|  |
| --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| Федеральное государственное автономное образовательное учреждение высшего профессионального образования |
| **Национальный исследовательский ядерный университет «МИФИ»**IFES-logo |

*ФАКУЛЬТЕТ*

*КИБЕРНЕТИКИ И ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ*

**Кафедра №75«Финансовый мониторинг»**

Реферат на тему

«Не реляционные СУБД»

по курсу:

«Специальные Технологии Баз Данных и Информационных Систем»

Выполнила: Чистякова М.В.

студентка группы Р01-75М

Проверила: Евстифеева Н.А.

Москва, 2015г

Оглавление

[1. Не реляционные СУБД. 3](#_Toc435619518)

[*1.1.* *История* 3](#_Toc435619519)

[*1.2.* *Основные черты NoSQL баз данных* 3](#_Toc435619520)

[*1.3.* *Основные типы хранилищ данных* 4](#_Toc435619521)

[2. MongoDB 5](#_Toc435619522)

[*2.1.* *Модель устройства базы данных* 5](#_Toc435619523)

[*2.2.* *Формат данных* 5](#_Toc435619524)

[*2.3.* *Работа с базой данных* 6](#_Toc435619525)

# Не реляционные СУБД.

Не реляционные или NoSQL (Not only SQL) СУБД в отличие от привычных реляционных СУБД используют не только средства языка SQL для доступа к базам данных.

* 1. История.

Использование не реляционных хранилищ данных началось ещё во времена первых компьютеров. После изобретения реляционных СУБД их популярность упала, но они нашли применение в специализированных хранилищах. С появлением в начале 2000-х высоко масштабируемых интернет-приложений, возникла необходимость в использовании не реляционных баз дынных. Первыми такие базы данных стали использовать Google, Amazon, IBM и др.

Термин NoSQL появился случайно. В июне 2009 в Сан-Франциско Йоханом Оскарссоном была организована встреча, на которой планировалось обсудить новые веяния на IT рынке хранения и обработки данных, в том числе и не реляционные базы данных. Для яркой вывески требовалось найти емкий и лаконичный термин, который отлично укладывался бы в Твиттеровский хэштег. Один из таких терминов предложил Эрик Эванс из RackSpace — «NoSQL». Термин планировался лишь на одну встречу и не имел под собой глубокой смысловой нагрузки, но так получилось, что он распространился по мировой сети наподобие вирусной рекламы и стал де-факто названием целого направления в IT-индустрии.

* 1. Основные черты NoSQL баз данных.

Традиционные СУБД ориентируются на требования ACID к транзакционной системе: атомарность (англ. atomicity), согласованность (англ. consistency), изолированность (англ. isolation), надёжность (англ. durability), тогда как в NoSQL СУБД ориентируется на набор требований BASE:

* базовая доступность (англ. basic availability) — каждый запрос гарантированно завершается (успешно или безуспешно).
* гибкое состояние (англ. soft state) — состояние системы может изменяться со временем, даже без ввода новых данных, для достижения согласования данных.
* согласованность в конечном счёте (англ. eventual consistency) — данные могут быть некоторое время рассогласованы, но приходят к согласованию через некоторое время.

Системы, ориентированные на BASE, не могут использоваться там, где необходимо обеспечить применение транзакций. Например, для функционирования биржевых и банковских система. Но в то же время они идеально подходят для обеспечения работы систем с большой веб-аудиторией.

Другими характерными чертами NoSQL являются:

* Применение различных типов хранилищ;
* Возможность разработки базы данных без задания схемы;
* Использование многопроцессорности;
* Линейная масштабируемость (добавление процессоров увеличивает производительность);
* Инновационность: «не только SQL» открывает много возможностей для хранения и обработки данных;
* Сокращение времени разработки;
* Скорость: даже при небольшом количестве данных конечные пользователи могут оценить снижение времени отклика системы с сотен миллисекунд до миллисекунд.
	1. Основные типы хранилищ данных.

Описание схемы данных в случае использования NoSQL-решений может осуществляться через использование различных структур данных.

* **Хранилище «ключ-значение»**

Хранилища «ключ-значение» является простейшим хранилищем данных, использующим ключ для доступа к значению. Такие хранилища используются для хранения изображений, создания специализированных файловых систем, в качестве кэшей для объектов, а также в системах, спроектированных с прицелом на масштабируемость. Примеры таких хранилищ — Berkeley DB, MemcacheDB, Redis, Riak, Amazon DynamoDB.

* **Хранилище семейств колонок (или Bigtable-подобные базы данных)**

В этом хранилище данные хранятся в виде разреженной матрицы, строки и столбцы которой используются как ключи. Типичным применением этого вида СУБД является веб-индексирование, а также задачи, связанные с большими данными, с пониженными требованиями к согласованности данных. Примерами СУБД данного типа являются: Apache HBase, Apache Cassandra, Apache Accumulo, Hypertable, SimpleDB (Amazon.com).

* **Документо-ориентированная СУБД**

Документо-ориентированные СУБД служат для хранения иерархических структур данных. Находят своё применение в системах управления содержимым, издательском деле, документальном поиске и т. п. Примеры СУБД данного типа — CouchDB, Couchbase, MarkLogic, MongoDB, eXist, Berkeley DB XML.

* **Базы данных на основе графов**

Графовые базы данных применяются для задач, в которых данные имеют большое количество связей, например, социальные сети, выявление мошенничества. Примеры: Neo4j, OrientDB, AllegroGraph, Blazegraph, InfiniteGraph, FlockDB, Titan.

# MongoDB.

Для практической работы была выбрана СУБД MongoDB. Она реализует новый подход к построению баз данных, где нет таблиц, схем, запросов SQL, внешних ключей и многих других вещей, которые присущи объектно-реляционным базам данных.

MongoDB предлагает документо-ориентированную модель данных, благодаря чему она работает быстрее, обладает лучшей масштабируемостью, ее легче использовать. Данную СУБД можно использоваться, если есть необходимость в хранение сложных по структуре данных.

Реализована на языке С++, что обеспечивает кроссплатформенность.

* 1. Модель устройства базы данных.



Рисунок 1. Модель устройства базы данных MondoDB

Вместо таблиц и строк в базе данных хранятся документы (в качества строки) и коллекции (в качестве таблицы).

Каждая коллекция имеет свое уникальное имя - произвольный идентификатор, состоящий из максимум 128 различных алфавитно-цифровых символов и знака подчеркивания.

Главное отличие от реляционных СУБД заключается в том, что в документах может храниться разнородная информация, но они все равно будут объединены в коллекции.

Документ представляет собой набор пар ключ-значение. Проводя аналогию с реляционной БД, ключ это название столбца, а значение – содержимое атрибута таблицы. Так же у каждого документа существует индивидуальный номер \_id, который может быть присвоен автоматически, или задан при добавлении документа.

* 1. Формат данных.

Для хранения в MongoDB применяется формат, который называется BSON (БиСон) или сокращение от binary JSON. BSON позволяет работать с данными быстрее: быстрее выполняется поиск и обработка.

Все возможные значения:

* **String**: строковый тип данных, как в приведенном выше примере (для строк используется кодировка UTF-8)
* **Array (массив)**: тип данных для хранения массивов элементов
* **Binary data (двоичные данные)**: тип для хранения данных в бинарном формате
* **Boolean**: булевый тип данных, хранящий логические значения TRUE илиFALSE, например, {"married": FALSE}
* **Date**: хранит дату в формате времени Unix
* **Double**: числовой тип данных для хранения чисел с плавающей точкой
* **Integer**: используется для хранения целочисленных значений, например,{"age": 29}
* **JavaScript**: тип данных для хранения кода javascript
* **Min key/Max key**: используются для сравнения значений с наименьшим/наибольшим элементов BSON
* **Null**: тип данных для хранения значения Null
* **Object**: строковый тип данных, как в приведенном выше примере
* **ObjectID**: тип данных для хранения id документа
* **Regular expression**: применяется для хранения регулярных выражений
* **Symbol**: тип данных, идентичный строковому. Используется преимущественно для тех языков, в которых есть специальные символы.
* **Timestamp**: применяется для хранения времени
	1. Работа с базой данных.

Работа с базой данных начинается с запуска сервера.



Рисунок 2. Запуск сервера

После чего запускаем командную строку, в которой и происходит основная работа.



Рисунок 3. Запросы к базе данных

Работу начинаем с установки одной из баз данных в качестве текущей с помощью команды use. Если такой базы не существует, то она будет создана автоматически. Для просмотра всех возможных баз данных используется команда show dbs. Выбранной базы info в перечне еще нет, т.к. в нее пока не добавлена никакая информация.

Переходим к добавлению документов. После db добавляем имя коллекции, и она будет создана автоматически. Впоследствии ее можно будет переименовать с помощью команды renameCollection("новое\_название"). При необходимости коллекцию можно явно создать коллекцию с помощью команды createCollection("accounts").

Для вставки документа в коллекцию используется команда insert(). В скобках передается документ в фигурных скобках. В базу данных вносится документы одинаковые по структуре, но при необходимости в коллекцию можно добавить документы отличный по структуре.

Перейдем к запросам к базе данных. Для поиска используется команда find, с ее помощью осуществляются все запросы аналогичные запросам Select на языке SQL. Команда, используемая без параметров в скобках, аналогична запросу SELECT \*, выводит всю информацию в коллекции. Так же в скобках можно указывать необходимое значение одного или нескольких атрибутов, что равносильно условию WHERE. А так же можно выбирать необходимые для вывода атрибуты, что равносильно условию SELECT.



Рисунок 4. Запросы к базе данных продолжение

Для условных конструкций используются специальные операторы сравнения:

* $gt (больше чем)
* $lt (меньше чем)
* $gte (больше или равно)
* $lte (меньше или равно).

Так же существует оператор $all, который находит документ только со всеми возможными значениями в массиве. Оператор $in находит те документы, значение ключей которых представлено в массиве, а оператор $nin наоборот.

Так же еще один полезный оператор это $size. Он используется для нахождения документов, в которых массивы имеют число элементов, равное значению.

# Используема литература

* 1. Введение в MongoDB. URL: <http://metanit.com/nosql/mongodb/1.1.php>. (Дата последнего обращения: 18.11.2015).
	2. Статья по NoSQL. URL: <https://ru.wikipedia.org/wiki/NoSQL>. (Дата последнего обращения: 11.11.2015).
	3. NoSQL базы данных: понимаем суть. URL: <http://habrahabr.ru/post/152477/> (Дата последнего обращения: 11.11.2015).