CLASS: CSE7345

Name: Randeep Hanspal

SMUID: 47812509

Quest6: Mongo1

In [16]:
```
import pymongo
import json
```

In [17]:
```
from pymongo import MongoClient
```

In [18]:
```
client=MongoClient('mongodb://rhanspal:ar43hRn4@smgo7db01.smu.edu:27017/rhansp
aldb')
```

In [19]:
```
db=client.rhanspaldb
```

In [20]:
```
#show all collections except system collections
collection = db.collection_names(include_system_collections=False)
for collect in collection:
    print (collect)
```

```
zorro
```

In [21]:
```
# Create some documents for insertion
posts =[ {"name": "rollo", "topic": "computing", "post":"python rules" },
{"name": "wally", "topic": "computing", "post":"c++ rules" },
{"name": "marla", "topic": "physics", "post":"e=mc**2" }
]
```

In [22]:
```
for p in posts:
    db.blog.insert(p)
```

```
/usr/local/es6/lib/python2.7/site-packages/ipykernel_launcher.py:2: Deprecati
onWarning: insert is deprecated. Use insert_one or insert_many instead.
```

In [23]:
```
# load documents from a json file
lines = open('worldcup.json').readlines()
```

In [24]:
```
for line in lines:
    mydict = json.loads(line)
    db.blog.insert(mydict)
```

```
/usr/local/es6/lib/python2.7/site-packages/ipykernel_launcher.py:3: Deprecati
onWarning: insert is deprecated. Use insert_one or insert_many instead.
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
```

In [25]:
```
#find() returns cursor - iterate to show collection blog
for p in db.blog.find():
    print(p)
```

```
{u'topic': u'computing', u'post': u'python rules', u'_id': ObjectId('5bb16e6e
58b71ca348bafa4c'), u'name': u'rollo'}
{u'topic': u'computing', u'post': u'c++ rules', u'_id': ObjectId('5bb16e6e58b
71ca348bafa4d'), u'name': u'wally'}
{u'topic': u'physics', u'post': u'e=mc**2', u'_id': ObjectId('5bb16e6e58b71ca
348bafa4e'), u'name': u'marla'}
{u'attendance': u'590549', u'WorldCup': u'wc1930', u'matchesPlayed': u'18',
u'location': u'Uruguay', u'year': u'1930', u'_id': ObjectId('5bb16e9358b71ca3
48bafa4f'), u'goalsScored': u'70'}
{u'attendance': u'363000', u'WorldCup': u'wc1934', u'matchesPlayed': u'17',
u'location': u'Italy', u'year': u'1934', u'_id': ObjectId('5bb16e9358b71ca348
bafa50'), u'goalsScored': u'70'}
```

In [26]:
```
print ("display without _id -------- ")
for p in db.blog.find({}, {"_id": 0} ):
    print(p)
    print ("success - bulk insert from file")
```

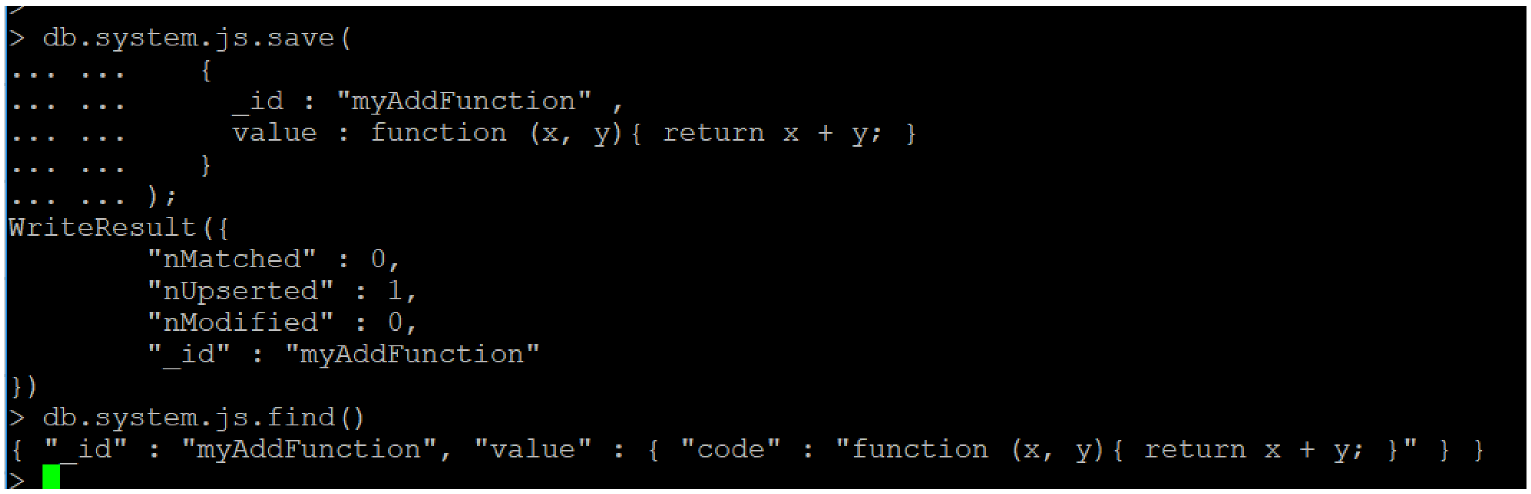
```
display without _id --------
{u'topic': u'computing', u'post': u'python rules', u'name': u'rollo'}
success - bulk insert from file
{u'topic': u'computing', u'post': u'c++ rules', u'name': u'wally'}
success - bulk insert from file
{u'topic': u'physics', u'post': u'e=mc**2', u'name': u'marla'}
success - bulk insert from file
{u'attendance': u'590549', u'WorldCup': u'wc1930', u'matchesPlayed': u'18',
u'location': u'Uruguay', u'year': u'1930', u'goalsScored': u'70'}
success - bulk insert from file
{u'attendance': u'363000', u'WorldCup': u'wc1934', u'matchesPlayed': u'17',
u'location': u'Italy', u'year': u'1934', u'goalsScored': u'70'}
success - bulk insert from file
```

In [27]:
```
#show all collections except system collections
collection = db.collection_names(include_system_collections=False)
for collect in collection:
    print (collect)
```

```
zorro
blog
```

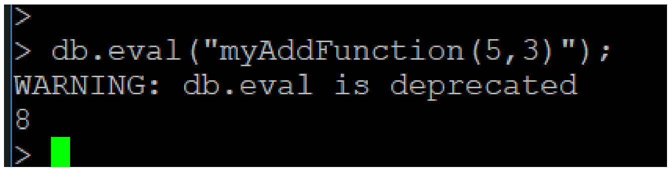
System.js

There is a special system collection named system.js that can store JavaScript functions for reuse.



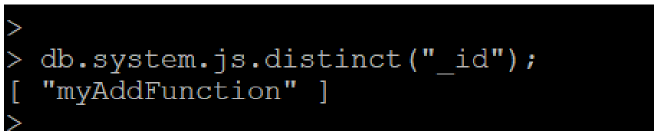
```
> db.system.js.save(
... ...     {
... ...       _id : "myAddFunction" ,
... ...       value : function (x, y){ return x + y; }
... ...     }
... ... );
WriteResult({
        "nMatched" : 0,
        "nUpserted" : 1,
        "nModified" : 0,
        "_id" : "myAddFunction"
})
> db.system.js.find()
{ "_id" : "myAddFunction", "value" : { "code" : "function (x, y){ return x + y; }" } }
>
```

Here, id field holds the name of the function and is unique per database. The value field holds the function definition.
Once you save a function in the system.js collection, you can use the function from any JavaScript context; e.g. $where operator or mapReduce command.



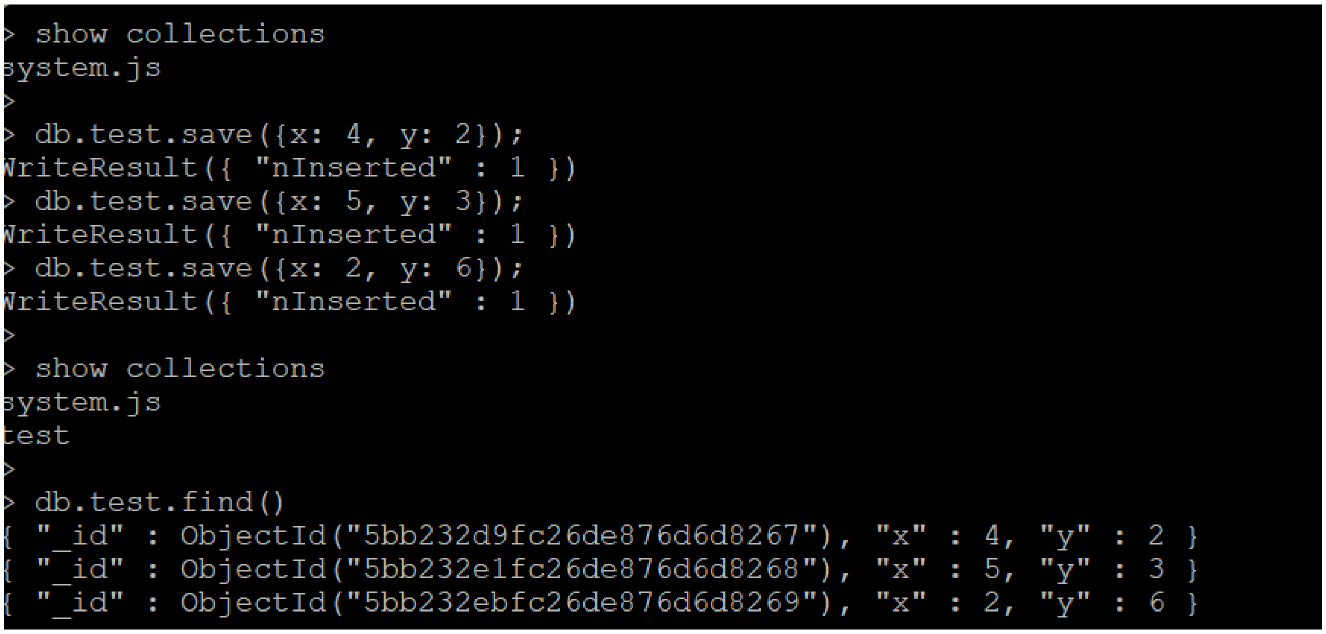
```
>
> db.eval("myAddFunction(5,3)");
WARNING: db.eval is deprecated
8
>
```

If we want to list the stored functions, we can use below command:

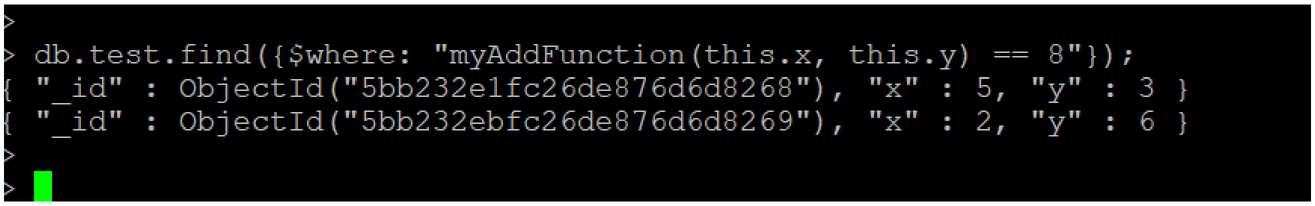


```
>
> db.system.js.distinct("_id");
[ "myAddFunction" ]
>
```

In this example, we use myAddFunction within a **$where** clause to get all documents where the addition of **x** and **y** is 8.
For that we'll create a new collection called as test passing different values of x and y.
example: > db.test.save({x:4, y:2});



```
> show collections
system.js
>
> db.test.save({x: 4, y: 2});
WriteResult({ "nInserted" : 1 })
> db.test.save({x: 5, y: 3});
WriteResult({ "nInserted" : 1 })
> db.test.save({x: 2, y: 6});
WriteResult({ "nInserted" : 1 })
>
> show collections
system.js
test
>
> db.test.find()
{ "_id" : ObjectId("5bb232d9fc26de876d6d8267"), "x" : 4, "y" : 2 }
{ "_id" : ObjectId("5bb232e1fc26de876d6d8268"), "x" : 5, "y" : 3 }
{ "_id" : ObjectId("5bb232ebfc26de876d6d8269"), "x" : 2, "y" : 6 }
```



```
>
> db.test.find({$where: "myAddFunction(this.x, this.y) == 8"});
{ "_id" : ObjectId("5bb232e1fc26de876d6d8268"), "x" : 5, "y" : 3 }
{ "_id" : ObjectId("5bb232ebfc26de876d6d8269"), "x" : 2, "y" : 6 }
>
>
```

We can also remove the function from system.js:
> db.system.js.remove({_id:"myAddFunction"});