Lab-13 Index, Cursor and Schema Validation

Mongo

DB

Perform the following queries on Index, Cursor, Schema Validation, Embedded and Multivalued

Documents:

Part – A (Use collection “Stock” created in Lab-12)

1. Create an index on the company field in the stocks collection.

db.Stocks.createIndex({ company : 1})

2. Create a compound index on the sector and sales fields in the stocks collection.

db.Stocks.createIndex({ sector : 1}, { sales : 1 })

3. List all the indexes created on the stocks collection.

db.Stocks.getIndexes()

4. Drop an existing index on the company field from the stocks collection.

db.Stocks.dropIndex("company\_1")

5. Use a cursor to retrieve and iterate over documents in the stocks collection, displaying each document.

const cursor = db.Stocks.find()

cursor.forEach( doc => printjson(doc));

6. Limit the number of documents returned by a cursor to the first 3 documents in the stocks collection.

const cursor = db.Stocks.find().limit(3)

cursor.forEach( doc => printjson(doc));

7. Sort the documents returned by a cursor in descending order based on the sales field.

const cursor = db.Stocks.find().sort({sales : -1})

cursor.forEach( doc => printjson(doc));

8. Skip the first 2 documents in the result set and return the next documents using the cursor.

const cursor = db.Stocks.find().skip(2)

cursor.forEach( doc => printjson(doc))

9. Convert the cursor to an array and return all documents from the stocks collection.

cursor = db.Stocks.find().toArray()

or

var cursorArray=db.Stocks.find().toArray();printjson(cursorArray);

10. Create a collection named "Companies" with schema validation to ensure that each document

Must contains a company field (string) and a sector field (string).

db.createCollection(

'Company',

{

validator : {

$jsonSchema : {

bsonType : "object",

required : ['company', 'sector'],

properties : {

company: {

bsonType: "string",

description: "must be a string and is required"

},

sector : {

bsonType : "string",

description : "Must be a string and is required "

}

}

}

}

});

db.createCollection(

'Company',

{

validator : {

$jsonSchema : {

bsonType : "object",

required : ['company', 'sector'],

properties : {

company: {

bsonType: "string",

description: "must be a string and is required"

},

sector : {

bsonType : "string",

description : "Must be a string and is required "

}

}

}

}

});

Part – B

1. Create a collection named "Scripts" with validation for fields like eps, pe, and roe to ensure that they

are numbers and required/compulsory fields.

2. Create a collection named "Products" where each product has an embedded document for

manufacturer details and a multivalued field for categories that stores an array of category names the

product belongs to.

3.manufacturer details: The manufacturer will be an embedded document with fields like name,

country, and establishedYear.

4 categories: The categories will be an array field that holds multiple values. (i.e. Electronics,

Mobile, Smart Devices).

Part – C

1. Create a collection named “financial\_Reports” that requires revenue (a positive number) but allows

optional fields like expenses and netIncome (if provided, they should also be numbers).

2. Create a collection named "Student" where each student has name and address are embedded

document and mobilenumber and emailaddress are multivalued field that stores an array of values.