To export the file from mysql -->
SELECT * FROM author INTO OUTFILE '<your path>\Author.csv' FIELDS TERMINATED BY ','
ENCLOSED BY "" LINES TERMINATED BY '\n';

Import in MongoDB
mongoImport --db dbda --collection Book --type csv --file "<your path>\book.csv" --headerline

---

JSON <- JavaScript Object Notation

1) To see the list of databases
show databases

2) To choose the database
use lms

3) To know the current database
db

4) To create database
USE dbda

5) To see the list of collection
show collections

5) Inserting document inside collection
db.Book.insert({Name:'2 States'})
db.Book.insert({Name:'Lashkar', Pages:120})

6) To list the documents from a collection
db.Book.find()

7) To add multiple documents using single command to a collection
Note: Use [ ]
db.Book.insert([{Book:'Spider'},{author:'Mukul Deva'}])

8) To create the collection directly and insert the documents

db.Publisher.insert({PName:'Sterlings',Country:'Bharat'})
--> Note: Publisher collection is not precreated

9) To drop collection

db.Book.drop() --> collection Book will get dropped.

10) Limiting documents while listing

db.Book.find().limit(1)

11) Skipping the documents

db.Book.find().skip(3).pretty() <-- will skip the first three documents in result.

12) Showing only restricted key-value pairs

db.Book.find({},{BookName:1}).pretty() <-- will display BookName (and id as well)

db.Book.find({},{_id:0,BookName:1}).pretty()

Note: 0 <-- excluded from result , 1 <-- included in result

db.Author.find({},{_id:0,AuthorName:1})
db.Author.find({},{_id:0,AuthorName:1,Nationality:1}) <-- Two key-values included in result

13) Dropping a database

Use MoreBooks <-- dbname

db.dropDatabase() <-- drop the database

14) Showing data based on ONE codition

db.Author.find({Nationality:'UK'}).pretty()
db.Author.find({Gender:'F'})

db.Author.find({Gender:'M'})

15) Showing data based on MORE THAN ONE codition (AND)
db.Author.find({Gender:'M',Nationality:'USA'}).pretty()
db.Employee.find({Sex:'M',DeptId:'D003'}).pretty()

16) Showing only selected key-values

db.Author.find({Gender:'M'},{_id:0,Gender:1}).pretty()

16) Showing only selected key-values - With Condition

db.Book.find({Catogery:'Fiction'},{_id:0,BookName:1,Catogery:1,Cost:1}).pretty()

17) Using OR condition

db.Author.find({$or:[{Nationality:'UK'},{Nationality:'USA'}]}).pretty()

db.Author.find({$or:[{Nationality:'UK'},{Nationality:'USA'}]},{_id:0,Nationality:1})

18) Multiple OR conditions

db.Author.find({Nationality:{$in:['UK','USA','India']}})
db.Author.find({Nationality:{$in:['UK','USA','India']}},{_id:0,Nationality:1})

18A) Multiple OR conditions (other than these conditons)

db.Author.find({Nationality:{$nin:['UK','USA','India']}})
db.Author.find({Nationality:{$nin:['UK','USA','India']}},{_id:0,Nationality:1})

19) Data Sorting

db.Book.find({},{_id:0,Cost:1}).sort({Cost:1}).pretty() 1<-- ASC
db.Book.find({},{_id:0,Cost:1}).sort({Cost:1}).pretty() -1<-- Desc

20) > , < , >= , <=

db.Book.find({Cost:{$gt:300}}).pretty()
db.Book.find({Cost:{$lt:300}}).pretty()

db.Book.find({Cost:{$lt:300}},{_id:0,Cost:1}).pretty()

>100 but < 150
db.Book.find({Cost:{$gt:100,$lt:150}},{_id:0,Cost:1}).pretty()

21) Use of Distinct()

db.Book.distinct('Catogery')
db.Author.distinct('Nationality')

22) Use of Group By

Grouping on Status and taking count of same
db.Member.aggregate([{$group:{_id:'$Status',StatusCount:{$sum:1}}}])
db.Member.aggregate([{$group:{_id:'$Sex',GenderCount:{$sum:1}}}])
db.Employee.aggregate([{$group:{_id:'$Designation',DesignationCount:{$sum:1}}}])

Grouping on Catogery and taking average cost
db.Book.aggregate([{$group:{_id:'$Catogery',AvgCost:{$avg:'$Cost'}}}])

Grouping on Catogery and taking maximum pages
db.Book.aggregate([{$group:{_id:'$Catogery',MaxPages:{$max:'$NoOfPages'}}}])

23) Update , UpdateMany

All document
----------pd---
db.Location.update({},{$set:{Country:'Bharat'}}) <-- Update only the first document
db.Location.updateMany({},{$set:{Country:'Bharat'}}) <- Update ALL matching documents
{} < -- All document

Updating based on Condition
db.Author.update({Nationality:'India'},{$set:{Nationality:'Bharat'}})
db.Author.updateMany({Nationality:'India'},{$set:{Nationality:'Bharat'}})

Updating document/Key-Value
db.Author.find({Nationality:'UK'},{_id:0,Nationality:1,Gender:1,Comment:1})

db.Author.updateMany({AuthorId:'A008'},{$set:{Nationality:'Singapore'}}) <-- Based on AuthorId

24) Use of UNSET with Update

db.Location.updateMany({},{$unset:{Country:'Bharat'}})

Removing Key-Value from a SPECIFIC document
db.Location.update({LocationId:'L002'},{$unset:{City:'Pune'}})

25) Use of Remove <-- Removing the ENTIRE document based on condition
db.Book1.remove({Catogery:'Fiction'})

26) Delete & DeleteMany <-- Working on DOCUMENT LEVEL

db.Book.deleteOne({Catogery:'Fiction'})
db.Book.deleteMany({Catogery:'Fiction'})

27) Find And Modify

db.Employee.findAndModify({query:{Designation:'Accountant'},update:{$inc:{Salary:-1}},new:tru
e})

28) bulk insert
---
var bulk = db.abc.initializeUnorderedBulkOp();
bulk.insert({ first_name: "Sachin", last_name: "Tendulkar" });
bulk.insert({ first_name: "Virender", last_name: "Sehwag" });
bulk.insert({ first_name: "Shikhar", last_name: "Dhawan" });
bulk.insert({ first_name: "Mohammed", last_name: "Shami" });
bulk.insert({ first_name: "Shreyas", last_name: "Iyer" });
bulk.execute();