MongoDB

Christof Strauch ”NoSQL Databases”

Podkopaev Anton, podkoav239@gmail.com

SE, SPbSU

28 October 2013

MongoDB

- Schema-free document database
- Written in C++
- Open-source, 10gen Inc.
- Adjective *humongous*
- Goal:
  Gap between key-/value-stores and RDBMSs
- Users:
  SourceForge.net, foursquare, the New York Times, bit.ly, DIASPORA, Yandex, etc.

DB structure

- DBs reside on MongoDB server
- Server hosts many of such DBs
- DBs contain collections
- Collections consist of documents

Collection

- Def: named groupings of documents
- A collection may be heterogeneous
- Manual suggests:
  One database collection for each of your top level objects

Creation

- Created automatically at the first document is inserted into a DB
- Parameters
  - auto-indexing
  - preallocated disk space
  - size limits
  - ...
- Explicit creation

```
db.createCollection(
        "mycoll", {size: 100000000, autoIndexId: false}
)
```

Namespaces

- Hierarchical namespaces using a dot-notation

```
wiki.articles
wiki.categories
wiki.authors
```

- The collection namespace is flat from the DB's perspective

Document

- The abstraction and unit of data storable in MongoDB
- Comparable to
  - XML doc
  - JSON doc
  - Python dic
  - Ruby hash
- In fact, the format is called BSON
- 4Mb — limit of document size

Example

```
{
        title: "MongoDB",
      , last_editor: "172.5.123.91"
      , last_modified: new Date("9/23/2010")
      , body: "MongoDB is a..."
      , categories:
             ["Database"
             , "NoSQL"
             , "Document Database"
             ]
      , reviewed: false
}
```

Datatypes

- boolean, integer, double
- string (UTF-8), regular expressions, code (JS)
- object (BSON)
- object id (12 bytes)
  - timestamp (4 bytes)
  - Machine id (3 bytes)
  - MongoDB process id (2 bytes)
  - counter (3 bytes)
- null
- array
- date

References

- Doesn’t provide a foreign key mechanism
- Manual assignment of ref field the value of the *_id*
- DBRef
  - Can be dereferenced automatically
  - `{ $ref: <collectionName>, $id: <documentId> }`

References vs Embeddings

- Criteria for Object References
  - First-class domain objects
  - Many-to-many
  - Queried in large numbers
  - Large objects
- Criteria for Object Embeddings
  - ”Line-item detail” characteristic
  - Aggregation relationship
  - Not referenced by another object
  - Request performance

Selection

- Equivalent to the WHERE clause in SQL
- In case of empty object, all documents of a collection are returned

Example

```
db.collectionName.find( { title: "MongoDB" } )
```

General form

```
<fieldname>: {$<operator>: <value>}

<fieldname>: {$<operator>: <value>, $<operator>: <value>}
// AND-junction
```

Selection.Operators

```
\$ne                          // !=
\$gt, \$gte, \$lt, \$lte // >, >=, <, <=
{ \$mod: [2, 1] }
{ \$in : ["NoSQL", "Document Databases"] }
{ \$nin: ["NoSQL", "Document Databases"] }
{ \$all: ["NoSQL", "Document Databases"] }
{ \$size: 2 } // for array
{ \$exists: false }

// OR
{$or: [{ $reviewed: {$exists: true}, {$categories: {$size: 2}}
$nor
$not

{ field.index.field: value }

db.collection.find({ $where: "this.a==1" })
```

Projection

```
db.collection.find(
        {<selection criteria>}, {<f1>: 1, <f2>: 1, ...}
)
db.collection.find(
        {<selection criteria>}, {<f1>: 0, <f2>: 0, ...}
)
```

Result processing

- sort
- limit
- skip
- count

```
db.collection.find({<selection>},
        {inclusion/exclusion},
        <limit>, <skip>
)

db.collection(...).count()
```

Cursors

```
var cursor = db.collection.find(...)
cursor.forEach( function(result) {...} )

db.collection.findOne( {_id: 239239} )
```

Query optimizer

- Supports ad-hoc queries

- Not based on statistics
- Not model the costs of multiple possible query plans

- Execute different query plans in parallel
- Stops all of them as soon as the first has finished

- The system is non-relational
- There are no joins
- The space of possible query plans much smaller