Use db1

db.books.insertMany([

{name: "Hyperion", author: "Dan Simmons", quantity: 53, year: 1989, genres: ["Space Opera", "Hard Science Fiction"]},

{name: "The terror", author: "Dan Simmons", quantity: 234, year: 2007, genres: ["Horror", "Mystics", "Realism"]},

{name: "Dune chronicles", author: "Frank Herbert", quantity: 23, year: 1965, genres: ["Planetary Fiction", "Soft Science Fiction", "Hard Science Fiction"]},

{name: "Firefall", author: "Peter Watts", quantity: 34, year: 2006, genres: ["Planetary Fiction", "Hard Science Fiction"]},

{name: "Farenheit 451", author: "Ray Bradbury", quantity: 9, year: 1953, genres: ["Distopia"]},

{name: "Flashback", author: "Dan Simmons", quantity: 42, year: 2011, genres: ["Distopia", "Soft Science Fiction", "Postapocalypse", "Detective"]},

{name: "The dark tower", author: "Stephen King", quantity: 114, year: 1982, genres: ["Heroic Fantasy", "Postapocalypse", "Soft Science Fiction"]},

{name: "The running man", author: "Stephen King", quantity: 41, year: 1982, genres: ["Distopia", "Detective", "Thriller"]},

{name: "Witcher", author: "Andrzej Sopkowsy", quantity: 170, year: 1993, genres: ["Heroic Fantasy"]},

{name: "Second variety", author: "Philipp Dick", quantity: 42, year: 1953, genres: ["Postapocalypse", "Soft Science Fiction", "Thriller" ]}

]);

1. Продемонстрируйте различные комбинации индексов в MongoDB на

примере сортировки. Продемонстрируйте работу уникального индекса.

2. Продемонстрируйте работу explain для индексируемых и

неиндексируемых коллекций.

//Еще нужна неиндексируемая коллекция?

db.books.ensureIndex({quantity: 1});

db.books.find().sort({quantity: 1}).explain()

{

"queryPlanner" : {

"plannerVersion" : 1,

"namespace" : "db1.books",

"indexFilterSet" : false,

"parsedQuery" : {

},

"winningPlan" : {

"stage" : "FETCH",

"inputStage" : {

"stage" : "IXSCAN",

"keyPattern" : {

"quantity" : 1

},

"indexName" : "quantity\_1",

"isMultiKey" : false,

"multiKeyPaths" : {

"quantity" : [ ]

},

"isUnique" : false,

"isSparse" : false,

"isPartial" : false,

"indexVersion" : 2,

"direction" : "forward",

"indexBounds" : {

"quantity" : [

"[MinKey, MaxKey]"

]

}

}

},

"rejectedPlans" : [ ]

},

"serverInfo" : {

"host" : "sasha-HP-Pavilion-Laptop-13-an0xxx",

"port" : 27017,

"version" : "3.6.3",

"gitVersion" : "9586e557d54ef70f9ca4b43c26892cd55257e1a5"

},

"ok" : 1

}

db.books.ensureIndex({name: 1, year: -1})

db.books.find().sort({name: 1, year: -1}).explain()

db.books.find().sort({name: 1, year: 1}).explain()

db.books.ensureIndex({genres: -1})

db.books.find({genres: "Distopia"}).explain()

db.books.ensureIndex({name: 1, author: 1}, {unique: true})

db.books.insertOne({name: "Hyperion", author: "Dan Simmons", quantity: 101, year: 1975, genres: ["Space Opera"]})

2019-11-15T11:52:20.606+0200 E QUERY [thread1] WriteError: E11000 duplicate key error collection: db1.books index: name\_1\_author\_1 dup key: { : "Hyperion", : "Dan Simmons" }

***If MongoDB can use an index scan to obtain the requested sort order, the result will not include a SORT stage. Otherwise, if MongoDB cannot use the index to sort, the explain result will include a SORT stage.***

***Prior to MongoDB 3.0, cursor.explain() returned the scanAndOrder field to specify whether MongoDB could use the index order to return sorted results***

3. Продемонстрируйте сбор статистики для базы данных и коллекции.

> db.stats()

{

"db" : "db1",

"collections" : 1,

"views" : 0,

"objects" : 10, //число документов

"avgObjSize" : 167.7,

"dataSize" : 1677,

"storageSize" : 36864,

"numExtents" : 0,

"indexes" : 5,

"indexSize" : 151552,

"fsUsedSize" : 12562792448, //размер используемого пространства

"fsTotalSize" : 48113082368, //общий размер пространства

"ok" : 1

}

db.books.stats()

//Прочитать еще, что отбражает stats

4. Изучите веб-интерфейс MongoDB, сделайте вывод о возможностях его

применения.

5. Продемонстируйте работу профайлера для коллекций из задания 2.

> db.setProfilingLevel(2)

{ "was" : 0, "slowms" : 100, "sampleRate" : 1, "ok" : 1 }

> db.books.find({quantity: {$gt: 50}})

> db.system.profile.find().pretty()

{

"op" : "query",

"ns" : "db1.books",

"command" : {

"find" : "books",

"filter" : {

"quantity" : {

"$gt" : 50

}

},

"$db" : "db1"

},

"keysExamined" : 4, //число просмотр индексов

"docsExamined" : 4, //число проверенных документов

"cursorExhausted" : true,

"numYield" : 0,

"locks" : {

"Global" : {

"acquireCount" : {

"r" : NumberLong(2)

}

},

"Database" : {

"acquireCount" : {

"r" : NumberLong(1)

}

},

"Collection" : {

"acquireCount" : {

"r" : NumberLong(1)

}

}

},

"nreturned" : 4, //число возвращенных документов

"responseLength" : 732,

"protocol" : "op\_msg",

"millis" : 0,

"planSummary" : "IXSCAN { quantity: 1 }",

"execStats" : {

"stage" : "FETCH",

"nReturned" : 4,

"executionTimeMillisEstimate" : 0,

"works" : 5,

"advanced" : 4,

"needTime" : 0,

"needYield" : 0,

"saveState" : 0,

"restoreState" : 0,

"isEOF" : 1,

"invalidates" : 0,

"docsExamined" : 4,

"alreadyHasObj" : 0,

"inputStage" : {

"stage" : "IXSCAN",

"nReturned" : 4,

"executionTimeMillisEstimate" : 0,

"works" : 5,

"advanced" : 4,

"needTime" : 0,

"needYield" : 0,

"saveState" : 0,

"restoreState" : 0,

"isEOF" : 1,

"invalidates" : 0,

"keyPattern" : {

"quantity" : 1

},

"indexName" : "quantity\_1", //индекс

"isMultiKey" : false,

"multiKeyPaths" : {

"quantity" : [ ]

},

"isUnique" : false,

"isSparse" : false,

"isPartial" : false,

"indexVersion" : 2,

"direction" : "forward",

"indexBounds" : {

"quantity" : [

"(50.0, inf.0]"

]

},

"keysExamined" : 4,

"seeks" : 1,

"dupsTested" : 0,

"dupsDropped" : 0,

"seenInvalidated" : 0

}

},

"ts" : ISODate("2019-11-15T14:29:15.861Z"),

"client" : "127.0.0.1",

"appName" : "MongoDB Shell",

"allUsers" : [ ],

"user" : ""

}

6. Сделайте резервную копию одной из коллекций. Удалите ее и

восстановите из резервной копии.

mongodump --db db1 --collection books --out backu

2019-11-15T13:59:58.175+0200 writing db1.books to

2019-11-15T13:59:58.179+0200 done dumping db1.books (10 documents)

db.books.remove({})

mongorestore --db db1 --collection books backu/db1/books.bson

2019-11-15T14:03:13.409+0200 checking for collection data in backu/db1/books.bson

2019-11-15T14:03:13.412+0200 reading metadata for db1.books from backu/db1/books.metadata.json

2019-11-15T14:03:13.412+0200 restoring db1.books from backu/db1/books.bson

2019-11-15T14:03:13.474+0200 restoring indexes for collection db1.books from metadata

2019-11-15T14:03:13.476+0200 finished restoring db1.books (10 documents)

2019-11-15T14:03:13.477+0200 done

db.books.find()

7. Продемонстрируйте экспорт и импорт коллекций в JSON и CSV.

mongoexport --db db1 --collection books --out export/db1\_json

mongoimport --db db1 --collection books\_from\_json --file export/db1\_json

db.books\_from\_json.find()

mongoexport --db db1 --collection books --type csv --fields name,author --out export/db1.csv

mongoimport --db db1 --collection books\_from\_csv --type csv --headerline --file export/db1.csv

db.books\_from\_csv.find()