NAME: Atharva Chavan
TE-A IT
ROLL NO: T1851010
PRN: 71901316L

Group C: MongoDB

Assignment No. 14

AIM: Execute at least 10 queries on any suitable MongoDB database that demonstrates following:

- $ wherequeries
- Cursors (Limits, skips, sorts, advanced queryoptions)
- Database commands

THEORY:

The Limit() Method

To limit the records in MongoDB, you need to use **limit()** method. The method accepts one number type argument, which is the number of documents that you want to be displayed.

Syntax

```
>db.COLLECTION_NAME.find().limit(NUMBER)
```

Example

Consider the collection myycol has the following data.

```
{"_id":ObjectId(5983548781331adf45ec5),"title":"MongoDB Overview"}
{"_id":ObjectId(5983548781331adf45ec6),"title":"NoSQL Overview"}
{"_id":ObjectId(5983548781331adf45ec7),"title":"Tutorials Point Overview"}
```

Following example will display only two documents while querying the document.

```
>db.mycol.find({},{"title":1,_id:0}).limit(2)
{"title":"MongoDB Overview"}
{"title":"NoSQL Overview"}
>
```

If you don't specify the number argument in **limit()** method then it will display all documents from thecollection.

MongoDBSkip() Method

Apart from limit() method, there is one more method **skip()** which also accepts number type argument and is used to skip the number of documents.

Syntax

```
>db.COLLECTION_NAME.find().limit(NUMBER).skip(NUMBER)
```

Example

Following example will display only the second document.

```
>db.mycol.find({},{"title":1,_id:0}).limit(1).skip(1)
{"title":"NoSQL Overview"}
>
```

The sort() Method

To sort documents in MongoDB, you need to use **sort()** method. The method accepts a document containing a list of fields along with their sorting order. To specify sorting order 1 and -1 are used. 1 is used for ascending order while -1 is used for descending order.

Syntax

```
>db.COLLECTION_NAME.find().sort({KEY:1})
```

Example

Consider the collection myycol has the following data.

```
{"_id":ObjectId(5983548781331adf45ec5),"title":"MongoDB Overview"}
{"_id":ObjectId(5983548781331adf45ec6),"title":"NoSQL Overview"}
{"_id":ObjectId(5983548781331adf45ec7),"title":"Tutorials Point Overview"}
```

Following example will display the documents sorted by title in the descending order.

```
>db.mycol.find({},{"title":1,_id:0}).sort({"title":-1})
{"title":"Tutorials Point Overview"}
{"title":"NoSQL Overview"}
{"title":"MongoDB Overview"}
>
```

if you don't specify the sorting preference, then **sort()** method will display the documents in ascending order.

RDBMS Where Clause Equivalents in MongoDB

To query the document on the basis of some condition, you can use following operations.

Operation	Syntax	Example	RDBMS Equivalent

Equality	{<key>:<value>}	db.mycol.find({"by":"tutorials point"}).pretty()	where by = 'tutorials point'
Less Than	{<key>:{$lt:<value>}}	db.mycol.find({"likes":{$lt:50}}).pretty()	where likes < 50
Less Than Equals	{<key>:{$lte:<value>}}	db.mycol.find({"likes":{$lte:50}}).pretty()	where likes <= 50
Greater Than	{<key>:{$gt:<value>}}	db.mycol.find({"likes":{$gt:50}}).pretty()	where likes > 50
Greater Than Equals	{<key>:{$gte:<value>}}	db.mycol.find({"likes":{$gte:50}}).pretty()	where likes >= 50
Not Equals	{<key>:{$ne:<value>}}	db.mycol.find({"likes":{$ne:50}}).pretty()	where likes != 50

Conclusion:

Executed MongoDB queries using Where,Limit, Skip conditions.

Code & Output: -

```
Atharva@BRAINMETRON:~$ mongo
MongoDB shell version: 2.6.10
connecting to: test
>db.employee.find().pretty()
{
        "_id" : 1,
        "lid" : 1,
        "lname" : "Atharva",
        "salary" : 90000,
        "address" : "pune"
}
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
{ "_id" : 5, "lid" : 5, "lname" : "suraj", "salary" : 40000 }
{ "_id" : 6, "lid" : 6, "lname" : "aditya", "salary" : 30000 }
{ "_id" : 7, "lid" : 7, "lname" : "pratiksha", "salary" : 20000 }
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.employee.find().limit(3)
{ "_id" : 1, "lid" : 1, "lname" : "Atharva", "salary" : 90000, "address" : "pune" }
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }

>db.employee.find().limit(3).pretty()
{
        "_id" : 1,
        "lid" : 1,
        "lname" : "Atharva",
        "salary" : 90000,
        "address" : "pune"
}
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
>db.employee.find().limit(5).pretty()
{
        "_id" : 1,
        "lid" : 1,
        "lname" : "Atharva",
        "salary" : 90000,
        "address" : "pune"
}
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
{ "_id" : 5, "lid" : 5, "lname" : "suraj", "salary" : 40000 }
>db.employee.find().skip(7).pretty()
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.employee.find().skip(1).pretty()
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
{ "_id" : 5, "lid" : 5, "lname" : "suraj", "salary" : 40000 }
{ "_id" : 6, "lid" : 6, "lname" : "aditya", "salary" : 30000 }
{ "_id" : 7, "lid" : 7, "lname" : "pratiksha", "salary" : 20000 }
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.employee.find().skip(1).limit(1).pretty()
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
>db.employee.find().skip(1).limit(2).pretty()
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
>db.employee.find().skip(2).limit(2).pretty()
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
>db.employee.find().skip(7).limit(1).pretty()
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.employee.find().skip(7).limit(2).pretty()
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.employee.find().limit(7).skip(1).pretty()
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
{ "_id" : 5, "lid" : 5, "lname" : "suraj", "salary" : 40000 }
{ "_id" : 6, "lid" : 6, "lname" : "aditya", "salary" : 30000 }
{ "_id" : 7, "lid" : 7, "lname" : "pratiksha", "salary" : 20000 }
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.employee.find().limit(1).skip(2).pretty()
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
>db.employee.find().skip(1).limit(2).pretty()
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
>db.employee.find().skip(2).limit(1).pretty()
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
>db.employee.find().sort(1)
error: {
        "$err" : "Can't canonicalize query: BadValue sort must be object or array",
        "code" : 17287
}
>db.employee.find().sort({"salary":1})
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
{ "_id" : 7, "lid" : 7, "lname" : "pratiksha", "salary" : 20000 }
{ "_id" : 6, "lid" : 6, "lname" : "aditya", "salary" : 30000 }
{ "_id" : 5, "lid" : 5, "lname" : "suraj", "salary" : 40000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 1, "lid" : 1, "lname" : "Atharva", "salary" : 90000, "address" : "pune" }
>db.employee.find().sort({"salary":-1})
{ "_id" : 1, "lid" : 1, "lname" : "Atharva", "salary" : 90000, "address" : "pune" }
{ "_id" : 2, "lid" : 2, "lname" : "kirti", "salary" : 80000 }
{ "_id" : 3, "lid" : 3, "lname" : "kirti", "salary" : 60000 }
{ "_id" : 4, "lid" : 3, "lname" : "aditi", "salary" : 60000 }
{ "_id" : 5, "lid" : 5, "lname" : "suraj", "salary" : 40000 }
{ "_id" : 6, "lid" : 6, "lname" : "aditya", "salary" : 30000 }
{ "_id" : 7, "lid" : 7, "lname" : "pratiksha", "salary" : 20000 }
{ "_id" : 8, "lid" : 7, "lname" : "Atharva", "salary" : 10000 }
>db.shop.insert({"itm":1,"quantity":20,"color":["red,blue"]})
```

```
WriteResult({ "nInserted" : 1 })
> show dbs;
admin   (empty)
local   0.078GB
Atharva  0.078GB
test    0.078GB
> show collections;
Atharva
employee
Atharva
shop
student
system.indexes
>db.shop.insert({"itm":2,"quantity":10,"color":["red,green"]})
WriteResult({ "nInserted" : 1 })
>db.shop.find({"color":"blue"})
>db.shop.find({"color":"blue"}).pretty()
>db.shop.insert({"itm":2,"quantity":10,"color":["red","green"]})
WriteResult({ "nInserted" : 1 })
>db.shop.insert({"itm":1,"quantity":20,"color":["red","blue"]})
WriteResult({ "nInserted" : 1 })
>db.shop.find({"color":"blue"})
{ "_id" : ObjectId("5da9b40d2990b17ac488dd74"), "itm" : 1, "quantity" : 20, "color" : [ "red",
"blue" ] }
>
```