

1) Create a Database called music.

> use music

switched to db music

```
ukistu01@ukipc01: ~  
http://docs.mongodb.org/  
Questions? Try the support group  
http://groups.google.com/group/mongodb-user  
Server has startup warnings:  
2019-01-08T11:47:51.763+0530 I STORAGE [initandlisten]  
2019-01-08T11:47:51.763+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS  
filesystem is strongly recommended with the WiredTiger storage engine  
2019-01-08T11:47:51.763+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten]  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten]  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten]  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten] ** We suggest setting it to 'never'  
2019-01-08T11:47:53.818+0530 I CONTROL [initandlisten]  
> use music  
switched to db music  
>
```

2) Create a collection called songdetails.

> db.createCollection("songdetails")

{ "ok": 1 }

```
ukistu01@ukipc01: ~  
[sudo] password for ukistu01:  
MongoDB shell version v3.6.9  
connecting to: mongodb://127.0.0.1:27017  
Implicit session: session { "id" : UUID("7bca0e89-007f-40e4-be0c-2cba4ce429d2") }  
MongoDB server version: 3.6.9  
Server has startup warnings:  
2019-01-09T21:19:54.058+0530 I STORAGE [initandlisten]  
2019-01-09T21:19:54.058+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine  
2019-01-09T21:19:54.058+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem  
2019-01-09T21:19:56.283+0530 I CONTROL [initandlisten]  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** WARNING: You are running this process as the root user, which is not recommended.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten]  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --bind_ip_all to  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** bind to all interfaces. If this behavior is desired, start the  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to disable this warning.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten]  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten]  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten] ** We suggest setting it to 'never'  
2019-01-09T21:19:56.284+0530 I CONTROL [initandlisten]  
> use music  
switched to db music  
> db.createCollection("songdetails")  
{ "ok" : 1 }  
>
```

3) Create the above 5 song documents.

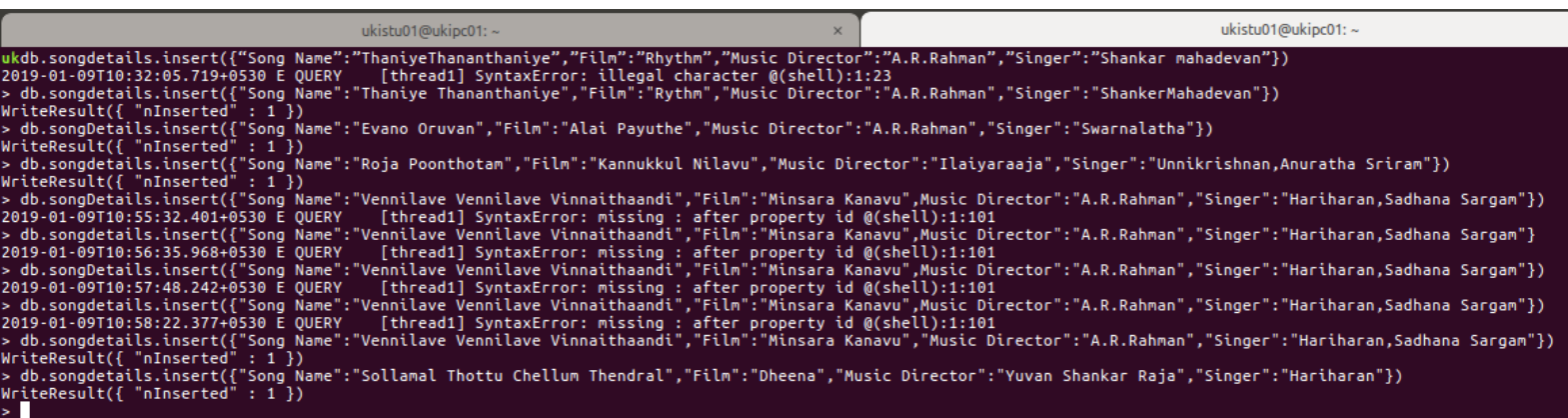
```
>db.songdetails.insert({"Song Name":"ThaniyeThananthaniye","Film":"Rhythm","Music
Director":"A.R.Rahman","Singer":"Shankar mahadevan"})
WriteResult({ "nInserted" : 1 })
```

```
>db.songdetails({"Song Name":"Evano Oruvan","Film":"Alai Payuthey","Music
Director":"A.R.Rahman","Singer":"Swarnalatha"})
WriteResult({ "nInserted" : 1 })
```

```
>db.songdetails({"Song Name":"Roja poonthotam","Film":"Kannukul nilavuy","Music
Director":"Ilaiyaraaja","Singer":"Unnikrishnan,Anuradha Sriram"})
WriteResult({ "nInserted" : 1 })
```

```
>db.songdetails({"Song Name":"Vennilavae Vennilavae Vinnaitaandi","Film":"Minsara Kanavu","Music
Director":"A.R.Rahman","Singer":"Hariharan ,Sadhana Sargam"})
WriteResult({ "nInserted" : 1 })
```

```
>db.songdetails({"Song Name":"Sollamal Thottu Chellum Thendral","Film":"Dheena","Music
Director":"Yuvan Shankar Raja","Singer":"Hariharan "})
WriteResult({ "nInserted" : 1 })
```



```
ukdb.songdetails.insert({"Song Name":"ThaniyeThananthaniye","Film":"Rhythm","Music Director":"A.R.Rahman","Singer":"Shankar mahadevan"})
2019-01-09T10:32:05.719+0530 E QUERY [thread1] SyntaxError: illegal character @(shell):1:23
> db.songdetails.insert({"Song Name":"Thaniye Thananthaniye","Film":"Rythm","Music Director":"A.R.Rahman","Singer":"ShankerMahadevan"})
WriteResult({ "nInserted" : 1 })
> db.songDetails.insert({"Song Name":"Evano Oruvan","Film":"Alai Payuthe","Music Director":"A.R.Rahman","Singer":"Swarnalatha"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Roja Poonthotam","Film":"Kannukul Nilavu","Music Director":"Ilaiyaraaja","Singer":"Unnikrishnan,Anuratha Sriram"})
WriteResult({ "nInserted" : 1 })
> db.songDetails.insert({"Song Name":"Vennilave Vennilave Vinnaitaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:32:05.719+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaitaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:56:35.968+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songDetails.insert({"Song Name":"Vennilave Vennilave Vinnaitaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:57:48.242+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaitaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:58:22.377+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaitaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Sollamal Thottu Chellum Thendral","Film":"Dheena","Music Director":"Yuvan Shankar Raja","Singer":"Hariharan"})
WriteResult({ "nInserted" : 1 })
>
```

4) List all documents created.

```
> db.songdetails.find()
{ "_id" : ObjectId("5c3581ea34ce5ee63a0e9a93"), "Song Name" : "Thaniye Thananthaniye", "Film" :
"Rythm", "Music Director" : "A.R.Rahman", "Singer" : "ShankerMahadevan" }
{ "_id" : ObjectId("5c35848934ce5ee63a0e9a95"), "Song Name" : "Roja Poonthotam", "Film" :
"Kannukul Nilavu", "Music Director" : "Ilaiyaraaja", "Singer" : "Unnikrishnan,Anuratha Sriram" }
```

```
{ "_id" : ObjectId("5c358ec434ce5ee63a0e9a96"), "Song Name" : "Vennilave Vennilave
Vinnaihaandi", "Film" : "Minsara Kanavu", "Music Director" : "A.R.Rahman", "Singer" :
"Hariharan,Sadhana Sargam" }
{ "_id" : ObjectId("5c35948d34ce5ee63a0e9a97"), "Song Name" : "Sollamal Thottu Chellum Thendral",
"Film" : "Dheena", "Music Director" : "Yuvan Shankar Raja", "Singer" : "Hariharan" }
{ "_id" : ObjectId("5c35b06f34ce5ee63a0e9a98"), "Song Name" : "Evano Oruvan", "Film" : "Alai
Payuthe", "Music Director" : "A.R.Rahman", "Singer" : "Swarnalatha" }
>
```

```
ukdb.songdetails.insert({"Song Name":"ThaniyeThananthaniye","Film":"Rhythm","Music Director":"A.R.Rahman","Singer":"Shankar mahadevan"})
2019-01-09T10:32:05.719+0530 E QUERY [thread1] SyntaxError: illegal character @(shell):1:23
> db.songdetails.insert({"Song Name":"Thaniye Thananthaniye","Film":"Rythm","Music Director":"A.R.Rahman","Singer":"ShankerMahadevan"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Evano Oruvan","Film":"Alai Payuthe","Music Director":"A.R.Rahman","Singer":"Swarnalatha"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Roja Poonthotan","Film":"Kannukkul Nilavu","Music Director":"Ilaiyaraaja","Singer":"Unnikrishnan,Anuratha Sriram"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaihaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:55:32.401+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaihaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:56:35.968+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaihaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:57:48.242+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaihaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
2019-01-09T10:58:22.377+0530 E QUERY [thread1] SyntaxError: missing : after property id @(shell):1:101
> db.songdetails.insert({"Song Name":"Vennilave Vennilave Vinnaihaandi","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Sollamal Thottu Chellum Thendral","Film":"Dheena","Music Director":"Yuvan Shankar Raja","Singer":"Hariharan"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.find()
{ "_id" : ObjectId("5c3581ea34ce5ee63a0e9a93"), "Song Name" : "Thaniye Thananthaniye", "Film" : "Rythm", "Music Director" : "A.R.Rahman", "Singer" : "ShankerMahadevan" }
{ "_id" : ObjectId("5c35848934ce5ee63a0e9a95"), "Song Name" : "Roja Poonthotan", "Film" : "Kannukkul Nilavu", "Music Director" : "Ilaiyaraaja", "Singer" : "Unnikrishnan,Anuratha Sriram" }
{ "_id" : ObjectId("5c358ec434ce5ee63a0e9a96"), "Song Name" : "Vennilave Vennilave Vinnaihaandi", "Film" : "Minsara Kanavu", "Music Director" : "A.R.Rahman", "Singer" : "Hariharan,Sadhana Sargam" }
{ "_id" : ObjectId("5c35948d34ce5ee63a0e9a97"), "Song Name" : "Sollamal Thottu Chellum Thendral", "Film" : "Dheena", "Music Director" : "Yuvan Shankar Raja", "Singer" : "Hariharan" }
> db.songdetails.insert({"Song Name":"Evano Oruvan","Film":"Alai Payuthe","Music Director":"A.R.Rahman","Singer":"Swarnalatha"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.find()
{ "_id" : ObjectId("5c3581ea34ce5ee63a0e9a93"), "Song Name" : "Thaniye Thananthaniye", "Film" : "Rythm", "Music Director" : "A.R.Rahman", "Singer" : "ShankerMahadevan" }
{ "_id" : ObjectId("5c35848934ce5ee63a0e9a95"), "Song Name" : "Roja Poonthotan", "Film" : "Kannukkul Nilavu", "Music Director" : "Ilaiyaraaja", "Singer" : "Unnikrishnan,Anuratha Sriram" }
{ "_id" : ObjectId("5c358ec434ce5ee63a0e9a96"), "Song Name" : "Vennilave Vennilave Vinnaihaandi", "Film" : "Minsara Kanavu", "Music Director" : "A.R.Rahman", "Singer" : "Hariharan,Sadhana Sargam" }
{ "_id" : ObjectId("5c35948d34ce5ee63a0e9a97"), "Song Name" : "Sollamal Thottu Chellum Thendral", "Film" : "Dheena", "Music Director" : "Yuvan Shankar Raja", "Singer" : "Hariharan" }
{ "_id" : ObjectId("5c35b06f34ce5ee63a0e9a98"), "Song Name" : "Evano Oruvan", "Film" : "Alai Payuthe", "Music Director" : "A.R.Rahman", "Singer" : "Swarnalatha" }
```

5) List A.R.Rahman's songs.

```
db.songdetails.find({"Music Director":"A.R.Rahman"},{"Song Name":1,_id:0}).pretty()
{ "Song Name" : "ThaniyeThananthaniye" }
```

```
ukdb.songdetails.find({"Music Director":"A.R.Rahman"},{"Music Director":1,_id:0}).pretty()
2019-01-11T11:29:32.898+0530 E QUERY [thread1] SyntaxError: missing : after p
property id @(shell):1:32
> db.songdetails.find({"Music Director":"A.R.Rahman"},{"Music Director":1,_id:0}).pretty()
2019-01-11T11:34:30.999+0530 E QUERY [thread1] SyntaxError: missing : after p
property id @(shell):1:48
> db.songdetails.find({"Music Director":"A.R.Rahman"},{"Music Director":1,_id:0}).pretty()
> db.songdetails.find({"Music Director":"A.R.Rahman"},{"Music Director":1,_id:0}).pretty()
{ "Music Director" : "A.R.Rahman" }
{ "Music Director" : "A.R.Rahman" }
{ "Music Director" : "A.R.Rahman" }
> db.songdetails.find({"Music Director":"A.R.Rahman"},{"Song Name":1,_id:0}).pretty()
{ "Song Name" : "ThaniyeThananthaniye" }
{ "Song Name" : "Evano Oruvan" }
{ "Song Name" : "Vennilave Vennilave" }
```

```
{ "Song
Name" :
"Evano
Oruvan" }
{ "Song
Name" :
"Vennilave
Vennilave" }
```


6) List ilaiyaraaja songs sung by Unnikrishnan.

```
db.songdetails.find({"Music Director":"Ilaiyaraja","Singer":"Unnikrishnan"},{"Song Name":1})
{ "_id" : ObjectId("5c3848cb65c2f61a943beee6"), "Song Name" : "Roja poonthotam" }
```

```
ukistu01@ukipc01: ~
ukistu01@ukipc01:~$ sudo mongo
[sudo] password for ukistu01:
MongoDB shell version v3.6.9
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("379605dc-727e-4d72-8ab1-f75f73039549") }
MongoDB server version: 3.6.9
Server has startup warnings:
2019-01-14T07:34:22.406+0530 I STORAGE [initandlisten]
2019-01-14T07:34:22.406+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly
2019-01-14T07:34:22.406+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/p
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten]
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** Read and write access to data and con
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** WARNING: You are running this process as the r
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten]
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** Remote systems will be unable to conn
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <add
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** addresses it should serve responses t
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** bind to all interfaces. If this behav
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to d
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten]
20db.songdetails.find({"Music Director":"Ilaiyaraja"},{"Singer":"Unnikrishnan"},)
2019-01-14T07:43:26.982+0530 E QUERY [thread1] SyntaxError: expected expression, got ')' @(shell):1:78
> db.songdetails.find({"Music Director":"Ilaiyaraja"},{"Singer":"Unnikrishnan"},{"Song Name":1})
{ "id" : ObjectId("5c3848cb65c2f61a943beee6"), "Singer" : [ "Unnikrishnan", "Anuradha Sriram" ] }
> db.songdetails.find({"Music Director":"Ilaiyaraja"},{"Singer":"Unnikrishnan"},{"Song Name":1})
{ "_id" : ObjectId("5c3848cb65c2f61a943beee6"), "Song Name" : "Roja poonthotam" }
>
```

7) Delete the song which you don't like.

```
db.songdetails.remove({"Song Name":"Vennilave Vennilave","Film":"Minsara Kanavu","Music
Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
WriteResult({ "nRemoved" : 1 })
>
```

```
ukistu01@ukipc01: ~
ukistu01@ukipc01: ~
{ "Song Name" : "ThaniyeThananthaniye" }
{ "Song Name" : "Evano Oruvan" }
{ "Song Name" : "Vennilave Vennilave" }
>
> db.songdetails.find({"Music Director":"Ilaiyaraaja"},{"Song Name":1,_id:0}).so
rt("Singer":"Unnikrishnan").pretty()
2019-01-11T12:10:00.294+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:89
> db.songdetails.find({"Music Director":"Ilaiyaraja"},{"Song Name":1,_id:0}).sor
t("Singer":"Unnikrishnan").pretty()
2019-01-11T12:11:15.447+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:88
> db.songdetails.find({"Music Director":"Ilaiyaraja"},{"Song Name":1,_id:0}).sor
t("Singer":"Unnikrishnan").pretty()
2019-01-11T12:12:42.637+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:88
> db.songdetails.find({"Music Director":"Ilaiyaraja"},{"Song Name":1,_id:0}).sor
t("Singer":"Unnikrishnan":1).pretty()
2019-01-11T12:16:00.316+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:88
> db.songdetails.remove({"Song Name":"Vennilave Vennilave","Film":"Minsara Kanav
u","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
WriteResult({ "nRemoved" : 1 })
>
```

```
db.songdetails.insert({"Song Name":"Vennilave Vennilave","Film":"Minsara Kanavu","Music  
Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})  
WriteResult({ "nInserted" : 1 })
```

```
>
> db.songdetails.find({"Music Director":"Ilaiyaraaja"},{"Song Name":1,_id:0}).sort("Singer":"Unnikrishnan")pretty()
2019-01-11T12:10:00.294+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:89
> db.songdetails.find({"Music Director":"Ilaiyaraaja"},{"Song Name":1,_id:0}).sort("Singer":"Unnikrishnan")pretty()
2019-01-11T12:11:15.447+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:88
> db.songdetails.find({"Music Director":"Ilaiyaraaja"},{"Song Name":1,_id:0}).sort("Singer":"Unnikrishnan").pretty()
2019-01-11T12:12:42.637+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:88
> db.songdetails.find({"Music Director":"Ilaiyaraaja"},{"Song Name":1,_id:0}).sort("Singer":"Unnikrishnan":1).pretty()
2019-01-11T12:16:00.316+0530 E QUERY [thread1] SyntaxError: missing ) after a
rgument list @(shell):1:88
> db.songdetails.remove({"Song Name":"Vennilave Vennilave","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
WriteResult({"nRemoved" : 1 })
> db.songdetails.insert({"Song Name":"Vennilave Vennilave","Film":"Minsara Kanavu","Music Director":"A.R.Rahman","Singer":"Hariharan,Sadhana Sargam"})
WriteResult({"nInserted" : 1 })
>
```

9) List Songs sung by Hariharan from Minsara kanavu film.

```

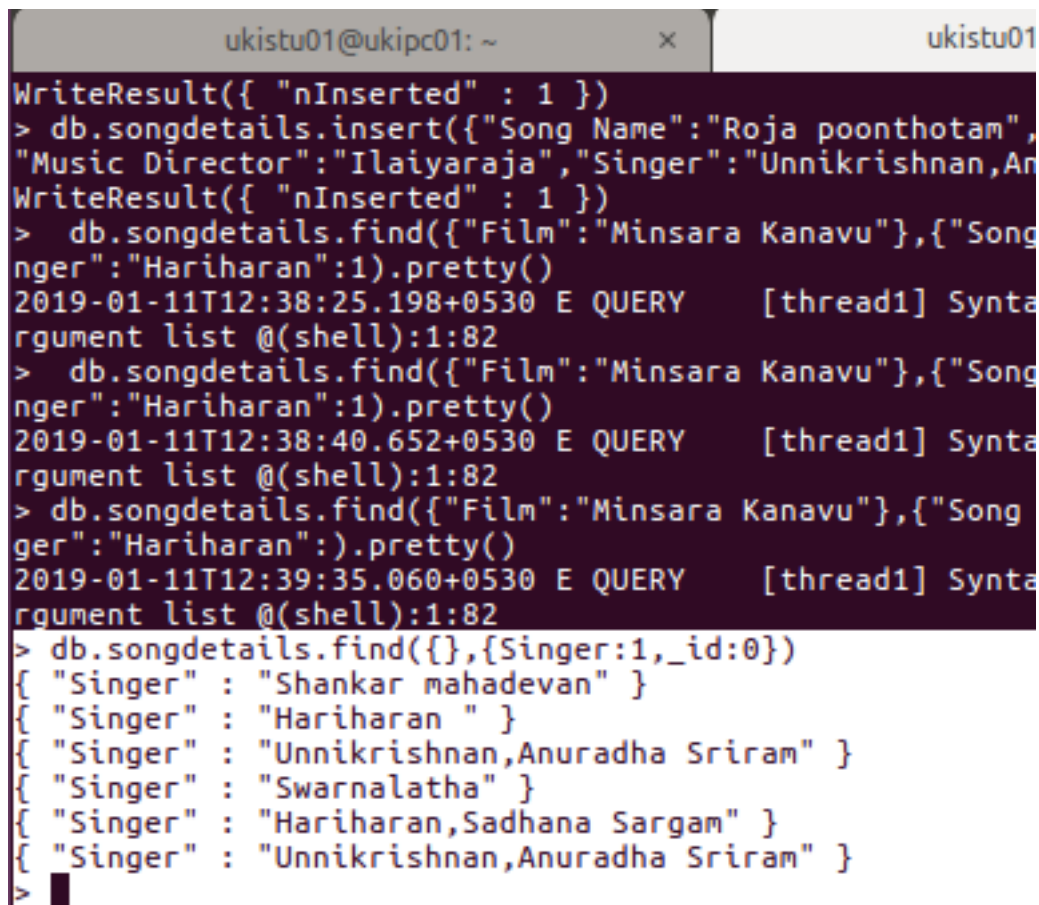
xistuto0@bukipco1:~$ sudo mongo
ukistu>use bukipco1:-$ sudo mongo
[ukistu] password for ukistu0@1:
MongoDB shell version v3.6.9
Connecting to mongodb://127.0.0.1:27017
Implicit session: session { "id": "UIDWIT["379605dc-727e-4d72-8ab1-f1f7379039549"] }
MongoDB server version: 3.6.9
Server has startup warnings:
2019-01-14T07:34:22.406+0530 I STORAGE [initandlisten]
2019-01-14T07:34:22.406+0530 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is
2019-01-14T07:34:22.406+0530 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten]
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** Read and write access to da
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** WARNING: You are running this process
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten]
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to local
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** Remote systems will be unabl
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** Start the server with -bind
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** addresses it should serve r
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** bind to all interfaces. If
2019-01-14T07:34:24.649+0530 I CONTROL [initandlisten] ** server with --bind_ip 127.0
20db.songdetails.find(("Music Director":{"$regex":"Unnikrishnan"}))
2019-01-14T07:43:26.982+0530 E QUERY [threadrid] SyntaxError: expected expression, got ')' @($
> db.songdetails.find(("Music Director":{"$regex":"Unnikrishnan"},"(Song Name":{"
    "id" : ObjectId("5c3b48cb65cf2f1a943bee6"), "Singer" : [ "Unnikrishnan", "Anuradha Sritran
    > db.songdetails.find(("Music Director":{"$regex":"Unnikrishnan"},"(Song Name":{"
    "id" : ObjectId("5c3b48cb65cf2f1a943bee6"), "Singer" : [ "Roja poonhotan" ]
    > db.songdetail.remove({"Song Name":"Vennilavae Vennilavae Vinnaithaandi","Filn":"Minsara Kanavu
...
...
> db.songdetail.remove({"Song Name":"Vennilavae Vennilavae Vinnaithaandi","Filn":"Minsara Kanavu
WriteResult({ "nRemoved" : 0 })
> db.songdetails.remove({"Song Name":"Vennilavae Vennilavae Vinnaithaandi","Filn":"Minsara Kanavu
WriteResult({ "nRemoved" : 0 })
> db.songdetails.find()
{ "_id" : ObjectId("5c361ae5219eb8e0a9cdabaaf"), "Song Name" : "ThaniyeThananthaniye", "Film" :
{ "_id" : ObjectId("5c361ae5219eb8e0a9cdabaaf"), "Song Name" : "ThaniyeThananthaniye", "Film" :
{ "_id" : ObjectId("5c3620ba6ef534fd3abb8b59"), "Song Name" : "SolLena! Thottu Chellun Thendral
{ "_id" : ObjectId("5c3620ba6ef534fd3abb8b59"), "Song Name" : "Evano Oruvan", "Film" : "Alai Pa
{ "_id" : ObjectId("5c3638a3b65cf2f1a943bee6"), "Song Name" : "Vennilave Vennilave", "Film" : "
{ "_id" : ObjectId("5c3638a3b65cf2f1a943bee6"), "Song Name" : "Roja poonhotan", "Film" : "Kann
m" ] }
> db.songdetails.remove({"Song Name" : "Vennilave Vennilave", "Film" : "Minsara Kanavu"}, "Music
WriteResult({ "nRemoved" : 1 })
> db.songdetails.remove({"Song Name" : "Vennilave Vennilave", "Film" : "Minsara Kanavu", "Music
WriteResult({ "inserted" : 1 })
> db.songdetails.find({"Singer":"Haritharan"},"(Filn":"Minsara Kanavu)","(Song Name":{"
2019-01-14T08:02:55.035+0530 E QUERY [threadrid] TypeError: db.songdetails is not a function:
M(shell):1:1
> db.songdetails.find({"Singer":"Haritharan","Film":"Minsara Kanavu"},"(Song Name":{"
    "id" : ObjectId("5c3bf405eeffb701716d17ea"), "Song Name" : "Vennilave Vennilave")

```

```
> db.songdetails.find({"Singer":"Hariharan","Film":"Minsara Kanavu"},"Song Name":1})
{ "_id" : ObjectId("5c3bf4056efbb707176d17ea"), "Song Name" : "Vennilave Vennilave" }
```

10)List out the singers' names in your document.

```
> db.songdetails.find({}, {Singer:1, _id:0})
{ "Singer" : "Shankar mahadevan" }
{ "Singer" : "Hariharan " }
{ "Singer" : "Unnikrishnan,Anuradha Sriram" }
{ "Singer" : "Swarnalatha" }
{ "Singer" : "Hariharan,Sadhana Sargam" }
{ "Singer" : "Unnikrishnan,Anuradha Sriram" }
```



```
ukistu01@ukipc01: ~ x ukistu01
WriteResult({ "nInserted" : 1 })
> db.songdetails.insert({"Song Name":"Roja poonthotam",
"Music Director":"Ilaiyaraja","Singer":"Unnikrishnan,Anuradha Sriram"})
WriteResult({ "nInserted" : 1 })
> db.songdetails.find({"Film":"Minsara Kanavu"},"Song Name":1).pretty()
2019-01-11T12:38:25.198+0530 E QUERY [thread1] SyntaxError: argument list @(shell):1:82
> db.songdetails.find({"Film":"Minsara Kanavu"},"Song Name":1).pretty()
2019-01-11T12:38:40.652+0530 E QUERY [thread1] SyntaxError: argument list @(shell):1:82
> db.songdetails.find({"Film":"Minsara Kanavu"},"Song Name":1).pretty()
2019-01-11T12:39:35.060+0530 E QUERY [thread1] SyntaxError: argument list @(shell):1:82
> db.songdetails.find({}, {Singer:1, _id:0})
{ "Singer" : "Shankar mahadevan" }
{ "Singer" : "Hariharan " }
{ "Singer" : "Unnikrishnan,Anuradha Sriram" }
{ "Singer" : "Swarnalatha" }
{ "Singer" : "Hariharan,Sadhana Sargam" }
{ "Singer" : "Unnikrishnan,Anuradha Sriram" }
>
```