Документ-ориентированные хранилища

Объектные базы данных. Лекция 3

Документ-ориентированные хранилища

- Модель данных подобных хранилищ позволяет объединять множество пар ключ-значение в абстракцию, называемую «документ». Документы могут иметь вложенную структуру и объединяться в коллекции.
- Примеры СУБД: MongoDB, CouchDB, Couchbase, MarkLogic, eXist, Berkeley DB

MongoDB

- ***MongoDB*** – документо-ориентированная система управления базами данных (СУБД) с открытым исходным кодом, не требующая описания схемы таблиц.

- Разработчик: американская компания MongoDB Inc. (ранее называлась «10gen»)
- MongoDB написана на языке C++.
- Первый выпуск: 2009 г.
- Последняя версия: 3.2.10 (30 сентября 2016 года)
- Лицензия: GNU AGPL
- MongoDB используется в Microsoft Azure, Foursquare, bit.ly, ЦЕРН (для хранения данных, поступающих с адронного коллайдера) и др.

MongoDB

- MongoDB сочетает в себе мощные средства запросов, характерные для реляционных баз данных, и распределенную архитектуру, свойственную таким хранилищам, как Riak.
- MongoDB – это хранилище JSON-документов (хранит данные в BSON-формате).
- Поддержка произвольных запросов.
- MongoDB – отличный выбор для растущего класса веб-проектов, в которых необходимо работать с большими массивами данных, но бюджет слишком мал для приобретения дорогостоящего оборудования.

Представление данных в MongoDB

Единица хранения данных в Mongo – ***документ*** – документ в формате JSON. ***Коллекция*** – набор документов в формате JSON.

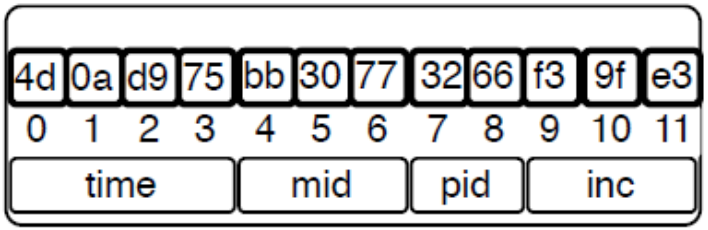
```
$ mongo book
> db.towns.insert
  ({
    name: "New York",
    population: 22200000,
    last_census: ISODate("2009-07-31"),
    famous_for: ["statue of liberty", "food" ],
    mayor : {
      name : "Michael Bloomberg",
      party : "I"
    }
  })

> show collections
  system.indexes
  towns
```

коллекция

Документ в формате JSON

Идентификатор документа

MongoDB добавляет к каждому документу поле _id типа ObjectID (12 байт), играющего роль первичного ключа.



_id = временная метка,
идентификатор клиентской машины,
идентификатор клиентского процесса,
3-байтовый инкрементируемый счетчик.

Поле _id можно сгенерировать самостоятельно с собственным значением.

```
> db.towns.find()
  ({
    "_id" : ObjectId("4d0ad975bb30773266f39fe3"),
    name: "New York",
    population: 22200000,
    last_census: ISODate("2009-07-31"),
    famous_for: ["statue of liberty", "food" ],
    mayor : {
      name : "Michael Bloomberg",
      party : "I"
    }
  })
```

Запросы к MongoDB

Запросы к MongoDB составляются с использованием языка JavaScript.

```
db.collection_name.comand
```

- `db` – JavaScript-объект, который содержит информацию о текущей базе данных.
- `collection_name` – JavaScript-объект, представляющий коллекцию с именем collection_name.
- `comand` – JavaScript-функции.

Команды:

- Стандартные (insert(), find(), count())
- Собственные

Операции CRUD

- ***CRUD*** — сокращенное именование 4-х базовых функций, используемых при работе с хранилищами данных:
  - создание (***Create***);
  - чтение (***Read***);
  - редактирование (***Update***);
  - удаление (***Delete***).

Операция	**Оператор в языке SQL**	**Операция в протоколе HTTP**
Создание (create)	INSERT	POST
Чтение (read)	SELECT	GET
Редактирование (update)	UPDATE	PUT или PATCH
Удаление (delete)	DELETE	DELETE

Извлечение данных

- Извлечение конкретного документа:

```
db.towns.find({ "_id" : ObjectId("4d0ada1fbb30773266f39fe4") })
```

- Извлечение конкретных полей конкретного документа:

```
db.towns.find({ "_id" : ObjectId("4d0ada1fbb30773266f39fe4") },
              { name : 1 })
```

- Запросы к вложенным массивам (поиск конкретного значения, сравнение с подстрокой и т.д.):

```
db.towns.find({ famous_for : 'food' })
```

- Формулировка запросов по значениям полей, диапазонам или с несколькими критериями:

  `field : { $op : value }`, где $op – операция (=, >, <, др.)

  Например, «найти все города, названия которых начинаются с буквы *P*, а численность населения меньше 10 000»:

```
db.towns.find({ name : /^P/, population : { $lt : 10000 } })
```

Некоторые булевские операции

Команда операции ($op)	Описание
$or	OR
$lt	Меньше
$lte	Меньше или равно
$regex	Соответствие строки регулярному выражению, совместимому с синтаксисом PCRE
$exists	Проверяет существование поля
$all	Соответствие всем элементам массива
$in	Соответствие хотя бы одному элементу массива
$nin	Несоответствие ни одному элементу массива
$elemMatch	Соответствие всех полей вложенного документа

Агрегированные запросы

- **count(<запрос>)** – принимает запрос и возвращает количество удовлетворяющих критерию документов.

```
db.phones.count({
  'components.number': { $gt : 5599999 }
})
```

Результат: 5000

- **distinct(<запрос>)** – возвращает удовлетворяющие критерию значения (не полные документы), если хотя бы одно существует.

```
db.phones.distinct('components.number',
{'components.number': { $lt : 5550005 } })
```

Результат: [5550000, 5550001, 5550002, 5550003, 5550004]

- **group(initial, reduce, cond, key)** - агрегирует результаты примерно так же, как запросы с фразой GROUP BY в SQL.
  - Initial – инициализация объекта-результата
  - key – поле, по которому производится группировка;
  - cond – условие, определяющее, какие значения нас интересуют;
  - reduce – функция, которая решает, как выводить результаты

Обновление/добавление/удаление данных в документе

Функция `update(criteria,operation)`

- `criteria` – критерий отбора – такой же, как для функции find()
- `operation` – либо объект, поля которого заменяют поля отобранных документов, либо модификатор.

Пример:

```
db.towns.update(
{ _id : ObjectId("4d0ada87bb30773266f39fe5") },
{ $set : { "state" : "OR" } }
);
```

Модификатор (operation)	Описание
$set	Записывает указанное значение в указанное поле
$unset	Удаляет поле
$inc	Прибавляет указанное число к указанному полю
$push	Помещает новый элемент в массив
…	…

Ссылки

- MongoDB не выполняет операцию соединения. Из-за распределенной природы системы соединение оказалось бы крайне неэффективной операцией.

- Допускается создание ссылок между документами:

```
{ $ref : “collection_name”, $id : “reference_id” }
```

Удаление документов

- remove() – функция удаления документа/документов из коллекции. Документ удаляется целиком.
- Параметры функции remove() такие же как и у find().

- Пример:

```
db.towns.remove({
    "_id" : ObjectId("4d0ada1fbb30773266f39fe4")
})
```

Индексирование в MongoDB

- Для повышения скорости выполнения запросов в MongoDB можно воспользоваться индексированием.
- MongoDB поддерживает несколько структур индексов:
  - классическое B-дерево
  - двумерные и сферические пространственные индексы

- Создание индекса: **`ensureIndex(fields,options)`**
  `fields` – это объект, содержащий поля, по которым строится индекс
  `options` описывает тип индекса.

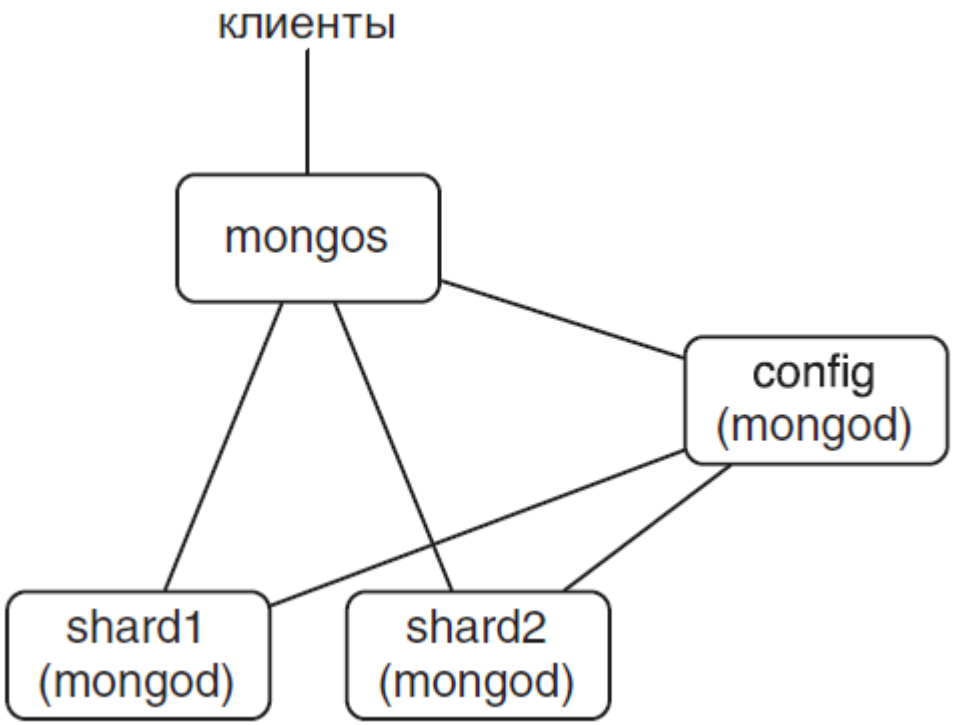
- *Замечание*: При создании любой коллекции автоматически строится индекс по полю _id. Эти индексы хранятся в коллекции system.indexes.

Высокопроизводительная обработка в MongoDB

- Mongo отличается умением масштабироваться на несколько серверов путем
  - репликации (копирования данных на другие серверы)
  - или сегментирования коллекций (разбиения коллекции на части),
- В Mongo возможно параллельное выполнение запросов.

- Репликация данных производится в соответствии с архитектурой Master-Slave. Т.о. достигается согласованность данных.

Сегментирование (sharding)



Конфигурационный сервер - управляет информацией о сегментировании

- При сегментации указывается коллекция и поле сегментирования. Дальнейшее распределение берет на себя механизм автосегментирования.