Roll No : 35040

TE IT(A)

Group : C

Batch : B

Assignment 2

Title: - Queries in MongoDB

Problem Statement: -- Execute at least 10 queries on any suitable MongoDB database that demonstrates following querying techniques:

- find and findOne (specific values)
- Query criteria (Query conditionals, OR queries, $not, Conditional semantics)
- Type-specific queries (Null, Regular expression, Querying arrays)

Requirements: -- MongoDB

Prerequisites: -- Basic Of MongoDB

Theory: --

find and findOne :

find() –
Nomatter number of documents matched, the find() method does not return null, it returns a cursor.
Eg. To select all the documents whose sid is 3
Query : db.staff.find({sid:3}).pretty();

findOne() –
The findOne() returns first document if query matches otherwise returns null.
Eg. To select first document whose sid is 3.

Query : db.staff.findOne({sid:3}).pretty();

Comparison Query Operators :

In MongoDB the conditional operators are :

(>) greater than - $gt

(<) less than - $lt

(>=) greater than equal to - $gte

(<=) less than equal to - $lte

(!=) not equal to - $ne

Syntax : db.collection_name.find({col_name:{comparison_operator:value}}).pretty();

Logical Query Operators :

Name	Description
$and	Joins query clauses with a logical AND returns all documents that match the conditions of both clauses.
$not	Inverts the effect of a query expression and returns documents that do *not* match the query expression.
$nor	Joins query clauses with a logical NOR returns all documents that fail to match both clauses.
$or	Joins query clauses with a logical OR returns all documents that match the conditions of either clause.

Regular Expression :

Provides regular expression capabilities for pattern matching strings in queries. MongoDB uses Perl compatible regular expressions (i.e. “PCRE”) version 8.42 with UTF-8 support.

To use $regex, use one of the following syntaxes:

{ <field>: { $regex: /pattern/, $options: '<options>' } }

{ <field>: { $regex: 'pattern', $options: '<options>' } }

{ <field>: { $regex: /pattern/<options> } }

1. String Match
   Eg. Find The Customers Whose Name Is “Qwe”

>db.customer_Info.find({Name:{$Regex;"QWE"}}).pretty()
eg. Find The Customers Who Have Orders In The Year 2016
>db.customer_Info.Find({Orddate:/^2016/}).Pretty()

2. Case Insensitive
   Eg. Find The Customers Whose Name Is "Tgb"
   >db.customer_Info.find({Name:{$Regex:"Tgb",$Options:"$I"}}).Pretty()
   >db.customer_Info.update({Cid:2},{$Set:{Email:Null}},{Multi:True})

3. Null
   Eg. Find The Customers Who Do Not Have An Email Id.
   >db.customer_Info.find({Email:Null}).Pretty()

4. Substring
   Eg. Find The Customers Whose Last Name Contains The Substring
   >db.customer_Info.Find({"Name.Lname":{$Regex:/MN?/i}}).Pretty()

Querying Arrays:

1) Size
   Eg. Display the records of customers who have more than two phone numbers
   > db.customer_Info.find({Phno:{"$Size":2}}).Pretty()
2) Push
   Eg. Add a new phone number to a customers record whose cid is 2.
   >db.customer_Info.update({Cid:2},{"$Push":{"Phno":8866543211}})

Conclusion : Thus, we have implemented queries in MongoDB.