CRUD

MongoDB CRUD Operations

Create

Read

Update

Delete

Explore Databases and Collections

- **Show active Database**

```
db
```

- **List all Databases**

```
show dbs
```

- **Change active database**

```
use <database name>
```

- **List all Collections inside active Database**

```
show collections
```

Create, Delete Database and Collection

(1) Change active Database

```
use <database name>
```

(2) Create new Collection

```
db.createCollection("<collection name>")
```

(3) List all Collections inside active Database

```
show collections
```

(4) Delete Collection

```
db.getCollection("<collection name>").drop()
db.<collection name>.drop()
```

(5) Delete Database

```
db.dropDatabase()
```

Note

There is no command to create new Database.

Database will be created together with first Collection.

Insert new Documents

- **Insert One or Many Documents**

```
db.<collection name>.insert(<Object> or <Array of Objects>)
```

returns WriteResult or BulkWriteResult Object

- **Insert One Document**

```
db.<collection name>.insertOne(<Object>)
```

returns "insertedId" of the inserted document

- **Insert Many Documents**

```
db.<collection name>.insertMany(<Array of Objects>)
```

returns "insertedIds" of the inserted documents

insert()

- **Insert Many Documents**

```
db.<collection
name>.insert(<Array of Objects>)
```

```
BulkWriteResult({
  "writeErrors" : [ ],
  "writeConcernErrors" : [ ],
  "nInserted" : 3,
  "nUpserted" : 0,
  "nMatched" : 0,
  "nModified" : 0,
  "nRemoved" : 0,
  "upserted" : [ ]
})
```

- **Insert One Document**

```
db.<collection
name>.insert(<Object>)
```

```
WriteResult({ "nInserted" : 1 })
```

insertOne()

- **Insert One Document**

```
db.<collection name>.insertOne(Object)
```

```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5ad06d63be9df1be43808588")
}
```

insertMany()

- **Insert Many Documents**

```
db.<collection name>.insertMany(<Array of Objects>)
```

```
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("5ad06d71be9df1be43808589"),
    ObjectId("5ad06d71be9df1be4380858a"),
    ObjectId("5ad06d71be9df1be4380858b")
  ]
}
```

Insert Document Example

▼ (15) ObjectId("5acd0b867e...	{ 9 fields }	Object
_id	ObjectId("5acd0b867e27e66e02f...	ObjectId
string	Hello World	String
▶ object	{ 1 field }	Object
▶ array	[3 elements]	Array
numberInt	10	Int32
numberLong	3234234234234	Int64
double	5.75	Double
boolean	true	Boolean
date	2018-04-10 19:07:50.109Z	Date

Read Documents

- **Find Documents**

```
db.<collection name>.find(<query>, <fields>)
```

Returns Cursor

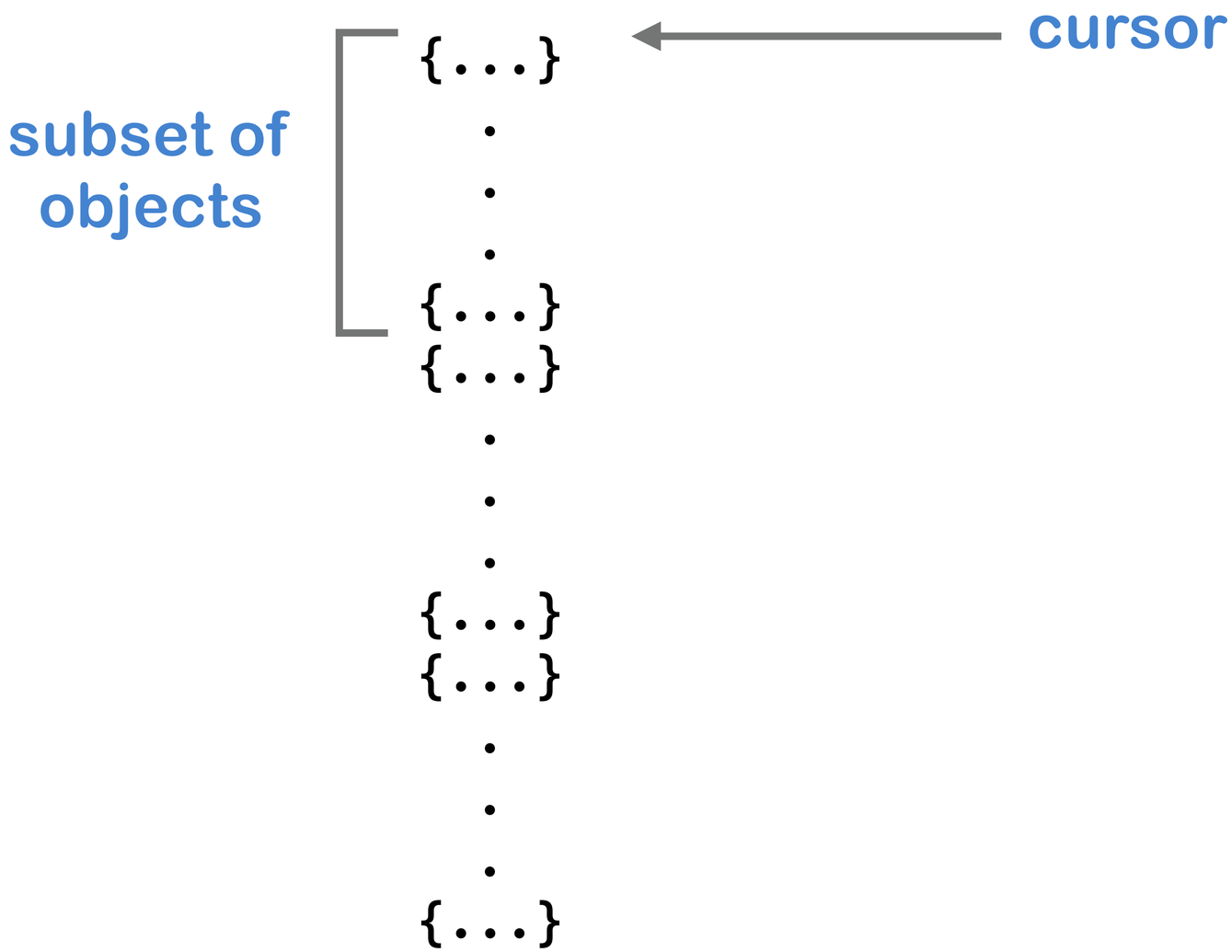
Cursor must be iterated to get documents in Extended JSON format

- **Find One Document**

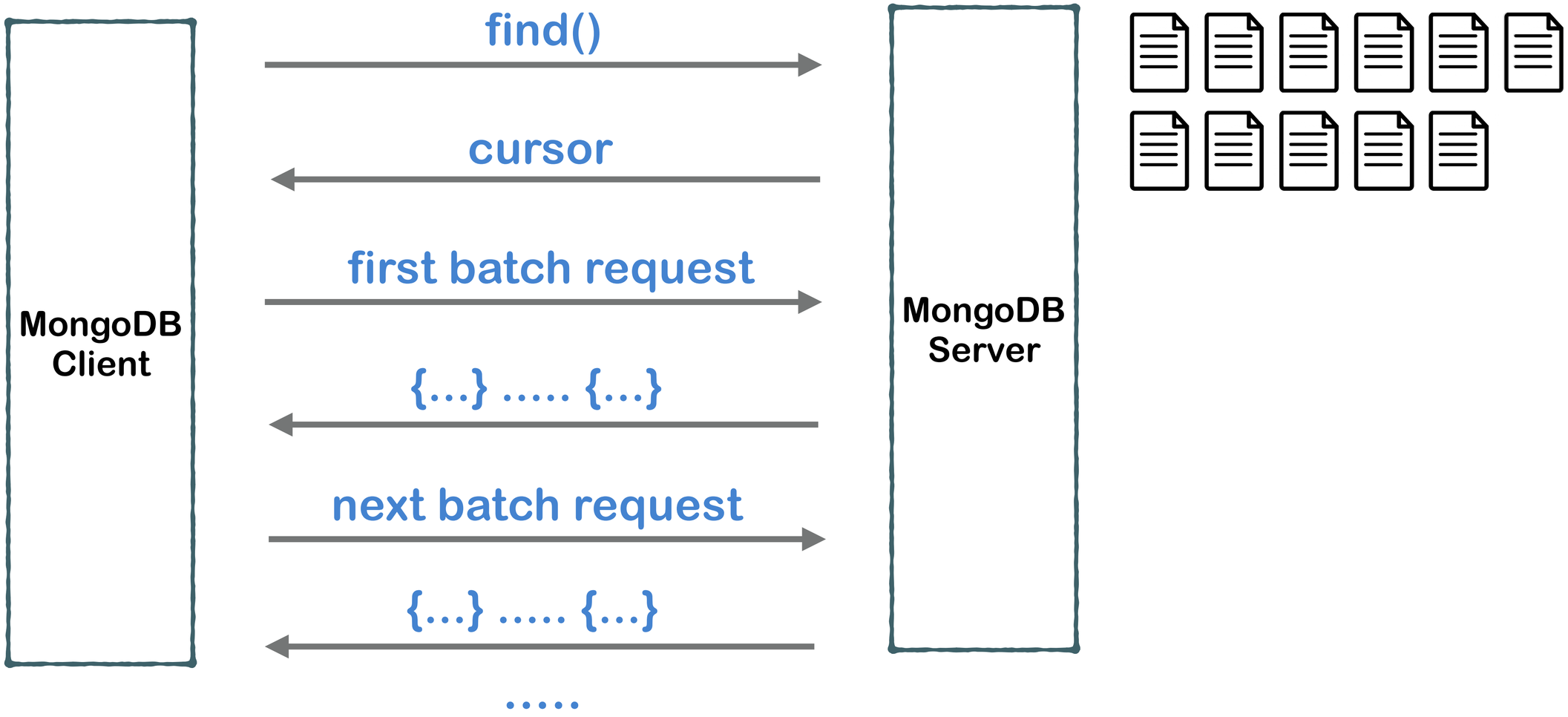
```
db.<collection name>.findOne(<query>, <fields>)
```

Returns Extended JSON Object

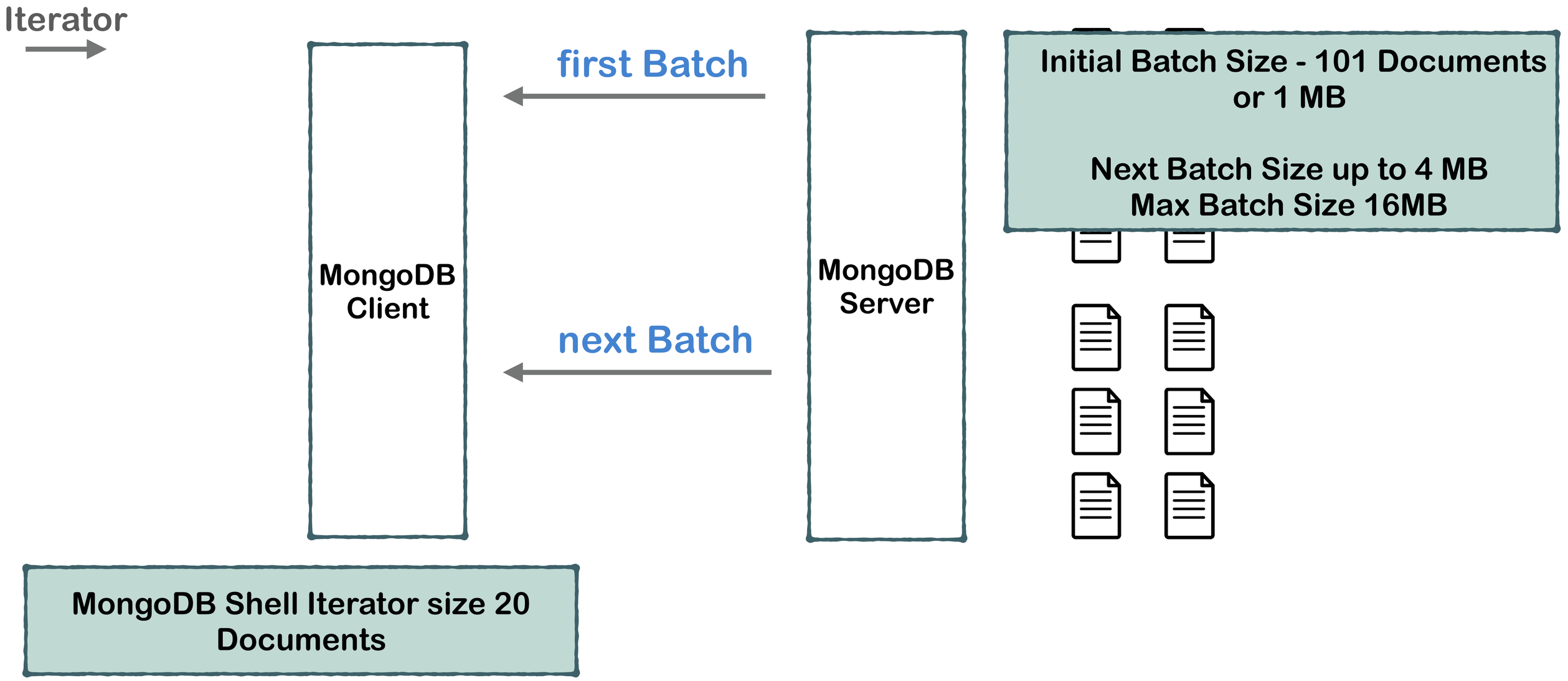
Cursor



Cursor in MongoDB



Cursor in MongoDB



Cursor Iterate Methods

- Assign Cursor to the variable

```
var cursor = db.<collection name>.find(<args>);
```

- Iterate one document

```
cursor.next()
```

- Check if cursor has next document

```
cursor.hasNext()
```

- Quantity of documents left in current batch

```
cursor.objsLeftInBatch()
```

Cursor Iterate Methods

- Iterate all documents in the cursor and push them to the array

```
cursor.toArray()
```

- Iterate all documents in the cursor and perform operation with each of them

```
cursor.forEach(<function>)
```

Count, Limit, Skip and Sort Methods

- Count number of the documents in the cursor

```
cursor.count()
```

- Limit number of the documents in the cursor

```
cursor.limit(<number>)
```

- Skip certain number of the documents

```
cursor.skip(<number>)
```

- Sort documents in the cursor

```
cursor.sort({<fieldName1>: 1, <fieldName2>:
-1, ...})
```

Limit, Skip and Sort Precedence

SORT

SKIP

LIMIT

findOne()

- Returns One Document in Extended JSON format

```
db.<collection name>.findOne(<query>, <fields>)
```

SUMMARY

- Insert methods

  ```
  insert()
  insertOne()
  insertMany()
  ```

- Cursor
- Cursor iterator
- batchSize()
- count(), limit(), sort() and skip()