УДК 004

К.А. Лаврентьев,
преподаватель кафедры информационных систем и технологий
Хабаровского государственного университета экономики и права
Е.А. Титова
Хабаровский государственный университет экономики и права

ПРОБЛЕМА НЕСТРОГО ПОИСКА В NOSQL ХРАНИЛИЩАХ ДАННЫХ

В работе авторами проведён сравнительный анализ способов хранения данных, необходимых для информационного обеспечения документных структур. Поставлена проблема нестрого поиска данных в NoSQL хранилищах данных и приведены методы решения поставленной проблемы.

Ключевые слова: *хранилище данных, NoSQL, нестрогий поиск, документные структуры, информационное обеспечение.*

This article presents an authors' comparative analysis of methods of storing data necessary for information support of document structures. Issue of informal search by noSQL in data warehouses is defined and methods of its solving are given.

Keywords: *data warehouse, noSQL, informal search, document structures, information support.*

Человечество долгие годы ищет лучшие способы систематизировать информацию за счёт создания описаний объектов в базах данных (БД). От выбранного варианта хранения данных зависит сложность запросов, скорость обработки данных и удобство пользователей при работе с информационной системой. В данный момент существует два основных способа хранения информации:

– реляционные базы данных (РБД);

– NoSQL базы данных.

У каждого из вариантов есть свои плюсы и минусы. Обычно выбор между ними зависит от типа хранимых структур, от обширности выполняемых операций над данными, от размеров и возможности расширяемости БД.

Задачей любой информационной системы является накопление и обработка данных, для этого используются базы данных и их структуры. Основных типов структур две – справочники и документные структуры. Так как информационная система (ИС) является отражением реальной предметной области, то любые действия с данными происходят путём создания и изменения документов.

Рассматривая способы представления документов в базах данных, стоит начать с первого варианта – реляционных баз данных. Это стандартный способ хранения любых данных и организации системы информационного обеспечения ИС.

Документ в РБД представляется обычно n-таблицами, где n – это количество функциональных областей документа. Для примера возьмем форму документа «Экзаменационная ведомость» (рисунок 1).

ЭКЗАМЕНАЦИОННАЯ ВЕДОМОСТЬ

Группа №_____ Дисциплина

№ п/п	Фамилия, имя, отчество	№ зачетной книжки	Оценка	Подпись экзаменатора

«отлично»_____
«хорошо»_____
«удовлетворительно»_____
«неудовлетворительно»_____
«неявки»_____
ИТОГО_____

Рисунок 1 – Форма документа «Экзаменационная ведомость»

Как видно, здесь две функциональные области – шапка документа, подвал документа (представляют собой оформительскую область) и предметная часть с таблицей оценок. Во многих документах предметная часть состоит из больших, чем одна, областей. Таким образом, можно сказать, что представление документа в реляционной базе данных всё-таки представляет некоторую сложность в проектировании и написании запросов, хотя и описано формальными правилами. Для примера приведём схему документа «Экзаменационная ведомость» в реляционной базе данных (рисунок 2).

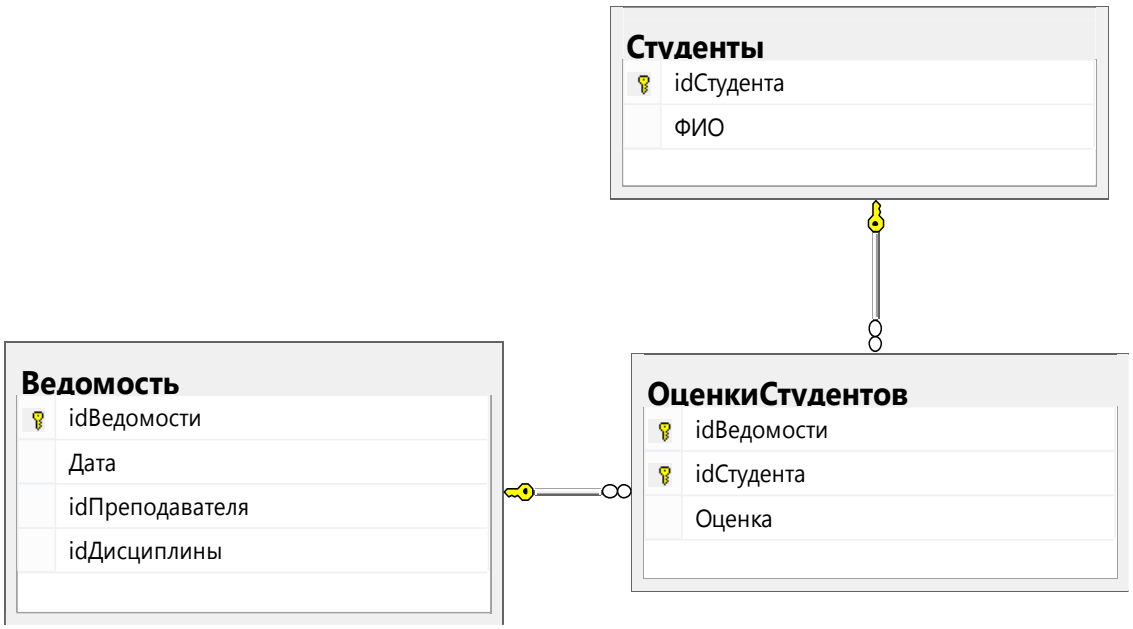

Рисунок 2 – Реляционная модель экзаменационной ведомости

Как видно из этой схемы, при чтении данных необходимо будет построить JOIN-запрос, что отразится на скорости работы. Для формирования полного документа (подставляя вместо значений внешних ключей соответствующие им данные) необходимо будет сделать n+2 JOIN-запросов, где n – это количество студентов в ведомости. Таким образом, можно сказать, что достоинствами этого метода хранения документов является чёткое формальное описание структуры документа в базе данных и отсутствие дублирования информации. Однако недостатком является понижение скорости обработки данных и необходимость построения объёмных JOIN-запросов.

Помимо реляционных баз данных существуют так называемые постреляционные решения (NoSQL), ориентированные на работу с документными структурами – документноориентированные базы данных (ДОБД). Такие базы данных обычно используются для аналитической обработки большого объёма данных и на сайтах с очень высокой нагрузкой, поскольку легко масштабируются горизонтально. Часто ДОБД используются совместно с традиционными решениями на реляционных базах данных. ДОБД не является заменой реляционных баз данных, это, скорее, альтернатива, инструмент, который может делать то же, что могут делать множество прочих.

Основой хранения информации в ДОБД является коллекция – специальный объект документоориентированной базы данных. Представляет собой совокупность атрибутов сущности в их объектном представлении. Коллекция похожа на объект в объектно-ориентированном программировании, она содержит в себе все атрибуты сущности в виде примитивных типов данных и более сложных (массивы). Основным форматом хранения в ДОБД является формат json. Таким образом, при использовании ДОБД исключается необходимость разбиения сущности на нормализованные таблицы реляционной базы, что избавляет от написания сложных JOIN-запросов. В итоге можно сказать, что достоинствами ДОБД является:

– высокая скорость работы за счёт отсутствия JOIN запросов и отсутствия необходимости нормализации базы данных;

– семантическое соответствие результирующей базы данных сущностям предметной области.

Однако любой подход не может быть лишён недостатков, их можно выделить и у ДОБД:

– сложность в обеспечении целостности базы данных за счёт отсутствия механизма внешних ключей;

– высокая вероятность дублирования данных в базе данных.

Рассмотрев оба способа представления документов в базе данных и узнав их достоинства и недостатки, необходимо провести сравнительный анализ этих двух способов и выбрать наиболее оптимальный для хранения документов и их последующего поиска по различным, заранее неизвестным запросам. Для сравнительного анализа были взяты следующие критерии:

– скорость работы (отсутствие необходимости построения сложных запросов соединения и объединения);

– удобство проектирования структуры базы данных;

– встроенные механизмы поддержки целостности;

– удобство работы пользователей.

Для сравнения авторы взяли два способа хранения документов в информационных системах – реляционные базы данных (РБД) и NoSQL базы данных. Для анализа взята примитивная шкала оценок: если рассматриваемый способ поддерживает рассматриваемую возможность, то ставится 1, иначе ставится 0. Результат сравнительного анализа приведён в таблице.

Таблица – Результаты сравнительного анализа

№ п/п	Критерий	РБД	NoSQL
	1	2	3
1	Скорость работы (отсутствие необходимости построения сложных запросов соединения и объединения)	0	1
2	Удобство проектирования структуры базы данных	1	1
3	Встроенные механизмы поддержки целостности	1	0
4	Удобство работы пользователей	0	1
Итого		**2**	**3**

Таким образом, можно сказать, что для хранения информации, необходимой для документов в базах данных, больше подходит NoSQL подход. Конечно, данный сравнительный анализ, как и любой сравнительный анализ, субъективен. В некоторых ситуациях применение РБД более уместно, всё зависит от выбранных критериев оценки двух подходов.

При работе с документными структурами важно учитывать не только создание документов, но и организацию поиска в них. Допустим, что в ДОБД хранится множество документов в которых содержатся, к примеру, списки всех возможных журналов для публикации статей. Пользователю нужно найти все документы, связанные с его предметной областью (ПрО). Что он будет делать? Он создаст поисковый запрос, в котором просто укажет название ПрО, в ответ на это система должна выдать то, что он хочет.

Как же организовать поиск по ДОБД? Самым очевидным вариантом будет поиск по полному совпадению. Но проблема в том, что все документы имеют чёткую структуру, а пользователи чаще всего формируют свои запросы в свободной форме. Процент нерелевантной информации будет значимым из-за неправильно сформулированной поисковой фразы. Также сложности возникают из-за структуры ДОБД: поиск придётся осуществлять по всем полям базы данных.

Решением проблемы будет использование нестрого поиска. Нечёткий поиск – это одна из важнейших функций поисковой системы. Он используется в таких задачах, как проверка орфографии, исправление опечаток, поиск похожих значений или поиск вхождения подстроки в тексте. Существует несколько алгоритмов для его реализации:

– расстояние Левенштейна;

– хеширование по сигнатуре;

– BK-деревья.

Расстояние Левенштейна

Наиболее часто используемым является расстояние Левенштейна, или расстояние редактирования. Оно позволяет субъективно оценить, насколько строки не похожи друг на друга. Это «расстояние» – минимальное количество правок одной строки (под правками подразумеваются три возможные операции: стирание символа, замена символа и вставка символа), чтобы превратить её во вторую. Пример:

Levenshtein ('ABC','ABC') = 0

Levenshtein ('ABC','ABCDEF') = 3

Levenshtein ('ABC','BCDE') = 3

Levenshtein ('BCDE','ABCDEF') = 2.

Данный алгоритм обычно применяется для исправления орфографических ошибок, для сравнения текстов. Из недостатков можно выделить неуниверсальность алгоритма. Существует несколько ситуаций, где данный алгоритм показывает плохой результат. Например, у совершенно разных коротких слов будет минимальное расстояние, когда у максимально похожих длинных слов расстояние может быть больше. Если просто поменять части слова местами, то расстояние получится довольно-таки большим.

Хеширование по сигнатуре

Данный способ основан на достаточно очевидном представлении «структуры» слова в виде битовых разрядов, используемой в качестве хеша (сигнатуры) в хеш-таблице (рисунок 3). Сигнатура в данном случае – это уникальная часть слова.

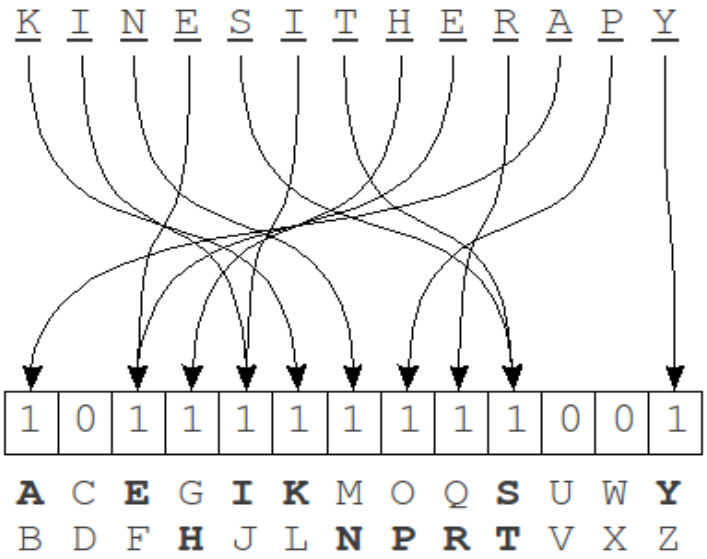

Рисунок 3 – Пример хеширования по сигнатуре

Каждому биту хеша сопоставляется группа символов из алфавита. Бит 1 на позиции i в хеше означает, что в исходном слове присутствует символ из i-й группы алфавита. Порядок букв в слове абсолютно никакого значения не имеет. Реализовать этот алгоритм можно двумя способами: хранить хеши всех слов, или постоянно его вычислять. Здесь нужно делать выбор между скоростью и памятью.

BK-деревья

Деревья Баркхарда – Келлера (Burkhard – Keller) являются метрическими деревьями, алгоритмы построения таких деревьев основаны на свойстве метрики отвечать неравенству треугольника:

$$\rho(x,y) \leqslant \rho(x,z) + \rho(z,y), \quad x,y,z \in X.$$

Это свойство позволяет метрикам образовывать метрические пространства произвольной размерности. На основании этого свойства можно построить структуру данных, осуществляющую поиск в таком метрическом пространстве, которой и являются деревья Баркхарда – Келлера (рисунок 4).

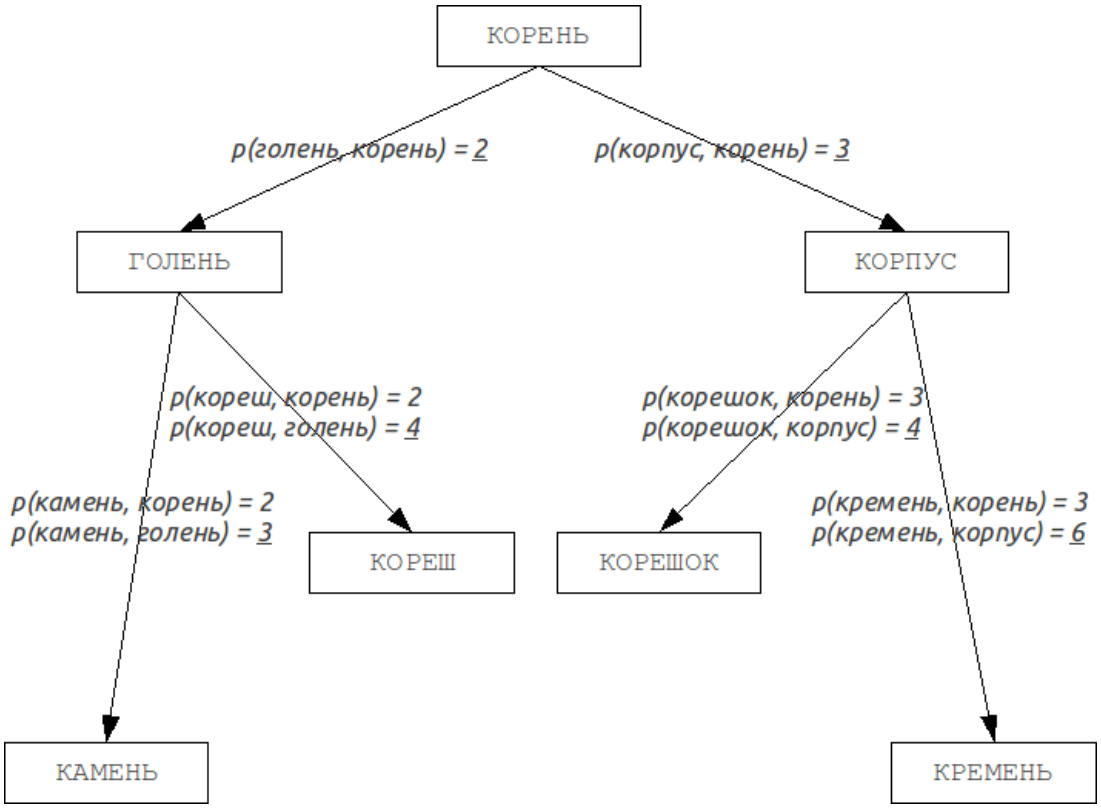

Рисунок 4 – БК-дерево

Все вышеперечисленные способы имеют свои достоинства и недостатки, но каждый из них помогает в решении проблемы нестрогого поиска. Для того чтобы выбрать лучший из них, необходимо сравнить их с точки зрения производительности.

Список использованных источников

1. Дейт К. Дж. Введение в системы баз данных / К. Дж. Дейт. – 8-е изд. М. : Вильямс, 2006.

2. Аносова Н. П. Распределённые базы и хранилища данных / Н. П. Аносова, О. О. Бородин, Е. С. Гаврилов, А. М. Марсанов. М. : Интуит, 2009.

3. Лаврентьев К. А. Проблемы проектирования архитектуры распределённых баз

данных / К. А. Лаврентьев Е. А. Титова // Вестник ХГАЭП. 2015. № 1 (75). С. 33–38.