BABU MADHAV INSTITUTE OF INFORMATION TECHNOLOGY, UTU
Integrated M.Sc.(IT)

Semester-IX

BigData Analytics

Practical Tutorial: 01

~: MongoDB Query Language (MQL):~

1) **To create database:**
   Syntax:
   use <Database_Name>**;**

2) **To display the list of databases available on the MongoDB server:**
   Command:
   show dbs;

3) **To use any database or to made any database active:**
   Syntax:
   use <Database_Name>**;**

4) **To drop database, first ensure that you are currently placed in the Database and then use:**
   Command:
   db.dropDatabase();

5) **To create a collection in database:**
   Syntax:
   db.createCollection(“ <Collection_Name> **”);**

6) **To display the list of collections in the current database:**
   Syntax:
   show collections;

7) **To drop collection:**
   Syntax:
   db . <collection_name> **. drop();**

8) **To create a collection and insert new document(s) in a collection:**
   Syntax:
   db . <collection_name> **. insert(**
    { <field1> **:** <value1> **,** <field2> **:** <value2> **});**

9) **To display/retrieve all documents from the collection:**

Syntax:

```
db . <collection_name> . find();
db . <collection_name> . find().pretty();
```

10) **To display/retrieve/search specific document(s) from the collection:**

Syntax:

```
db . <collection_name> . find(
{ <Selection_Criteria_field1> : <Selection_Criteria_value1> } );
```

11) **To display/retrieve/search only the specific field from the document(s) of the collection:**

Syntax:

```
db . <collection_name> . find( { }, { <field1> : 1 , …., <fieldn> : 1 , _id:0 } );
```

[Note: The identifier "_id" should be suppressed and NOT displayed.]

```
db . <collection_name> . find(
 { <field1> : <value1> },
 { <field1> : 1 , …., <fieldn> : 1 , _id:0 } );
```

12) **To update an existing document(s) in a collection:**

Syntax:

```
db . <collection_name> . update(
{ <Update_Criteria_field1> : <Update_Criteria_value1> },
{ $set:
      <Update_Action_field1> : <Update_Action_value1> ,
      < Update_Action_field2> : <Update_Action_value2> } );
```

13) **To delete/remove an existing document(s) from the collection:**

Syntax:

```
db . <collection_name> . remove({ }) // To remove all documents
db . <collection_name> . remove(
{ <Remove_Criteria_field1> : <Remove_Criteria_value1> } );
```

~: Relational Operators in MongoDB:~

Operator	Description
$eq	Equal to
$ne	Not equal to
$gte	Greater than or equal to
$lte	Less than or equal to
$gt	Greater than
$lt	Less than

Syntax:

```
db . <collection_name> . find(
 {
      <field> :
            { $<operator> : ‘ <'value2> ’ }
 )
```

~: Other Operators in Mongodb:~

14) IN:

Syntax:

```
db . <collection_name> . find(
 {
     <field1>:
                   { $in: [ ‘<value1>’ , ‘<value2>’ , ….., ‘<valuen>’ ] }
 })
```

15) NIN (NOT IN):

Syntax:

```
db . <collection_name> . find(
 {
      <field1>:
      { $nin: [ ‘<value1>’ , ‘<value2>’ , ….., ‘<valuen>’ ] }
 })
```

16) Patten Matching:

✓ To find the documents from the collections begins with…

Syntax:

```
db . <collection_name> . find( { <field1> : / ^<letter> /  }  )
```

- ✓ To find the documents from the collections ends with…

  Syntax:

  db . <collection_name> **. find(** { <field1> : / <letter>**$** / **}** **)**

- ✓ To find the documents from the collections for any position with…

  Syntax:

  db . <collection_name> **. find({** <field1> **: /** **<letter>** / **}** **)**

  OR

  db . <collection_name> **. find({** <field1> **: /.*<letter>.*** / **}** **)**

  OR

  db . <collection_name> **. find({** <field1> **: {$regex:** "**<letter>**" } } **)**

~: Logical Operators in Mongodb:~

17) AND:

Syntax:

```
db . <collection_name> . find(
 {
     $and: [
                 {<field1> : <value1> } , { <field2> : <value2> }
            ]
 })
```

```
db . <collection_name> . find(
 {
     {<field1> : <value1> } , { <field2> : <value2> }
 })
```

18) OR:

Syntax:

```
db . <collection_name> . find(
 {
     $or: [
                 {<field1> : <value1> } , { <field2> : <value2> }
            ]
 })
```

19) **NOR (For NOT Operation):**

Syntax:

```
db . <collection_name> . find(
 {
     $nor: [
                  {<field1> : <value1> } , { <field2> : <value2> }
             ]
 })
```

20) **NOT (is used with relational operators):**

Syntax:

```
db . <collection_name> . find(
 {
     $not: [
                  {<field1> : <value1> } , { <field2> : <value2> }
             ]
 })
```

~: Other Methods in MongoDB:~

21) **LIMIT()**

Syntax:

```
db . <collection_name> . find() . limit(<number>)
```

22) **SKIP()**

Syntax:

```
db . <collection_name> . find() . limit(<number>) . skip(<number>)
```

23) **SORT()**

Syntax:

```
db . <collection_name> . find() . sort( { <key> : 1 } )
db . <collection_name> . find() . sort( { <key> : -1 } )
```

24) **COUNT()**

Syntax:

```
db . <collection_name> . count()
db . <collection_name> . count( { <key> : '<value>'} )
```

~: Aggregate Function in MongoDB:~

Expression	Description
$sum	Adds up the definite values of every document of a collection.
$avg	Computes the average values of every document of a collection.
$min	Finds and returns the minimum of all values from within a collection.
$max	Finds and returns the maximum of all values from within a collection.
$push	Feeds in the values to an array in the associated document.
$first	Fetches out the first document.
$last	Fetches out the last document.
$addToSet	Feeds in the values to an array without duplication.

Syntax:

```
db . <collection_name> . aggregate(
    { $group: { _id:0 , <Expression_Object_Name> : { $<expression> : ' $<key> ' } } } );
```

For particular:

```
db . <collection_name> . aggregate(
    { $match: { <key> : ' <value> ' } },
    { $group: { _id:0 , <Expression_Object_Name> : {$<expression> : ' $<key> ' } } } );
```

~: Arrays:~

Syntax:

```
db . <collection_name> . insert(
    { _id:<value> , <array_name> : [ <element1> , <element2> , … ] } );
```