращается только к атрибутам одного отношения, например T_1;

b) $S_2\{T_{11},...,T_{1s1},...,T_{1i},T_{21},...,T_{2s2},...,T_{2j}\}$ – запрос, который обращается к атрибутам двух отношений: T_1, T_2 или T_2, T_3;

c) $S_3\{T_{11},...,T_{1s1},...,T_{1i},T_{21},...,T_{2s2},....,T_{2j},T_{31},...,T_{3s3},....,T_{3r}\}$ – запрос, который обращается к атрибутам всех трех отношений: T_1, T_2 и T_3;

d) $S_4\{T_{11},...,T_{1s1},...,T_{1i},T_{31},...,T_{3s3},...,T_{3r}\}$ – запрос, который обращается к атрибутам двух отношений: T_2 и T_3.

Случай a). Этот случай рассмотрен в пункте 5.1 при определении вложенных документов для коллекции, построенной из атрибутов двух отношений, и показано, что *вложенных документов в данной коллекции быть не может.*

Случай b). Этот случай рассмотрен в пункте 5.1 при определении вложенных документов для коллекции, построенной из атрибутов двух отношений, и показано, что новая структура коллекции Q примет вид (1).

Случай c). Этот случай является обобщением *случая b)* с двух отношений на три отношения. В данном случае новая структура коллекции Q примет вид:

$$\begin{aligned} &Q'\{T_{11},...,T_{1m},T'_2:\{T'_2(S_3)\cup\bigcup_{i=1}^{V_S}(S_{1i}\cap(S_{1i}-T'_2(S_3)))\};\\ &T'_3:\{T'_3(S_3)\cup\bigcup_{i=1}^{V_S}(S_{1i}\cap(S_{1i}-T'_3(S_3)))\},T''_2,T''_3\}, \end{aligned} \tag{3}$$

где T'_2, T'_3 – это имена новых ключей для вложенных документов; атрибуты $T'_2(S_3)$ – это все атрибуты отношения T_2, входящие в запросы типа S_3 к коллекции Q; атрибуты $T'_3(S_3)$ – это все атрибуты отношения T_3, входящие в запросы типа S_3 к коллекции Q; V_S – количество запросов типа S_1; T''_2 – множество атрибутов, которые не вошли в T'_2 из всех атрибутов отношения T_2, входящих в коллекцию Q; T''_3 – множество атрибутов, которые не вошли в T'_2 из всех атрибутов отношения T_3, входящих в коллекцию Q.

Случай d). Так как запрос к атрибутам отношений T_1 и T_3 может быть выполнен только через атрибуты отношения T_2, то этот случай в результате сводится к *случаю c)*.

Если коллекция Q построена более чем для трех отношений, то вложенные документы строятся так же, как и для трех отношений.

6. Тестирование эффективности методики определения структуры вложенных документов в БД типа ключ-документ. Тестирование проводилось на персональном компьютере Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz, с операционной системой Windows, MySQL Community Server Версия 8.0.17, MongoDB Community Server v4.0.5.

Тестирование было выполнено для трех реляционных баз данных с разными схемами (рис. 3-5).

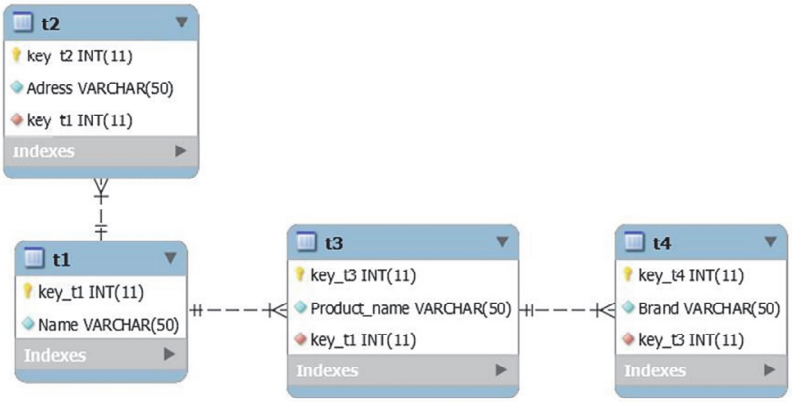


Рис. 3. Вариант I схемы тестируемой базы данных

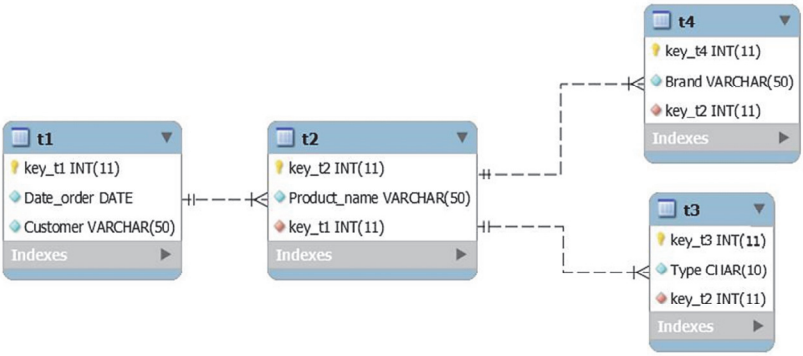


Рис. 4. Вариант II схемы тестируемой базы данных

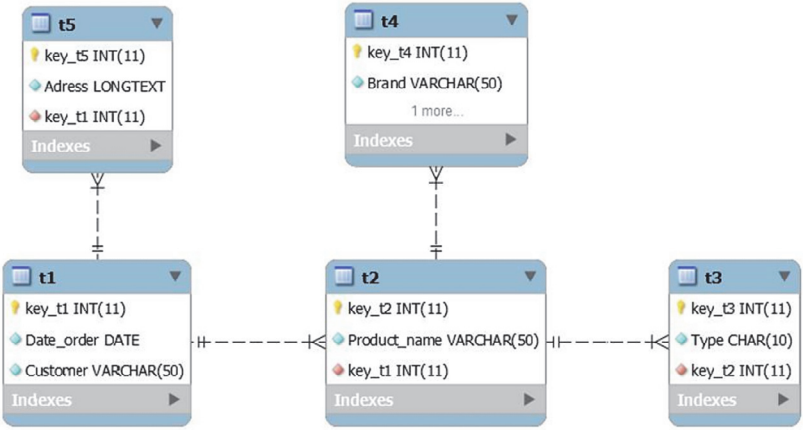


Рис. 5. Вариант III схемы тестируемой базы данных

Объемы тестируемых баз данных были следующие:

– вариант I**:** таблица T1 – 15000 записей, T2 – 50 000 записей, T3 – 110 000 записей, T4 – 200 000 записей;

– вариант II**:** таблица T1 – 50 000 записей, T2 – 120 000 записей, T3 – 350 000 записей, T4 – 300 000 записей;

– вариант III**:** таблица T1 – 60 000 записей, T2 – 170 000 записей, T3 – 360 000 записей, T4 – 400 000 записей, T5 –100 000 записей,.

Для каждой из схем баз данных, представленных на рисунках 4-6, были построены по две базы данных MongoDB: база данных, которая состоит из единой коллекции, объединяющей в себе все данные из всех таблиц реляционной базы данных без вложенности документов, и база данных, которая также состоит из единой коллекции, построенной по предлагаемому в данной статье методу со вложенными документами. В тестировании применяется совокупность описанных правил к базе данных MongoDB без предварительной оптимизации ее структуры по числу коллекций с помощью метода, приведенного в [40]. Это сделано специально с целью демонстрации именно данного подхода, но на практике эти методы рекомендуется применять последовательно. Сначала метод оптимизации по числу коллекций, затем метод оптимизации структуры коллекций, описанный в данной статье.

Фрагменты коллекций без вложенных документов представлены на рисунке 6:

Базы данных, построенные в соответствии с правилами 1-3, описанными в пункте 5, имеют следующие коллекции:

– вариант I:

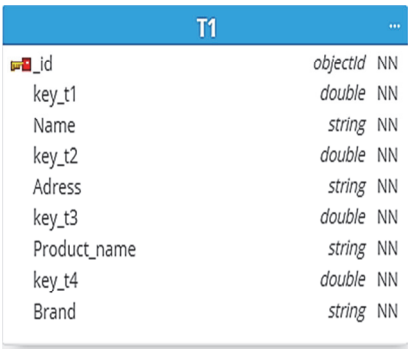
Q'{key_t1,Name,T2_of_T1:{key_t2,Adress},T3_of_T1:{key_t3,Product_name, T4_of_T3 :{key_t4, Brand}}};

– вариант II:

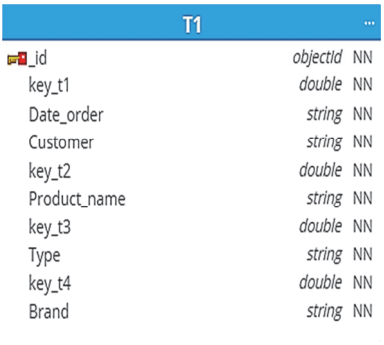
Q'{key_t1,Date_order, Customer ,T2_of_T1 :{ key_t2, Product_name, T3_of_T2 :{ key_t3, Type}, T4_of_T2 :{key_t4, Brand}}};

– вариант III:

Q'{key_t1,Date_order, Customer , T5_of_T1: {key_t5, Adress},T2_of_T1 : { key_t2, Product_name, T3_of_T2 :{ key_t3, Type}, T4_of_T2 :{key_t4,Brand}}}.



а) Вариант I

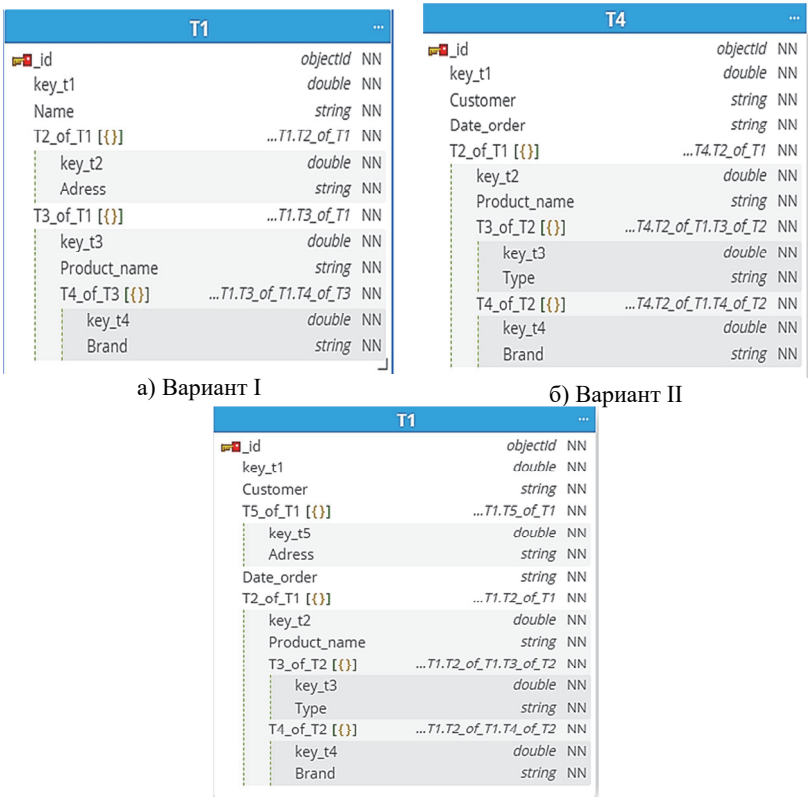


б) Вариант II

в) Вариант III

Рис. 6. Единая коллекция MongoDB без вложенных документов

Структуры коллекций, полученных для вариантов I-III баз данных, показаны на рисунке 7.



а) Вариант I

б) Вариант II

в) Вариант III

Рис. 7. Новые коллекции со вложенными документами

Для оценки эффективности применения предлагаемого подхода для баз данных с различными типами связей было проведено тестирование 8 разных запросов для каждого из вариантов полученных баз данных. Каждый запрос выполнялся 25 раз. Все 25 повторов каждого запроса выполнялись последовательно. Представленное время является усредненным значением (в миллисекундах).

Для чистоты эксперимента при тестировании не использовались индексы и было отключено кэширование. Исследования по дополнению метода информацией об индексах проводятся в настоящий момент и будут представлены в другой статье.

Учитывая, что функция кэширования в обеих СУБД была отключена, разброс по времени выполнения между 25 запусками для каждого запроса был небольшим. Среднее квадратическое отклонение значений времени выполнения запроса в выборке из 25 значений для каждого исследуемого запроса принадлежит интервалу [0.002; 0,004].

Структура запросов различалась по числу таблиц, к которым велось обращение, и наличию условий для поиска. Особенности запросов:

– запросы 1, 2 обращаются к двум таблицам. Содержат операцию естественного соединения таблиц. Таблицы отличаются в запросах. Коды запросов:

select name, address from t1, t2 where t1.key_1 = t2.key_1;
select product_name, brand from t2, t4 where t2.key_t2 = t4.key_t2;

– запрос 3 обращается к трем таблицам. Содержит операцию естественного соединения таблиц. Код запроса:

select name, product, brand from t1,t3,t4 where t1.key_t1 = t3.key_t1 and t3.key_t3 = t4.key_t3;

– запросы 4, 5 обращаются к двум таблицам. Содержат условие на выбор данных и операцию естественного соединения таблиц. Таблицы и операции отличаются в запросах. Коды запросов:

select adress from t1,t2 where t1.key_t1 = t2.key_t1 and name = 'charity dickerson';
select t3.product_name from t3,t4 where t3.key_t3 = t4.key_t3 and t4.brand = 'sureraquistor international corp.';

– запросы 6, 7 обращаются к трем таблицам. Содержат условие на выбор данных и операцию естественного соединения таблиц. Таблицы и операции отличаются в запросах. Коды запросов:

select Product_name, Brand from t1,t3,t4 where t1.key_t1 = t3.key_t1 and t3.key_t3 = t4.key_t3 and Name = 'Mindy Garcia'
select Name, Product_name from t1,t3,t2 where t1.key_t1 = t2.key_t1 and t1.key_t1 = t3.key_t1 and Adress = '662 South Second Drive';

– запрос 8 обращается к четырем таблицам. Содержит условие на выбор данных и операцию естественного соединения таблиц. Код запросов:

select Adress, Product_name, Brand from t1,t2,t3,t4 where t1.key_t1 = t2.key_t1 and t1.key_t1 = t3.key_t1 and t3.key_t3 = t4.key_t3 and Name = 'Sonia Chang'.

На рисунке 8 приведен график времени выполнения тестируемых запросов.

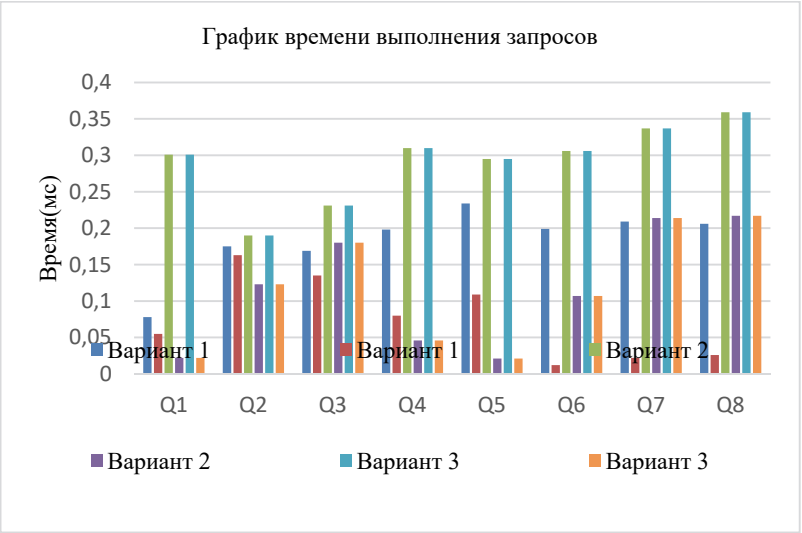


Рис. 8. Графики результатов выполнения запросов

Из диаграмм на рисунке 8 видно, что в зависимости от структуры запросов разность во времени их выполнения к обеим базам данных для всех вариантов I-III может быть как малой, так и очень большой. Но все запросы со вложенными документами выполнялись, как правило, намного быстрее. Это можно объяснить тем, что объем коллекции при использовании вложенных документов гораздо меньше, чем в случае построения коллекции без вложенных документов. Например, для варианта I при трансляции данных с БД MySQL в MongoDB новая коллекция со вложенными документами состояла из 15000 документов и имела объем ≈ 21,2 Мб, а коллекция без вложенных документов состояла из 597686 документов и имела объем ≈ 112,3 Мб. Именно поэтому очень важно оценить, в каких коллекциях должны быть вложенные документы, а в каких нет. Эти примеры показывают эффективность применения описанных в пунктах 3-4 подходов и правил построения коллекций с учетом связей между объектами в базе данных и структуры запросов, которые к ним выполняются.

7. Заключение. В результате проведенных исследований был разработан формализованный метод, основанный на теории множеств и позволяющий автоматизировать процесс построения документной базы дан-

ных по заданной совокупности свойств объектов и связей между ними. Учет структуры запросов к базе данных позволяет определить число и состав коллекций для СУБД MongoDB таким образом, чтобы можно было избежать операций объединения коллекций. Учет связей между объектами позволяет оценить необходимость создания встроенных документов.

Предлагаемый метод был тщательно протестирован на документных базах данных различной структуры с разными связями между таблицами. Показаны только самые характерные примеры со встроенными документами. В каждом тестировании замерялось среднее время выполнения запросов, часть из них отражена в разделе 6. Проведенный анализ результатов выполнения запросов показал, что организация базы данных с применением предлагаемого метода позволяет ускорить выполнение запросов на 10-50% по сравнению с другими формами организации баз данных. Величина ускорения запросов зависит от двух обстоятельств: начальной схемы базы данных (до применения к этой схеме метода, описанного в данной статье) и структуры запроса.

В целом предлагаемый метод может применяться для:

1) Создания эффективной структуры новой базы данных типа ключ-документ.

2) Трансляции данных из реляционной базы данных в базу данных типа ключ-документ.

3) Трансляции данных из произвольной базы данных в базу данных типа ключ-документ.

4) Консолидации баз данных.

Есть и другие возможности для применения данного метода, например создание временных коллекций.

И последнее, что стоит отметить: в новую структуру данные переносятся в последнюю очередь, и делается это, как правило, с помощью программных надстроек или с помощью алгебры кортежей, что частично описано в [41].

Литература

1. *Rocha L. et al.* A Framework for Migrating Relational Datasets to NoSQL // Procedia Computer Science. 2015. vol. 51. pp. 2593–2602.
2. Рейтинг баз данных [DB-Engines Ranking]. URL: https://db-engines.com/en/ranking (дата обращения 15.07.2020).
3. *Jose B., Abraham S.* Performance analysis of NoSQL and relational databases with MongoDB and MySQL // Materials Today: Proceedings. 2020. vol. 24. pp. 2036–2043.
4. *Čerešňák R., Kvet M.* Comparison of query performance in relational a non-relation database // Transportation Research Procedia. 2019. vol. 40. pp. 170–177.
5. *Diogo M., Cabral B., Bernardino J.* Consistency Models of NoSQL Databases // Future Internet. 2019. vol. 11(2). pp. 43.
6. *Chen J., Lee W.* An Introduction of NoSQL Databases Based on Their Categories and Application Industries // Algorithms. 2019. vol. 12(5). pp. 106.

7. *Diogo M., Cabral B., Bernardino J.* Consistency Models of NoSQL Databases // Future Internet. 2019. vol. 11(2). pp. 43.
8. *Li W., Clifton C., Liu S.* Database Integration Using Neural Networks: Implementation and Experiences // Knowl. Inf. Syst. 2000. vol. 2. no. 1. pp. 73–96.
9. *Storey V.C., Song* I.Y. Big data technologies and Management: What conceptual modeling can do // Data Knowl. Eng. 2017. vol. 108. pp. 50–67.
10. *Corbellini A. et al.* S. Persisting big-data: The NoSQL landscape // Inf. Syst. 2017. vol. 63. pp. 1–23.
11. *Makris A., Tserpes K., Andronikou V., Anagnostopoulos D.* A classification of NoSQL data stores based on key design characteristics // Procedia Computer Science. 2016. vol. 97. pp. 94–103.
12. *Atzeni P., Bugiotti F., Rossi L.* Uniform access to NoSQL systems // Inf. Syst. 2014. vol. 43. pp. 117–133.
13. *Pardede E., Rahayu J.W., Taniar D.* Mapping Methods and Query for Aggregation and Association Relationship in Object-Relational Database using Collection // Proceedings of the 2004 IEEE International Conference on Information Technology: Coding and Computing (ITCC). 2004. pp. 539–543.
14. *Shichkina Y., Kupriyanov M., Shevsky V*. The Application of Graph Theory and Adjacency Lists to Create Parallel Queries to Relational Databases // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2018. vol. 10963 LNCS. pp. 61–77.
15. *Robinson I., Webber J., Eifrem E.* Graph Databases: New Opportunities for Connected Data // O'Reilly Media, Inc. 2015. 238 p.
16. *Chickerur S.* Comparison of Relational Database with Document-Oriented Database (MongoDB) for Big Data Applications // 8th International Conference on Advanced Software Engineering and Its Applications ASEA. 2015. pp. 41–47.
17. *Hanine M., Bendarag A., Boutkhoum O.* Data Migration Methodology from Relational to NoSQL Databases // World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering. 2015. vol. 9. no. 12. pp. 2566–2570.
18. *Li X., Ma Z., Chen H.* QODM : A Query-Oriented Data Modeling Approach for NoSQL Databases // Advanced Research and Technology in Industry Applications (WARTIA). 2014. pp. 338–345.
19. *Zhao G., Huang W., Liang S., Tang Y.* Modeling MongoDB with Relational Model // 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies (EIDWT). 2013. pp. 115–121.
20. *Alotaibi O., Pardede E.* Transformation of Schema from Relational Database (RDB) to NoSQL Databases // Data. 2019. vol. 4(4). pp. 148.
21. *Celesti A., Fazio M., Villari M.* A Study on Join Operations in MongoDB Preserving Collections Data Models for Future Internet Applications // Future Internet. 2019. vol. 11(4). pp. 83.
22. *Rocha L., Vale F., Cirilo E*. A Framework for Migrating Relational Datasets to NoSQL // Procedia Computer Science. 2015. vol. 51. pp. 2593–2602.
23. *Liang D., Lin Y., Ding G*. Mid-model Design Used in Model Transition and Data Migration between Relational Databases and NoSQL Databases // IEEE Int. Conf. Smart City. 2015. pp. 866–869.
24. *Hamid S., Rezapour M., Moradi M., Ghadiri N*. Performance evaluation of SQL and MongoDB databases for big e-commerce data // IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing. 2015. pp.1–7.
25. *Karnitis G., Arnicans G.* Migration of Relational Database to Document-Oriented Database : Structure Denormalization and Data Transformation // 7th International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN). 2015. pp. 113–118.

26. *Mason R.T.* NoSQL Databases and Data Modeling Techniques for a Document-oriented NoSQL Database // Computer Science. 2015. pp. 259–268.
27. *Gu Y., Shen S., Wang J., Kim J.* Application of NoSQL Database MongoDB // IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW). 2015. pp. 158–159.
28. *Самойлов Н.К.* Трансляция запросов с языка SQL в язык MongoQl // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. 2019. № 3. С. 104–111.
29. *Шепелева Ж.А., Аболмасова Е.С., Емельянова Е.С.* Анализ функционирования технологии NOSQL, реализованной на примере MONGODB // Программная инженерия: современные тенденции развития и применения. 2018. С. 113–116.
30. *Wang S. et al*. A Distributed Storage and Access Approach for Massive Remote Sensing Data in MongoDB // ISPRS Int. J. Geo-Inf. 2019. vol. 8(12). pp. 533.
31. *Marrara S., Pelucchi M., Psaila G.* Blind Queries Applied to JSON Document Stores // Information. 2019. vol. 10(10). pp. 291.
32. *Qian C. et al.* GeoSOT-Based Spatiotemporal Index of Massive Trajectory Data // ISPRS Int. J. Geo-Inf. 2019. vol. 8(6). pp. 284.
33. *Степовик А.Н., Ефанова Н.В.* Анализ реляционных и нереляционных баз данных // Цифровизация экономики: направления, методы, инструменты. 2019. С. 414–416.
34. *Королева Ю.А., Маслова В.О., Козлов В.К.* Разработка концепции миграции данных между реляционными и нереляционными системами БД // Программные продукты и системы. 2019. Т. 32. № 1. С. 063–067.
35. *Višnjevac N. et al.* Prototype of the 3D Cadastral System Based on a NoSQL Database and a JavaScript Visualization Application // ISPRS Int. J. Geo-Inf. 2019. vol. 8(5). pp. 227.
36. *Acquaviva A et al.* Forecasting Heating Consumption in Buildings: A Scalable Full-Stack Distributed Engine // Electronics. 2019. vol. 8(5). pp. 491.
37. *Marchiori A., Li Y., Evans J.* Design and Evaluation of IoT-Enabled Instrumentation for a Soil-Bentonite Slurry Trench Cutoff Wall // Infrastructures 2019. vol. 4(1). pp. 5.
38. *Alfian G. et al.* A Personalized Healthcare Monitoring System for Diabetic Patients by Utilizing BLE-Based Sensors and Real-Time Data Processing // Sensors. 2018. vol. 18(7). pp. 2183.
39. *Syafrudin M., Alfian G., Fitriyani N.L. Rhee J.* Performance Analysis of IoT-Based Sensor, Big Data Processing, and Machine Learning Model for Real-Time Monitoring System in Automotive Manufacturing // Sensors. 2018. vol. 18(9). pp. 2946.
40. *Ха В.М., Шичкина Ю.А., Костичев С.В.* Определение совокупности коллекций для баз данных типа ключ-документ по заданному набору свойств объектов и запросов к базе данных // Компьютерные инструменты в образовании. № (3). С. 15–28.
41. *Shichkina J., Degtyarev A., Kulik B.,Fridman A.* Optimization of relational databases schemas by means of n-tuple algebra // AIP Conference Proceedings. 2017. vol. 1863. no. 1. pp. 11008.

Шичкина Юлия Александровна — д-р техн. наук, профессор, кафедра вычислительной техники, Санкт-Петербургский государственный электротехнический университет "ЛЭТИ" (СПбГЭТУ "ЛЭТИ"). Область научных интересов: параллельные вычисления, базы данных и интеллектуальный анализ данных. Число научных публикаций — 88. strange.y@mail.ru; ул. Профессора Попова, 5/2, 197376, Санкт-Петербург, Россия; р.т.: +7 812 234-25-03.

Ха Ван Муон — аспирант, кафедра вычислительной техники, Санкт-Петербургский государственный электротехнический университет "ЛЭТИ" (СПбГЭТУ "ЛЭТИ"). Область научных интересов: параллельные вычисления, базы данных и интеллектуальный анализ данных. Число научных публикаций — 5. muon.ha@mail.ru; ул. Профессора Попова, 5/2, 197376, Санкт-Петербург, Россия; р.т.: +79650884495.

Поддержка исследований. Исследование выполнено при финансовой поддержке РФФИ и СИТМА в рамках научного проекта №18-57-3400.

DOI 10.15622/sp.2020.19.4.5

Yu. SHICHKINA, V.M. HA

METHOD FOR CREATING COLLECTIONS WITH EMBEDDED DOCUMENTS FOR DOCUMENT-ORIENTED DATABASES TAKING INTO ACCOUNT EXECUTABLE QUERIES

Shichkina Yu., Ha V.M. **Method for Creating Collections with Embedded Documents for Document-oriented Databases Taking into Account Executable Queries.**

Abstract. In the recent decades, NoSQL databases have become more popular day by day. And increasingly, developers and database administrators, for whatever reason, have to solve the problems of database migration from a relational model in the model NoSQL databases like the document-oriented database MongoDB database. This article discusses the approach to this migration data based on set theory. A new formal method of determining the optimal runtime searches aggregate collections with the attached documents NoSQL databases such as the key document. The attributes of the database objects are included in optimizing the number of collections and their structures in search queries. The initial data are object properties (attributes, relationships between attributes) on which information is stored in the database, and query the properties that are most often performed, or the speed of which should be maximal. This article discusses the basic types of connections (1-1, 1-M, M-M), typical of the relational model. The proposed method is the following step of the method of creating a collection without embedded documents. The article also provides a method for determining what methods should be used in the reasonable cases to make work with databases more effectively. At the end, this article shows the results of testing of the proposed method on databases with different initial schemes. Experimental results show that the proposed method helps reduce the execution time of queries can also significantly as well as reduce the amount of memory required to store the data in a new database.

Keywords: NoSQL, Database Query, Collections, Document Key, Broadcast Data, Data Format, Creating the Database Structure, Embedded Documents.

Shichkina Yulia — Ph.D., Dr.Sci., Professor, Department of Computer Engineering, Saint Petersburg Electrotechnical University "LETI". Research interests: parallel computing, databases, data mining. The number of publications — 88. strange.y@mail.ru; 5/2, Professor Popov str., 197376, St. Petersburg, Russia; office phone: +7 812 234-25-03.

Ha Van Muon — Ph.D. Student, Department of Computer Engineering, Saint Petersburg Electrotechnical University 'LETI'. Research interests: parallel computing, database, data mining. The number of publications — 5. muon.ha@mail.ru; 5/2, Professora Popova str., 197376, St. Petersburg, Russia; office phone: +79650884495.

Acknowledgements. This research is supported by RFBR and CITMA according to the research project No. 18-57-3400.

References

1. Rocha L. et al. A Framework for Migrating Relational Datasets to NoSQL. *Procedia Computer Science.* 2015. vol. 51. pp. 2593–2602.
2. Rejting baz dannyh [DB-Engines Ranking]. Rating of Databases [DB-Engines Ranking]. Available at : https://db-engines.com/en/ranking (accessed: 15.07.2020). (In Russ.).
3. Jose B., Abraham S. Performance analysis of NoSQL and relational databases with MongoDB and MySQL. *Materials Today: Proceedings.* 2020. vol. 24. pp. 2036–2043.
4. Čerešňák R., Kvet M. Comparison of query performance in relational a non-relation database. *Transportation Research Procedia.* 2019. vol. 40. pp 170–177.

5. Diogo M., Cabral B., Bernardino J. Consistency Models of NoSQL Databases. Future Internet. 2019. vol.11(2). pp. 43.
6. Chen J., Lee W. An Introduction of NoSQL Databases Based on Their Categories and Application Industries. *Algorithms.* 2019. vol. 12(5). pp. 106.
7. Diogo M., Cabral B., Bernardino J. Consistency Models of NoSQL Databases. *Future Internet*. 2019. vol. 11(2). pp. 43.
8. Li W., Clifton C., Liu S. Database Integration Using Neural Networks: Implementation and Experiences. *Knowl. Inf. Syst.* 2000. vol. 2. no. 1. pp. 73–96.
9. Storey V.C.; Song I.Y. Big data technologies and Management: What conceptual modeling can do. *Data Knowl. Eng.* 2017. vol. 108. pp. 50–67.
10. Corbellini A. et al. S. Persisting big-data: The NoSQL landscape. *Inf. Syst.* 2017. vol. 63. pp. 1–23.
11. Makris A., Tserpes K., Andronikou V., Anagnostopoulos D. A classification of NoSQL data stores based on key design characteristics. Procedia Computer Science. 2016. vol. 97. pp. 94–103.
12. Atzeni P., Bugiotti F., Rossi L. Uniform access to NoSQL systems. *Inf. Syst*. 2014. vol. 43. pp. 117–133.
13. Pardede E., Rahayu, J.W., Taniar D. Mapping Methods and Query for Aggregation and Association Relationship in Object-Relational Database using Collection. Proceedings of the 2004 IEEE International Conference on Information Technology: Coding and Computing (ITCC). 2004. pp. 539–543.
14. Shichkina Y., Kupriyanov M., Shevsky V. The Application of Graph Theory and Adjacency Lists to Create Parallel Queries to Relational Databases. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2018. vol. 10963 LNCS. pp. 61–77.
15. Robinson I., Webber J., Eifrem E. Graph Databases: New Opportunities for Connected Data. O'Reilly Media, Inc. 2015. 238 p.
16. Chickerur S. Comparison of Relational Database with Document-Oriented Database (MongoDB) for Big Data Applications. 8th International Conference on Advanced Software Engineering and Its Applications ASEA. 2015. pp. 41–47.
17. Hanine M., Bendarag A., Boutkhoum O. Data Migration Methodology from Relational to NoSQL Databases. *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering.* 2015. vol. 9. no. 12. pp. 2566–2570.
18. Li X., Ma Z., Chen H. QODM : A Query-Oriented Data Modeling Approach for NoSQL Databases. Advanced Research and Technology in Industry Applications (WARTIA). 2014. pp. 338–345.
19. Zhao G., Huang W., Liang S., Tang Y. Modeling MongoDB with Relational Model. 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies (EIDWT). 2013. pp. 115–121.
20. Alotaibi O., Pardede E. Transformation of Schema from Relational Database (RDB) to NoSQL Databases. *Data.* 2019. vol. 4(4). pp. 148.
21. Celesti A., Fazio M., Villari M. A Study on Join Operations in MongoDB Preserving Collections Data Models for Future Internet Applications. *Future Internet.* 2019. vol. 11(4). pp. 83.
22. Rocha L., Vale F., Cirilo E. A Framework for Migrating Relational Datasets to NoSQL. *Procedia Computer Science.* 2015. vol. 51. pp. 2593–2602.
23. Liang D., Lin Y., Ding G. Mid-model Design Used in Model Transition and Data Migration between Relational Databases and NoSQL Databases. IEEE Int. Conf. Smart City. 2015. pp. 866–869.
24. Hamid S., Rezapour M., Moradi M., Ghadiri N. Performance evaluation of SQL and MongoDB databases for big e-commerce data. IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing. 2015. pp.1–7.

25. Karnitis G., Arnicans G. Migration of Relational Database to Document-Oriented Database : Structure Denormalization and Data Transformation. 7th International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN). 2015. pp. 113–118.
26. Mason R.T. NoSQL Databases and Data Modeling Techniques for a Document-oriented NoSQL Database. *Computer Science.* 2015. pp. 259–268.
27. Gu Y., Shen S., Wang J., Kim J. Application of NoSQL Database MongoDB. IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW). 2015. pp. 158–159.
28. Samoilov N.K. [Translation of queries from SQL to MongoQl]. *Vestnik Voronezhskogo gosudarstvennogo universiteta. Seriya: Sistemnyj analiz i informacionnye tekhnologii – Proceedings of Voronezh State University. Series: Systems Analysis and Information Technology.* 2019. vol. 3. pp. 104–111. (In Russ.).
29. Shepeleva Zh.A., Abolmasova E.S., Emelyanova E.S. [Analysis of the functioning of NOSQL technology, implemented on the example of MONGODB]. *Programmnaya inzheneriya: sovremennye tendencii razvitiya i primeneniya* [Software engineering: modern trends in development and application]. 2018. pp. 113–116. (In Russ.).
30. Wang S. et al. A Distributed Storage and Access Approach for Massive Remote Sensing Data in MongoDB. *ISPRS Int. J. Geo-Inf.* 2019. vol. 8(12). pp. 533.
31. Marrara S., Pelucchi M., Psaila G. Blind Queries Applied to JSON Document Stores. *Information.* 2019. vol. 10(10). pp. 291.
32. Qian C. et al. GeoSOT-Based Spatiotemporal Index of Massive Trajectory Data. *ISPRS Int. J. Geo-Inf.* 2019. vol. 8(6). pp. 284.
33. Stepovik A.N, Efanova N.V. [Analysis of relational and non-relational databases]. *Cifrovizaciya ekonomiki: napravleniya, metody, instrumenty* [Digitalization of the economy: directions, methods, tools]. 2019. pp. 414–416. (In Russ.).
34. Koroleva Yu.A., Maslova V.O., Kozlov V.K. [Development of the concept of data migration between relational and non-relational database systems]. *Programmnye produkty i sistemy – Software products and systems.* 2019. Issue 32. vol. 1. pp. 063–067. (In Russ.).
35. Višnjevac N. et al. Prototype of the 3D Cadastral System Based on a NoSQL Database and a JavaScript Visualization Application. *ISPRS Int. J. Geo-Inf.* 2019. vol. 8(5). pp. 227.
36. Acquaviva A et al. Forecasting Heating Consumption in Buildings: A Scalable Full-Stack Distributed Engine. *Electronics.* 2019. vol. 8(5). pp. 491.
37. Marchiori A., Li Y., Evans J. Design and Evaluation of IoT-Enabled Instrumentation for a Soil-Bentonite Slurry Trench Cutoff Wall. I*nfrastructures*. 2019. vol. 4(1). pp. 5.
38. Alfian G. et al. A Personalized Healthcare Monitoring System for Diabetic Patients by Utilizing BLE-Based Sensors and Real-Time Data Processing. *Sensors.* 2018. vol. 18(7). pp. 2183.
39. Syafrudin M., Alfian G., Fitriyani N.L. Rhee J. Performance Analysis of IoT-Based Sensor, Big Data Processing, and Machine Learning Model for Real-Time Monitoring System in Automotive Manufacturing. *Sensors.* 2018. vol. 18(9). pp. 2946.
40. Ha V.M., Shichkina Yu.A., Kostichev S.V. [Determining the Composition of Collections for Key-Document Databases Based on a Given Set of Object Properties and Database Queries]. *Komp'yuternye instrumenty v obrazovanii – Computer tools in education.* 2019. vol. 3. pp. 15–28. (In Russ.).
41. Shichkina J., Degtyarev A., Kulik B.,Fridman A. Optimization of relational databases schemas by means of n-tuple algebra. AIP Conference Proceedings. 2017. vol. 1863. no. 1. pp. 11008.