mongoDB

RUTHIRAKUMAR

RUTHRAN.KUMAR@GMAIL.COM

91-9715531887

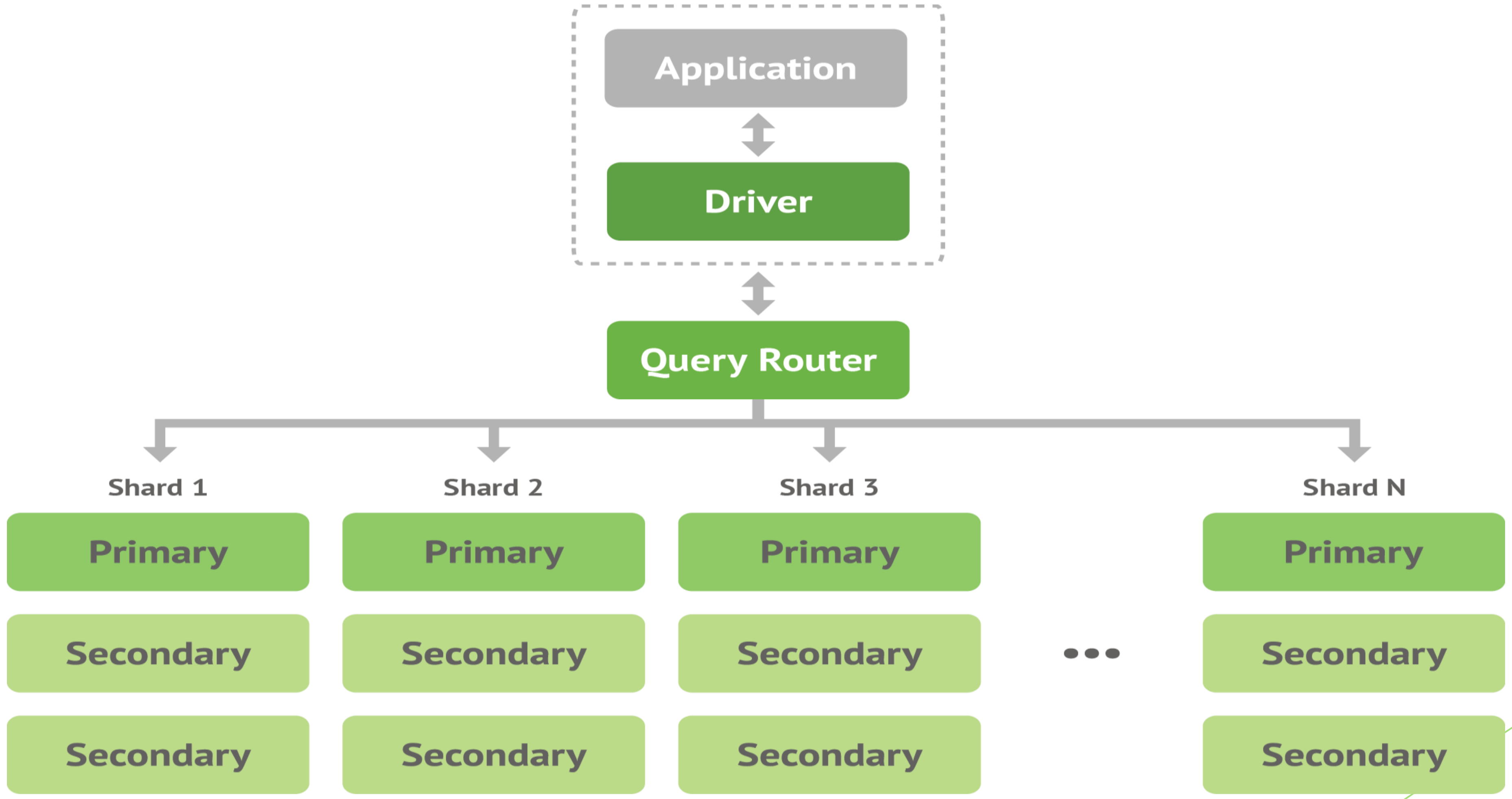
Agenda

- Overview
- Architecture
- Application set-up
- RDBMS vs MONGODB
- Mongo Shell
- Schema
- Model
- Read, Write, Update and Delete
- Sort & Limit
- Validation

Overview

- Open Source, NoSQL Database developed using C++
- Schema less – MongoDB is a document database
- Document Oriented Storage – Data is stored in the form of JSON style documents.
- Replication and high availability
- Auto-sharding
- Used in Big Data, Social Infrastructure

Architecture

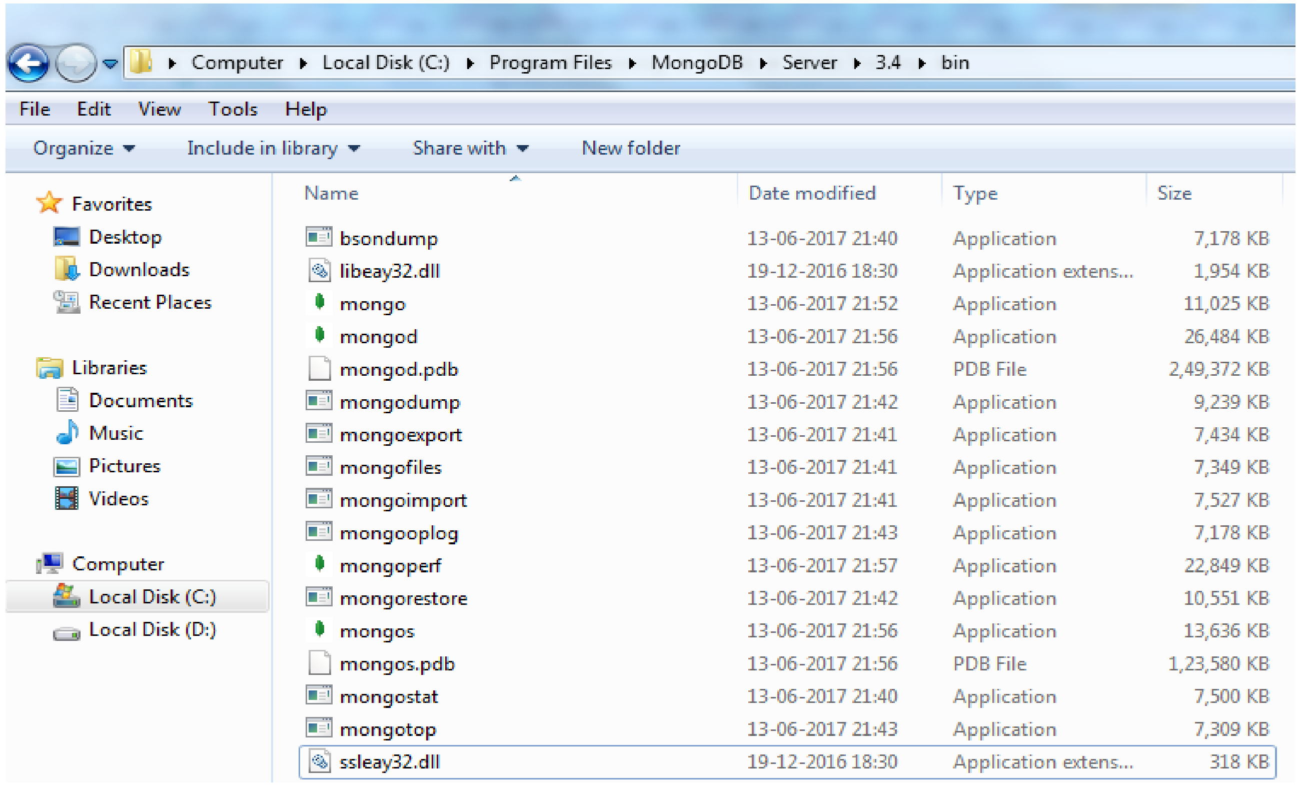


RDBMS vs MongoDB

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by mongodb itself)

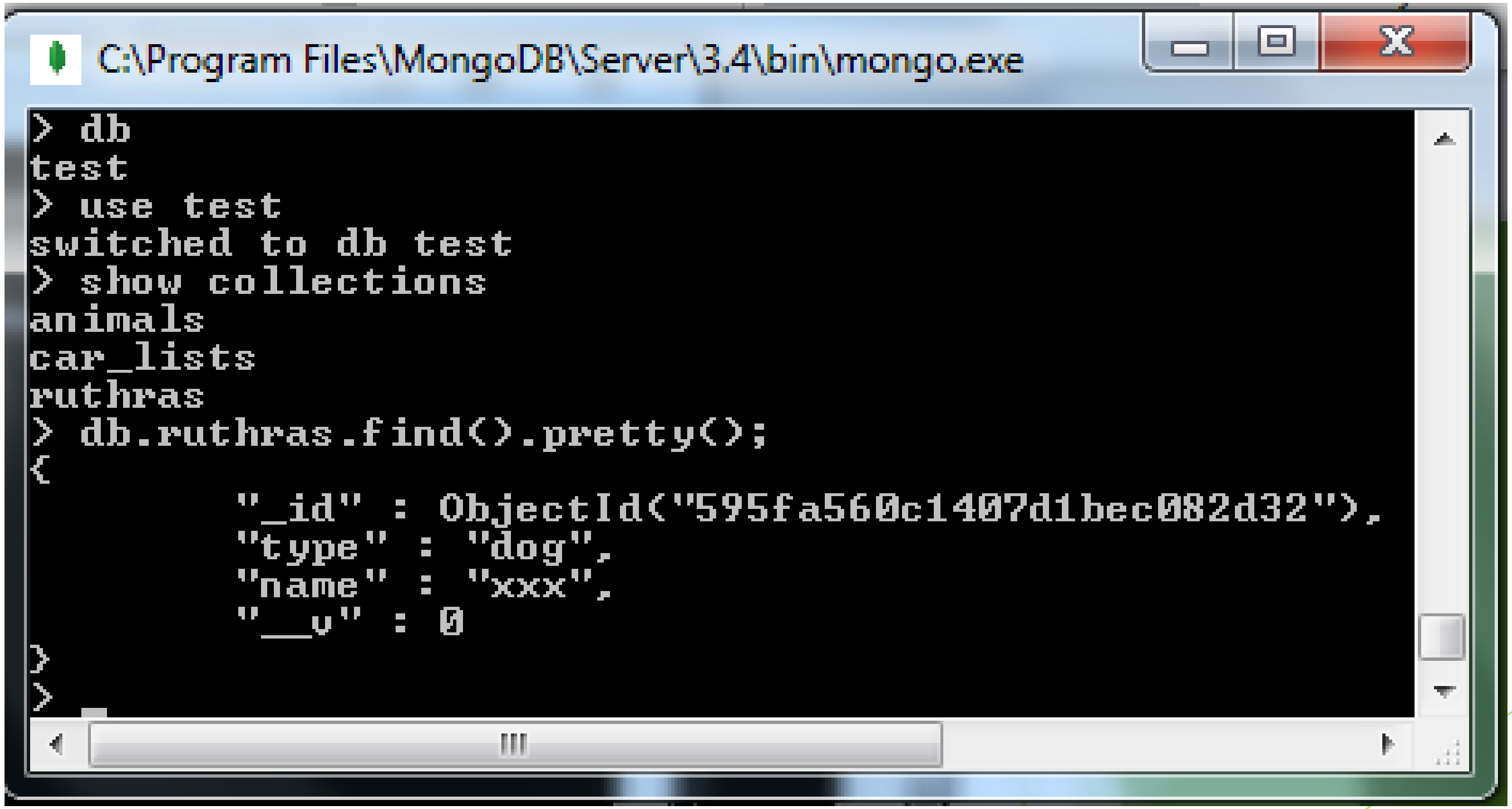
Application setup

- Download setup file from https://www.mongodb.com/ and install it.
- Navigate to installed location (C:\Program Files\MongoDB\Server\3.4\bin)
- Run Mongod and mongo


Computer ▸ Local Disk (C:) ▸ Program Files ▸ MongoDB ▸ Server ▸ 3.4 ▸ bin
File Edit View Tools Help
Organize ▾ Include in library ▾ Share with ▾ New folder
Favorites
Desktop
Downloads
Recent Places
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
Local Disk (D:)

Name	Date modified	Type	Size
bsondump	13-06-2017 21:40	Application	7,178 KB
libeay32.dll	19-12-2016 18:30	Application extens...	1,954 KB
mongo	13-06-2017 21:52	Application	11,025 KB
mongod	13-06-2017 21:56	Application	26,484 KB
mongod.pdb	13-06-2017 21:56	PDB File	2,49,372 KB
mongodump	13-06-2017 21:42	Application	9,239 KB
mongoexport	13-06-2017 21:41	Application	7,434 KB
mongofiles	13-06-2017 21:41	Application	7,349 KB
mongoimport	13-06-2017 21:41	Application	7,527 KB
mongooplog	13-06-2017 21:43	Application	7,178 KB
mongoperf	13-06-2017 21:57	Application	22,849 KB
mongorestore	13-06-2017 21:42	Application	10,551 KB
mongos	13-06-2017 21:56	Application	13,636 KB
mongos.pdb	13-06-2017 21:56	PDB File	1,23,580 KB
mongostat	13-06-2017 21:40	Application	7,500 KB
mongotop	13-06-2017 21:43	Application	7,309 KB
ssleay32.dll	19-12-2016 18:30	Application extens...	318 KB

Mongo Shell



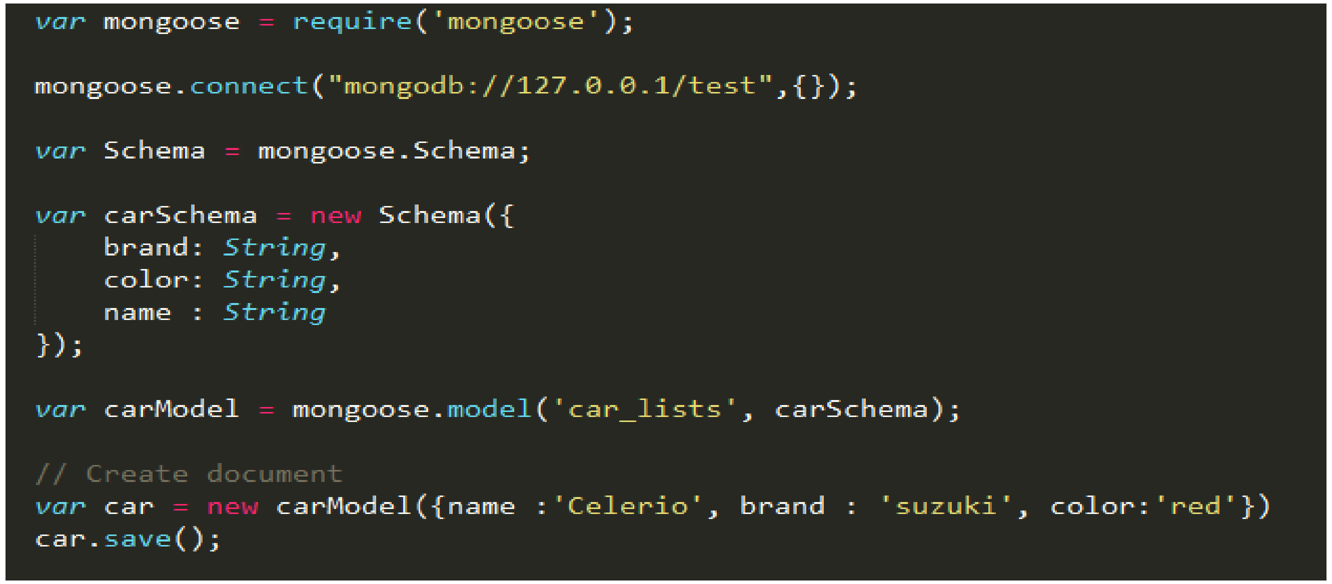
```
> db
test
> use test
switched to db test
> show collections
animals
car_lists
ruthras
> db.ruthras.find().pretty();
{
        "_id" : ObjectId("595fa560c1407d1bec082d32"),
        "type" : "dog",
        "name" : "xxx",
        "__v" : 0
}
> _
```

Mongoose

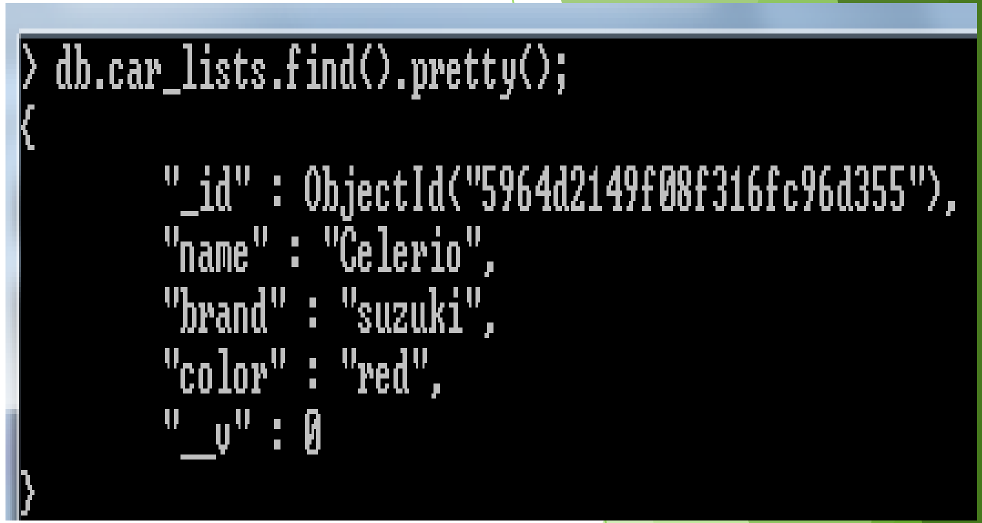
```
var mongoose = require('mongoose');
mongoose.connect('mongodb://localhost/test');

var Cat = mongoose.model('Cat', { name: String });

var kitty = new Cat({ name: 'Zildjian' });
kitty.save(function (err) {
  if (err) {
    console.log(err);
  } else {
    console.log('meow');
  }
});
```

Mongoose provides a straight-forward, schema-based solution to model your application data. It includes built-in type casting, validation, query building, business logic hooks and more, out of the box.

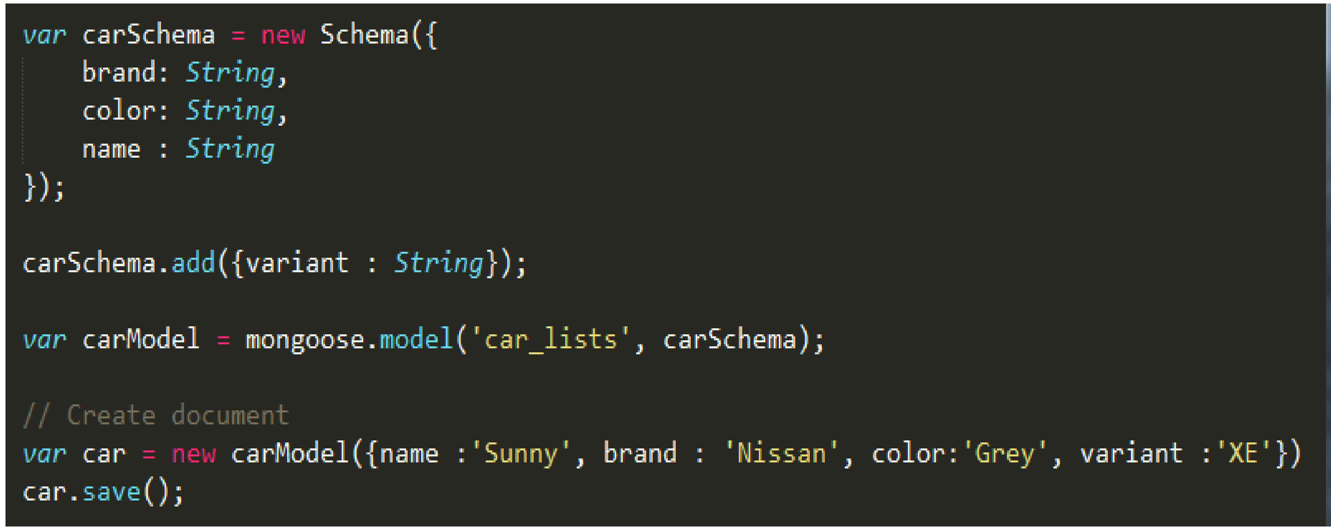
Schema (Insert)



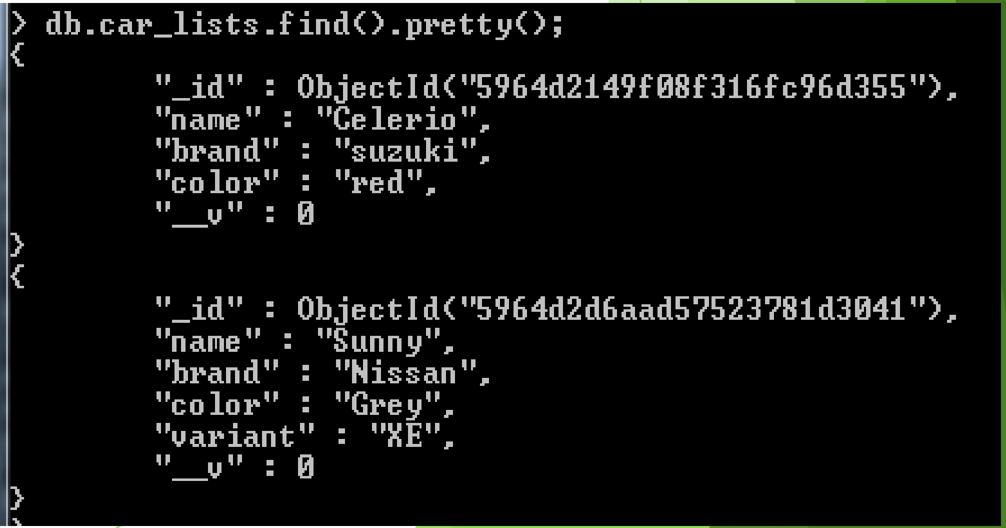
```
var mongoose = require('mongoose');

mongoose.connect("mongodb://127.0.0.1/test",{});

var Schema = mongoose.Schema;

var carSchema = new Schema({
    brand: String,
    color: String,
    name : String
});

var carModel = mongoose.model('car_lists', carSchema);

// Create document
var car = new carModel({name :'Celerio', brand : 'suzuki', color:'red'})
car.save();
```



```
> db.car_lists.find().pretty();
{
        "_id" : ObjectId("5964d2149f08f316fc96d355"),
        "name" : "Celerio",
        "brand" : "suzuki",
        "color" : "red",
        "__v" : 0
}
```

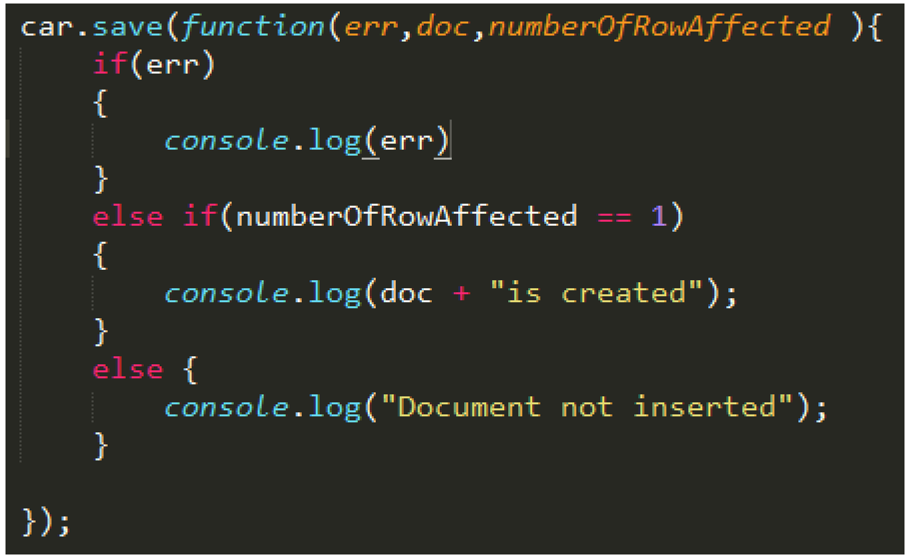


```
var carSchema = new Schema({
    brand: String,
    color: String,
    name : String
});

carSchema.add({variant : String});

var carModel = mongoose.model('car_lists', carSchema);

// Create document
var car = new carModel({name :'Sunny', brand : 'Nissan', color:'Grey', variant :'XE'})
car.save();
```

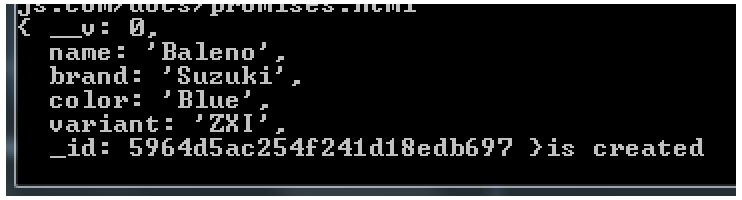


```
> db.car_lists.find().pretty();
{
        "_id" : ObjectId("5964d2149f08f316fc96d355"),
        "name" : "Celerio",
        "brand" : "suzuki",
        "color" : "red",
        "__v" : 0
}
{
        "_id" : ObjectId("5964d2d6aad57523781d3041"),
        "name" : "Sunny",
        "brand" : "Nissan",
        "color" : "Grey",
        "variant" : "XE",
        "__v" : 0
}
```

Callback for Save

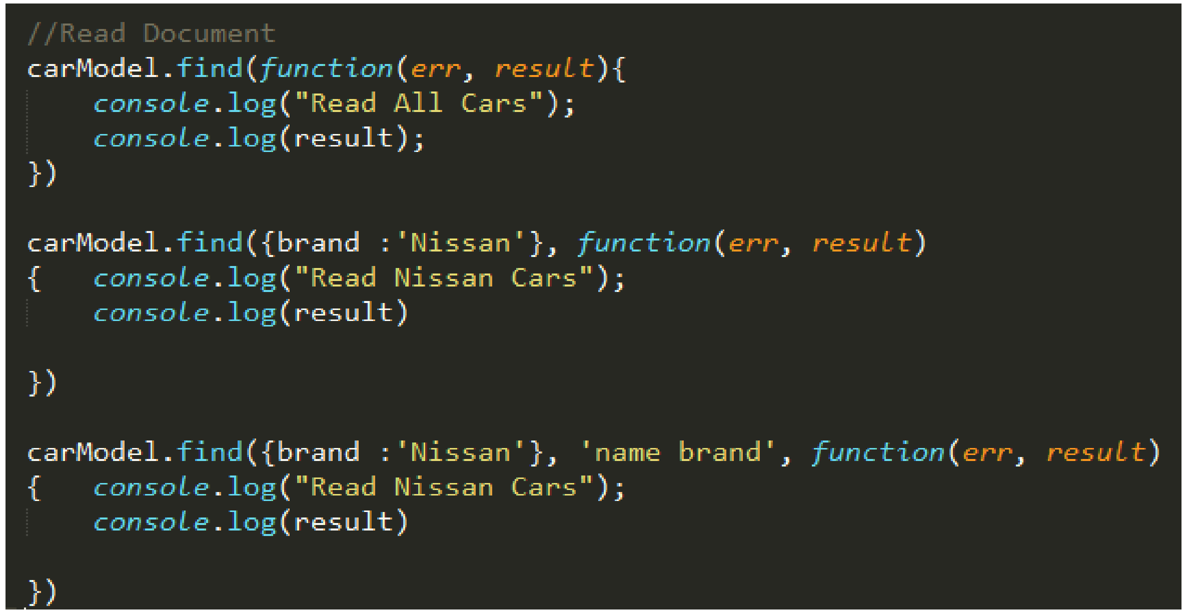


```
car.save(function(err,doc,numberOfRowAffected ){
    if(err)
    {
        console.log(err)
    }
    else if(numberOfRowAffected == 1)
    {
        console.log(doc + "is created");
    }
    else {
        console.log("Document not inserted");
    }

});
```

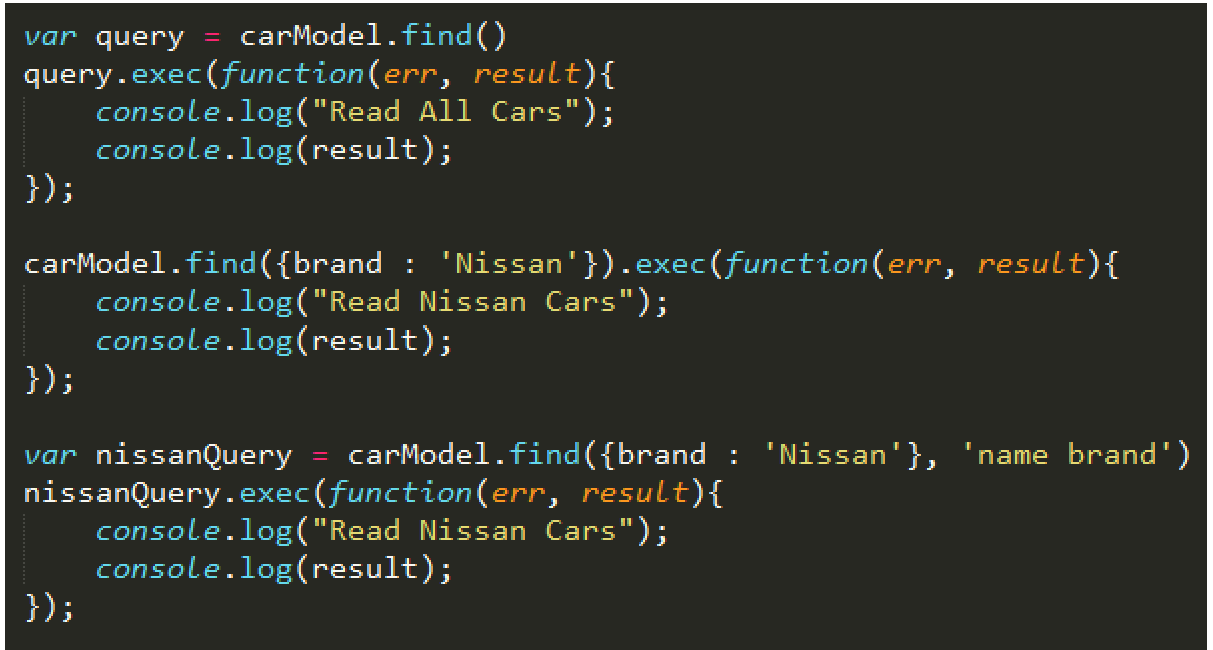


```
js.com/docs/promises.html
{ __v: 0,
  name: 'Baleno',
  brand: 'Suzuki',
  color: 'Blue',
  variant: 'ZXI',
  _id: 5964d5ac254f241d18edb697 }is created
```

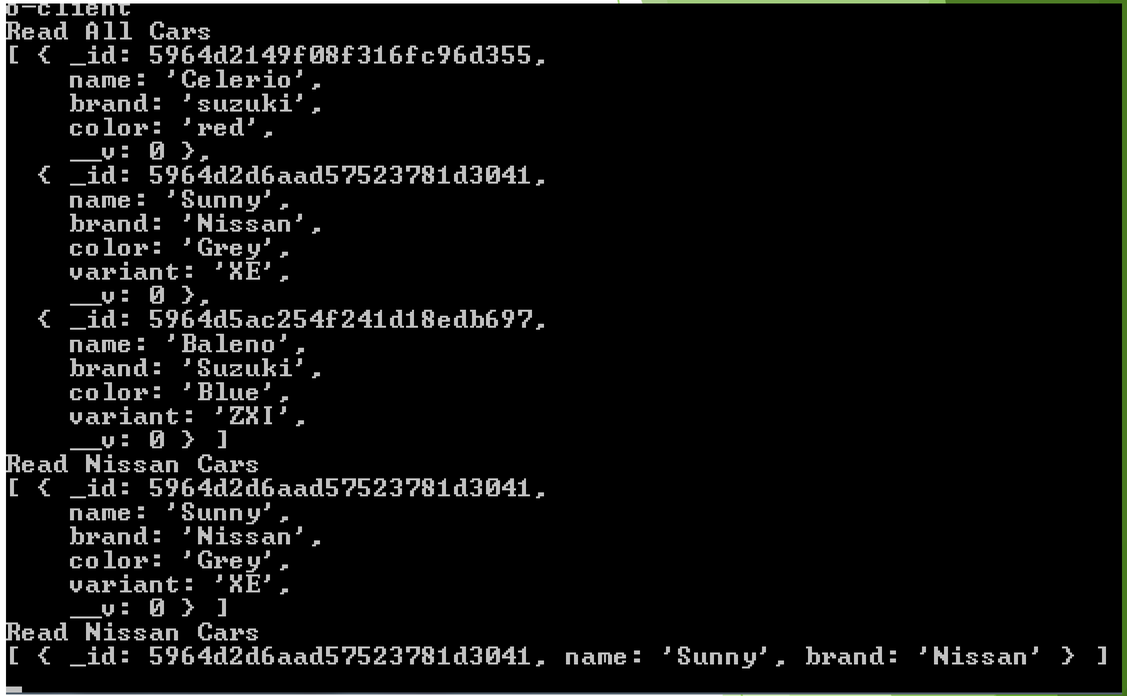
Read



```
//Read Document
carModel.find(function(err, result){
    console.log("Read All Cars");
    console.log(result);
})

carModel.find({brand :'Nissan'}, function(err, result)
{   console.log("Read Nissan Cars");
    console.log(result)

})

carModel.find({brand :'Nissan'}, 'name brand', function(err, result)
{   console.log("Read Nissan Cars");
    console.log(result)

})
```

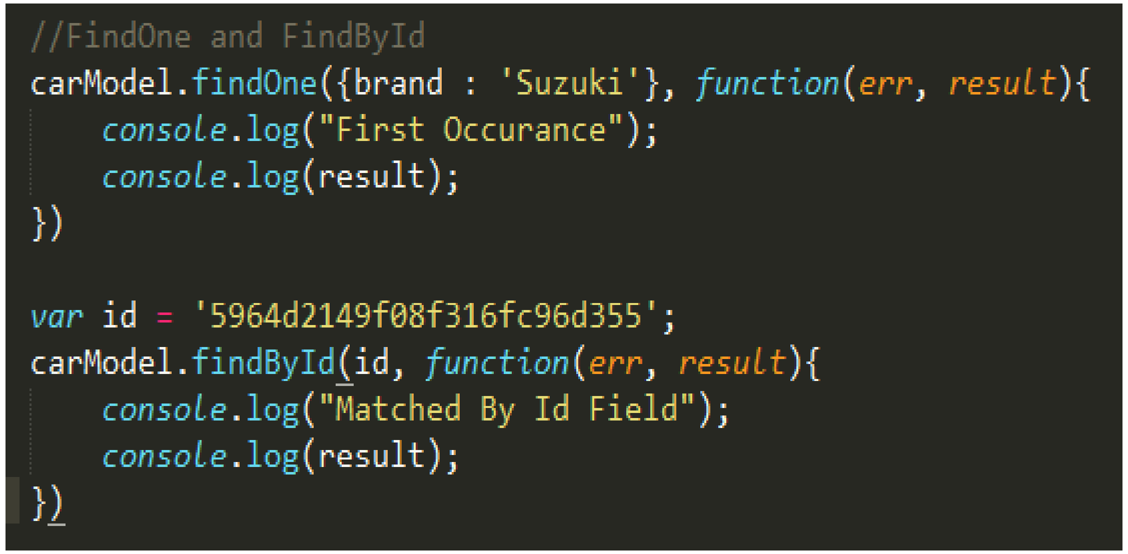


```
var query = carModel.find()
query.exec(function(err, result){
    console.log("Read All Cars");
    console.log(result);
});

carModel.find({brand : 'Nissan'}).exec(function(err, result){
    console.log("Read Nissan Cars");
    console.log(result);
});

var nissanQuery = carModel.find({brand : 'Nissan'}, 'name brand')
nissanQuery.exec(function(err, result){
    console.log("Read Nissan Cars");
    console.log(result);
});
```

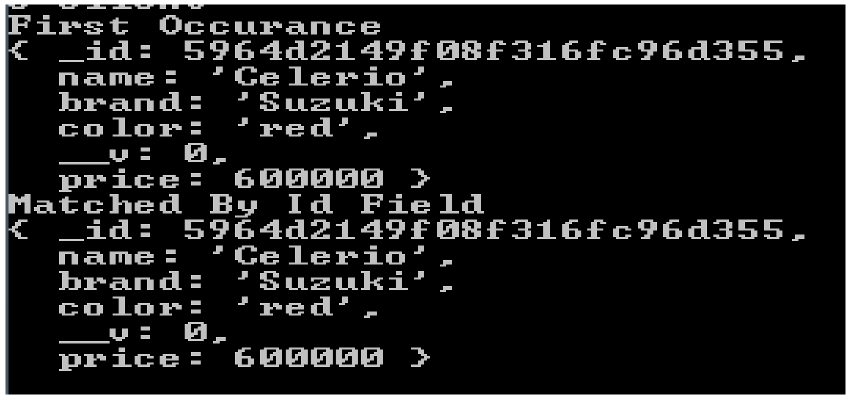


```
b-client
Read All Cars
[ { _id: 5964d2149f08f316fc96d355,
    name: 'Celerio',
    brand: 'suzuki',
    color: 'red',
    __v: 0 },
  { _id: 5964d2d6aad57523781d3041,
    name: 'Sunny',
    brand: 'Nissan',
    color: 'Grey',
    variant: 'XE',
    __v: 0 },
  { _id: 5964d5ac254f241d18edb697,
    name: 'Baleno',
    brand: 'Suzuki',
    color: 'Blue',
    variant: 'ZXI',
    __v: 0 } ]
Read Nissan Cars
[ { _id: 5964d2d6aad57523781d3041,
    name: 'Sunny',
    brand: 'Nissan',
    color: 'Grey',
    variant: 'XE',
    __v: 0 } ]
Read Nissan Cars
[ { _id: 5964d2d6aad57523781d3041, name: 'Sunny', brand: 'Nissan' } ]
```

Read

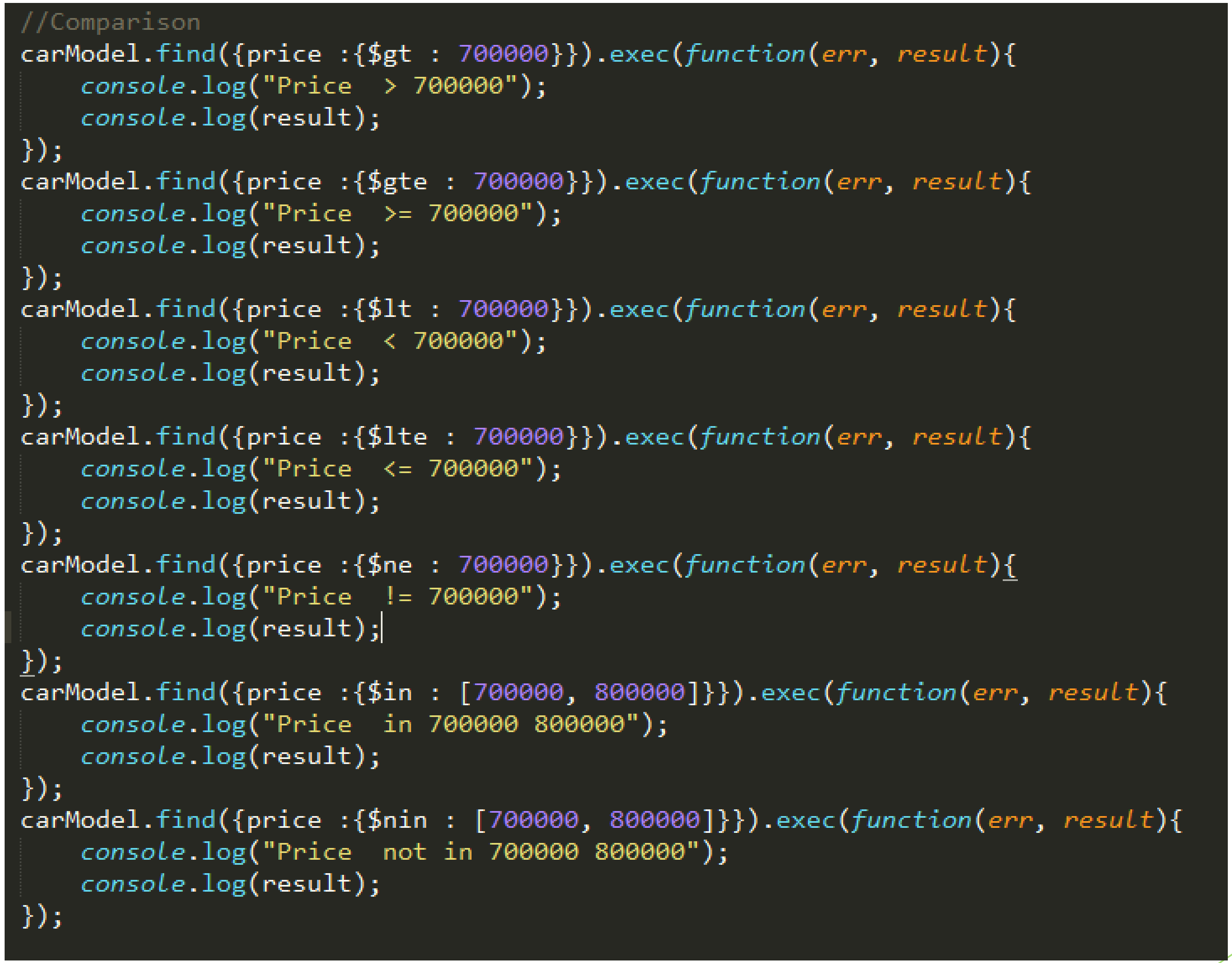


```javascript
//FindOne and FindById
carModel.findOne({brand : 'Suzuki'}, function(err, result){
    console.log("First Occurance");
    console.log(result);
})

var id = '5964d2149f08f316fc96d355';
carModel.findById(id, function(err, result){
    console.log("Matched By Id Field");
    console.log(result);
})
```



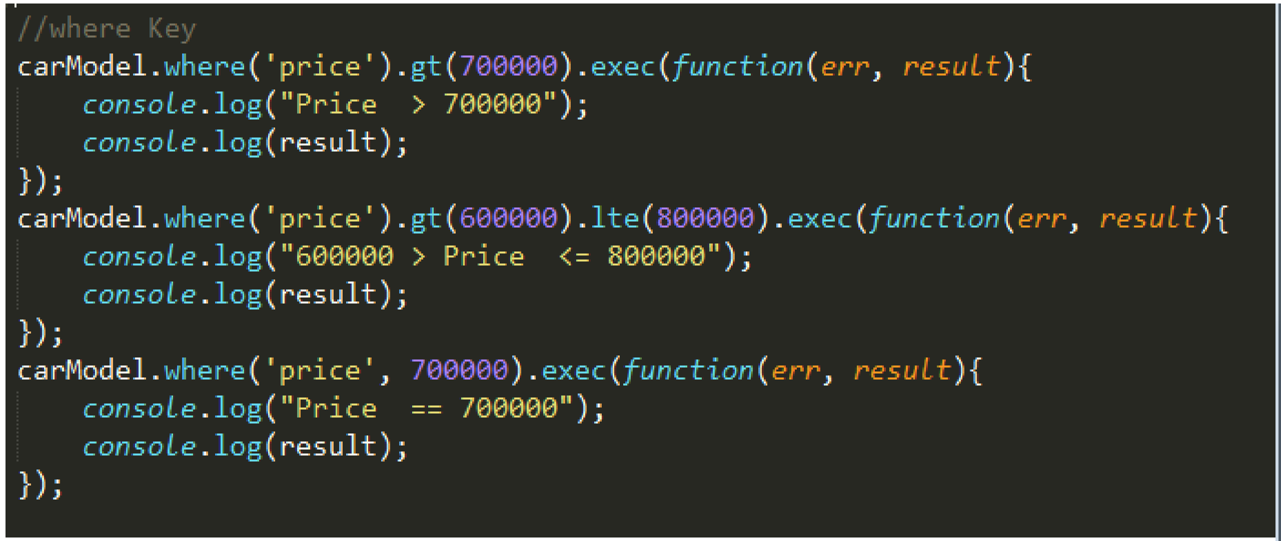
```
First Occurance
{ _id: 5964d2149f08f316fc96d355,
  name: 'Celerio',
  brand: 'Suzuki',
  color: 'red',
  __v: 0,
  price: 600000 }
Matched By Id Field
{ _id: 5964d2149f08f316fc96d355,
  name: 'Celerio',
  brand: 'Suzuki',
  color: 'red',
  __v: 0,
  price: 600000 }
```

Comparison Query Operators

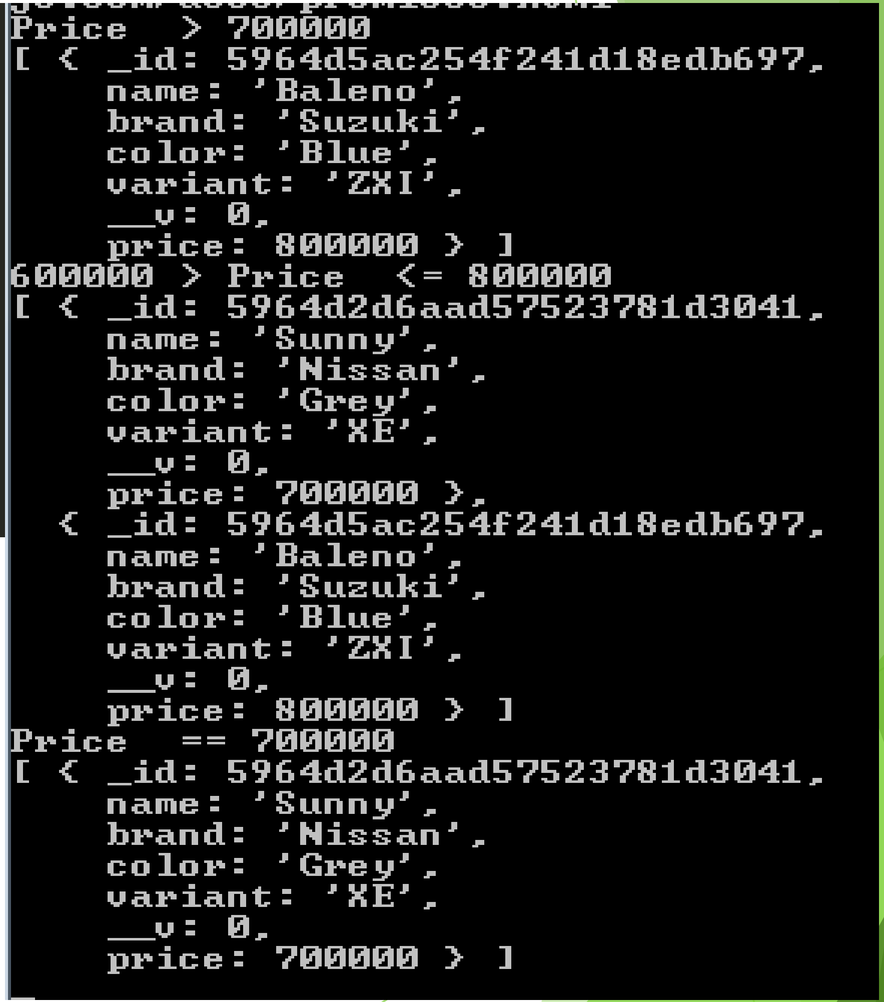


```
//Comparison
carModel.find({price :{$gt : 700000}}).exec(function(err, result){
    console.log("Price  > 700000");
    console.log(result);
});
carModel.find({price :{$gte : 700000}}).exec(function(err, result){
    console.log("Price  >= 700000");
    console.log(result);
});
carModel.find({price :{$lt : 700000}}).exec(function(err, result){
    console.log("Price  < 700000");
    console.log(result);
});
carModel.find({price :{$lte : 700000}}).exec(function(err, result){
    console.log("Price  <= 700000");
    console.log(result);
});
carModel.find({price :{$ne : 700000}}).exec(function(err, result){
    console.log("Price  != 700000");
    console.log(result);
});
carModel.find({price :{$in : [700000, 800000]}}).exec(function(err, result){
    console.log("Price  in 700000 800000");
    console.log(result);
});
carModel.find({price :{$nin : [700000, 800000]}}).exec(function(err, result){
    console.log("Price  not in 700000 800000");
    console.log(result);
});
```

Where Condition

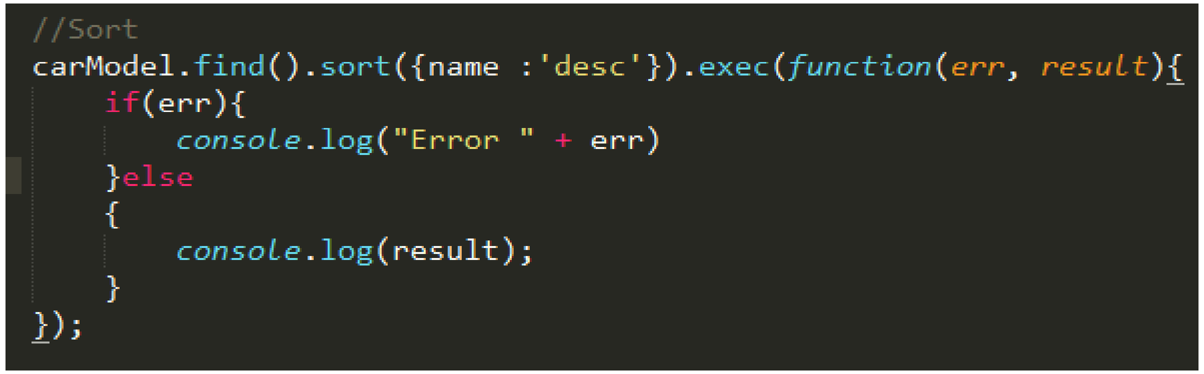


```
//where Key
carModel.where('price').gt(700000).exec(function(err, result){
    console.log("Price  > 700000");
    console.log(result);
});
carModel.where('price').gt(600000).lte(800000).exec(function(err, result){
    console.log("600000 > Price  <= 800000");
    console.log(result);
});
carModel.where('price', 700000).exec(function(err, result){
    console.log("Price  == 700000");
    console.log(result);
});
```

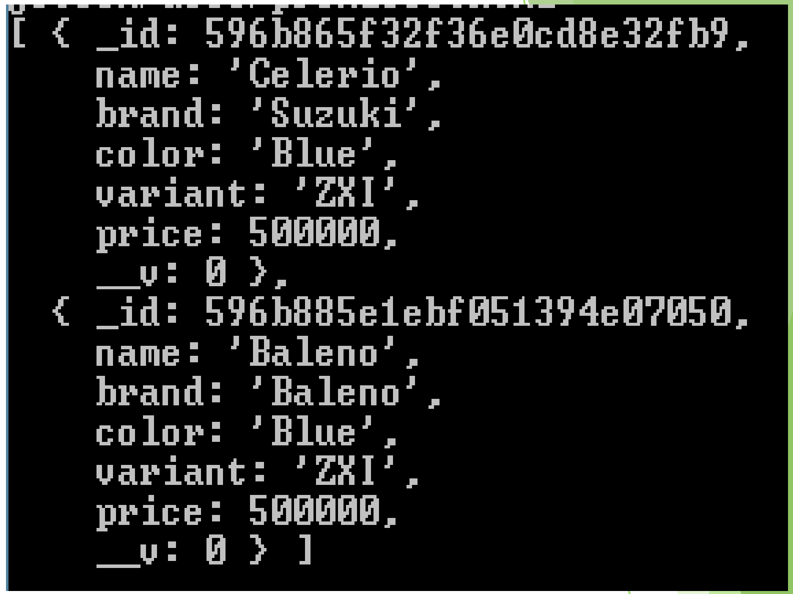


```
Price  > 700000
[ { _id: 5964d5ac254f241d18edb697,
    name: 'Baleno',
    brand: 'Suzuki',
    color: 'Blue',
    variant: 'ZXI',
    __v: 0,
    price: 800000 } ]
600000 > Price  <= 800000
[ { _id: 5964d2d6aad57523781d3041,
    name: 'Sunny',
    brand: 'Nissan',
    color: 'Grey',
    variant: 'XE',
    __v: 0,
    price: 700000 },
  { _id: 5964d5ac254f241d18edb697,
    name: 'Baleno',
    brand: 'Suzuki',
    color: 'Blue',
    variant: 'ZXI',
    __v: 0,
    price: 800000 } ]
Price  == 700000
[ { _id: 5964d2d6aad57523781d3041,
    name: 'Sunny',
    brand: 'Nissan',
    color: 'Grey',
    variant: 'XE',
    __v: 0,
    price: 700000 } ]
```

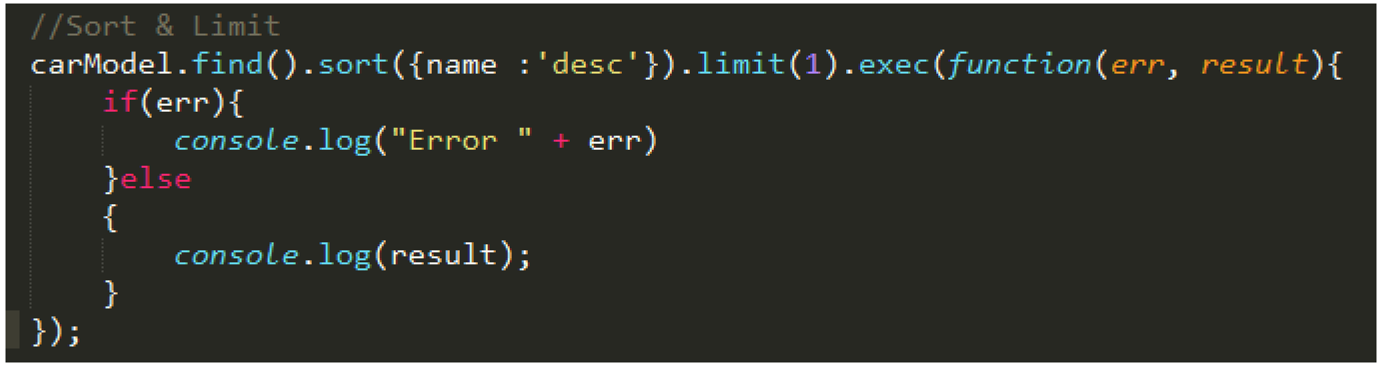
Sort & Limit



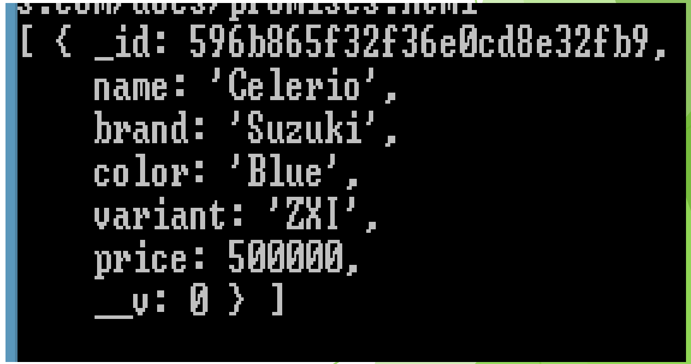
```
//Sort
carModel.find().sort({name :'desc'}).exec(function(err, result){
    if(err){
        console.log("Error " + err)
    }else
    {
        console.log(result);
    }
});
```



```
[ { _id: 596b865f32f36e0cd8e32fb9,
    name: 'Celerio',
    brand: 'Suzuki',
    color: 'Blue',
    variant: 'ZXI',
    price: 500000,
    __v: 0 },
  { _id: 596b885e1ebf051394e07050,
    name: 'Baleno',
    brand: 'Baleno',
    color: 'Blue',
    variant: 'ZXI',
    price: 500000,
    __v: 0 } ]
```

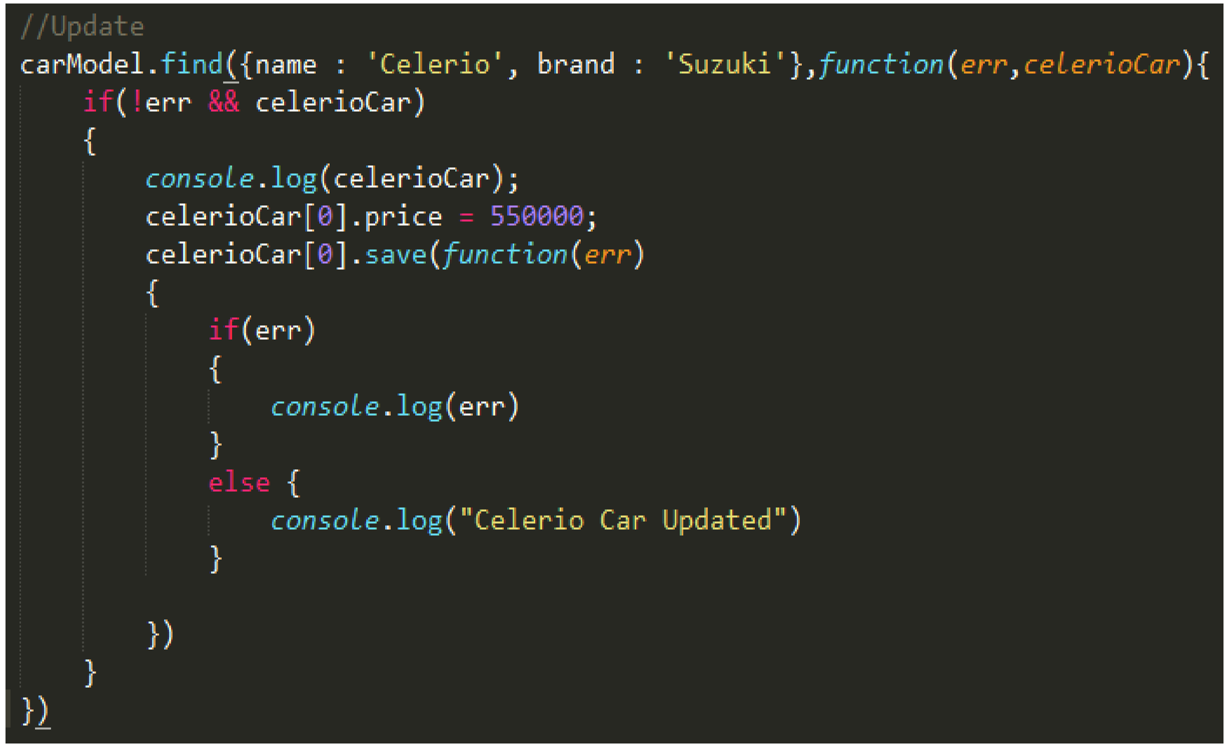


```
//Sort & Limit
carModel.find().sort({name :'desc'}).limit(1).exec(function(err, result){
    if(err){
        console.log("Error " + err)
    }else
    {
        console.log(result);
    }
});
```

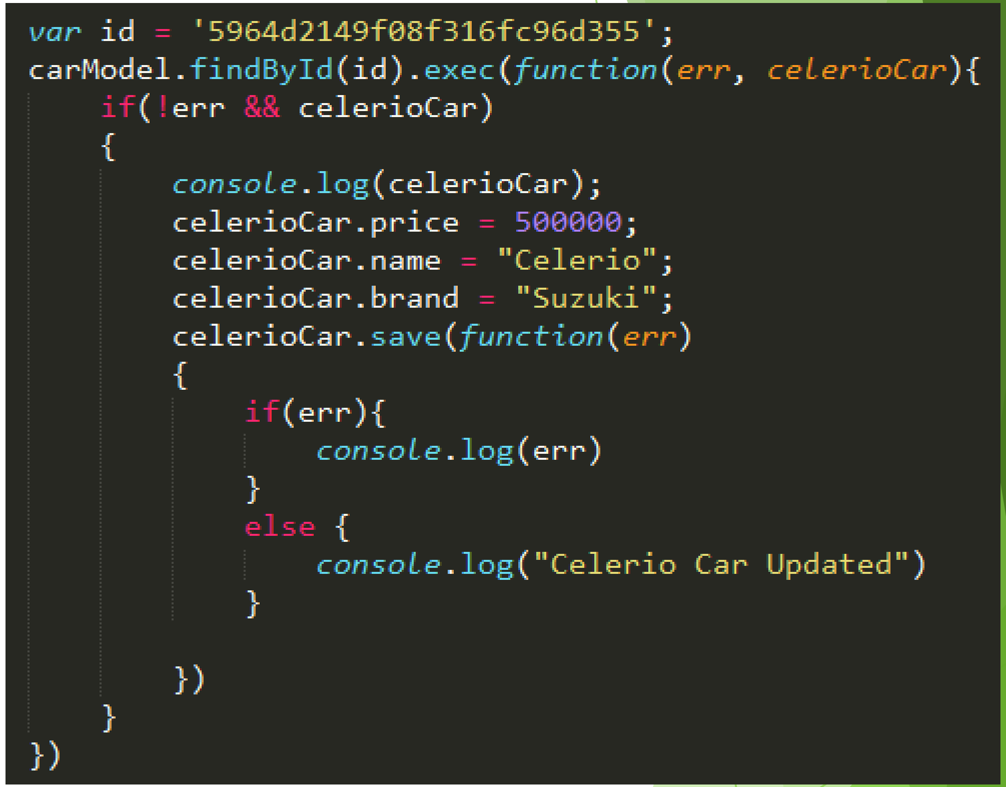


```
[ { _id: 596b865f32f36e0cd8e32fb9,
    name: 'Celerio',
    brand: 'Suzuki',
    color: 'Blue',
    variant: 'ZXI',
    price: 500000,
    __v: 0 } ]
```

Update

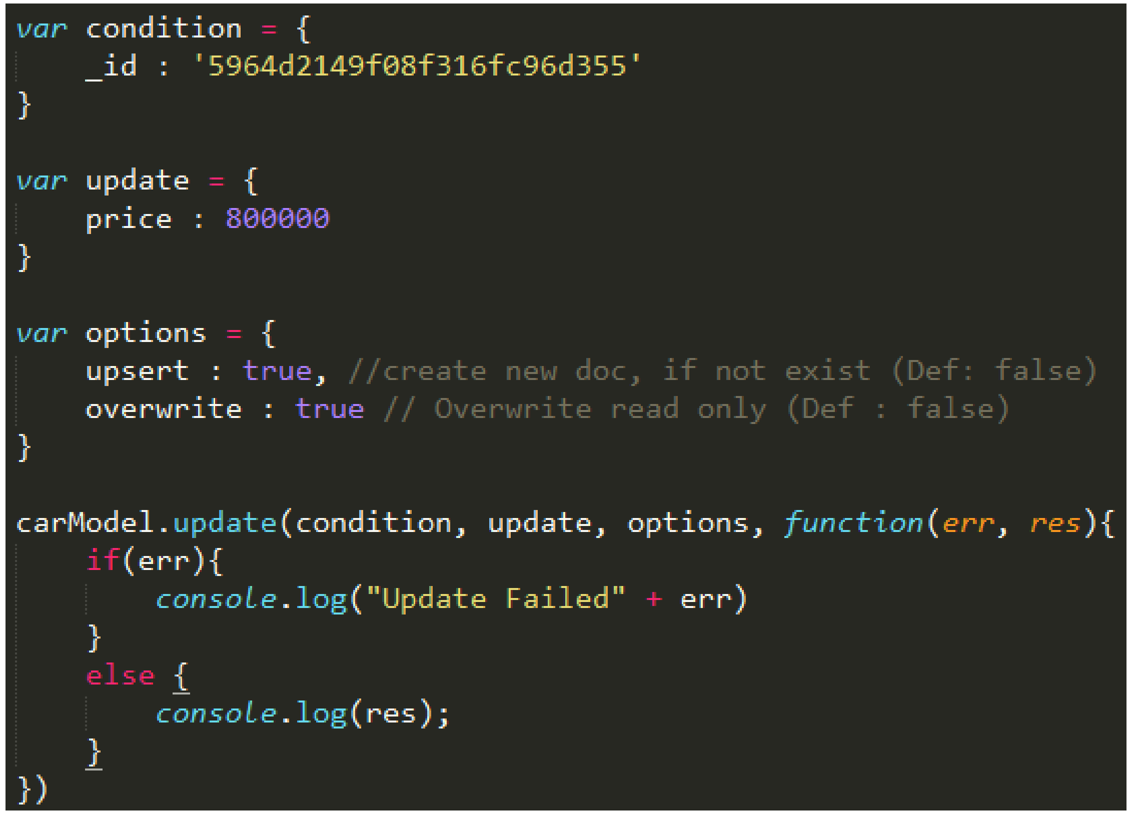


```
//Update
carModel.find({name : 'Celerio', brand : 'Suzuki'},function(err,celerioCar){
    if(!err && celerioCar)
    {
        console.log(celerioCar);
        celerioCar[0].price = 550000;
        celerioCar[0].save(function(err)
        {
            if(err)
            {
                console.log(err)
            }
            else {
                console.log("Celerio Car Updated")
            }

        })
    }
})
```

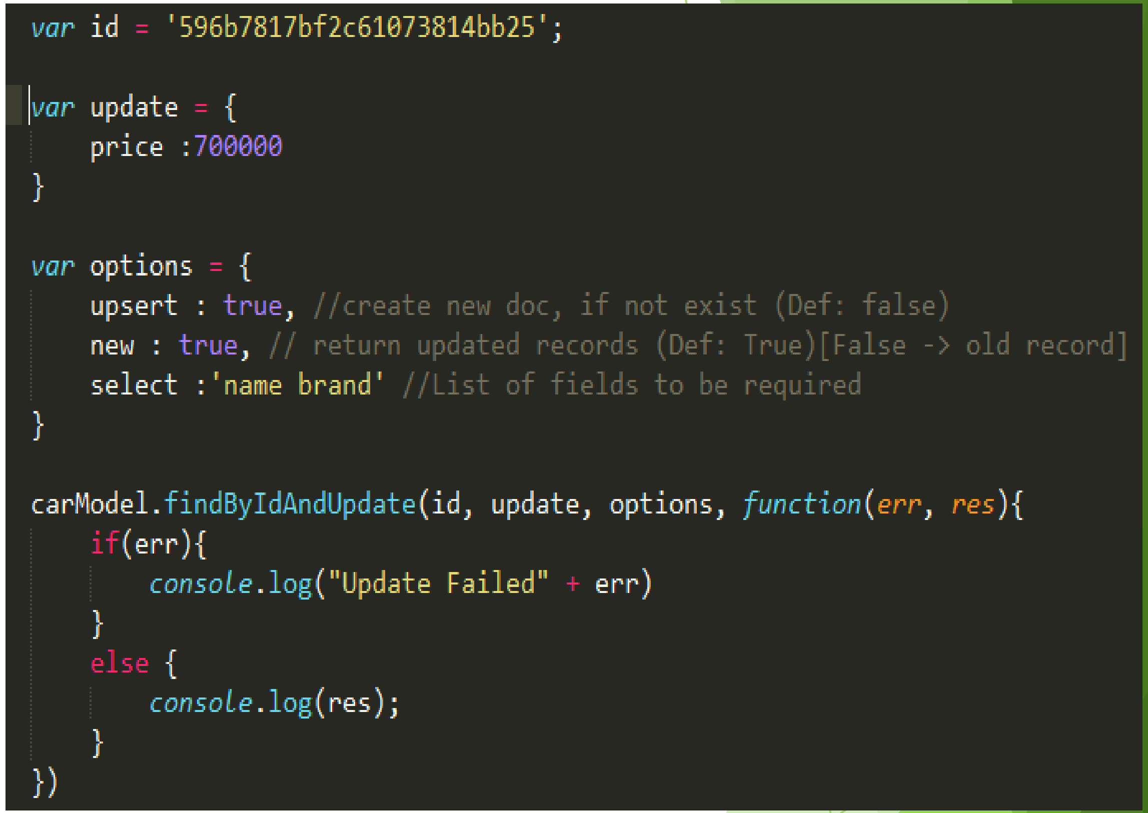


```
var id = '5964d2149f08f316fc96d355';
carModel.findById(id).exec(function(err, celerioCar){
    if(!err && celerioCar)
    {
        console.log(celerioCar);
        celerioCar.price = 500000;
        celerioCar.name = "Celerio";
        celerioCar.brand = "Suzuki";
        celerioCar.save(function(err)
        {
            if(err){
                console.log(err)
            }
            else {
                console.log("Celerio Car Updated")
            }

        })
    }
})
```

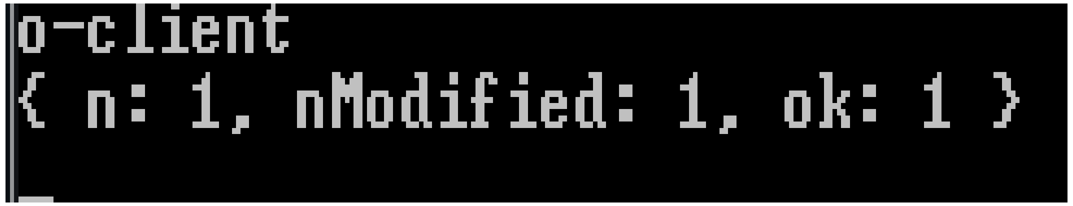
Update



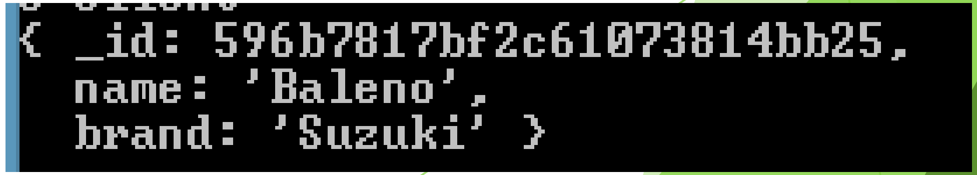
```
var condition = {
    _id : '5964d2149f08f316fc96d355'
}

var update = {
    price : 800000
}

var options = {
    upsert : true, //create new doc, if not exist (Def: false)
    overwrite : true // Overwrite read only (Def : false)
}

carModel.update(condition, update, options, function(err, res){
    if(err){
        console.log("Update Failed" + err)
    }
    else {
        console.log(res);
    }
})
```



```
var id = '596b7817bf2c61073814bb25';

var update = {
    price :700000
}

var options = {
    upsert : true, //create new doc, if not exist (Def: false)
    new : true, // return updated records (Def: True)[False -> old record]
    select :'name brand' //List of fields to be required
}

carModel.findByIdAndUpdate(id, update, options, function(err, res){
    if(err){
        console.log("Update Failed" + err)
    }
    else {
        console.log(res);
    }
})
```

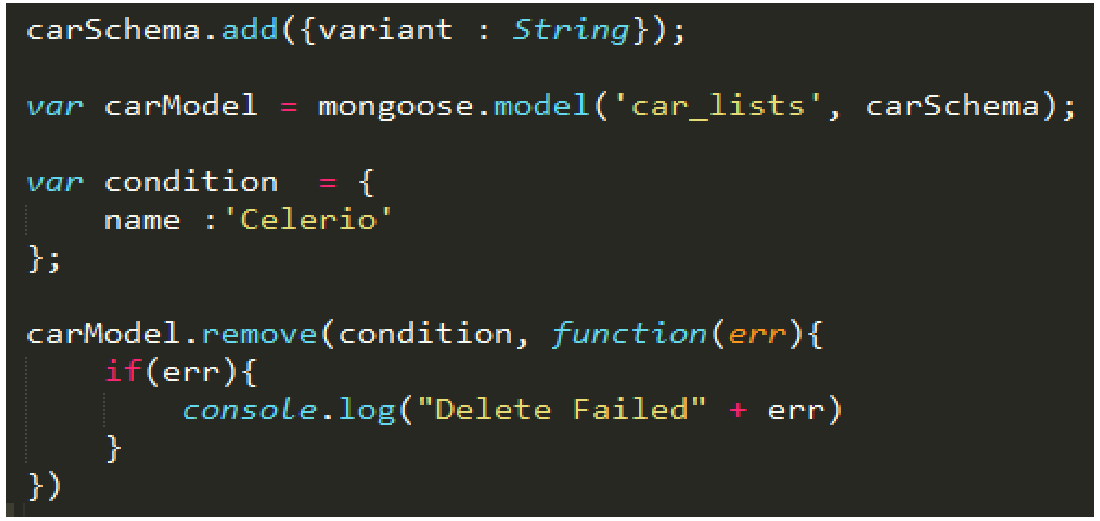


```
o-client
{ n: 1, nModified: 1, ok: 1 }
```

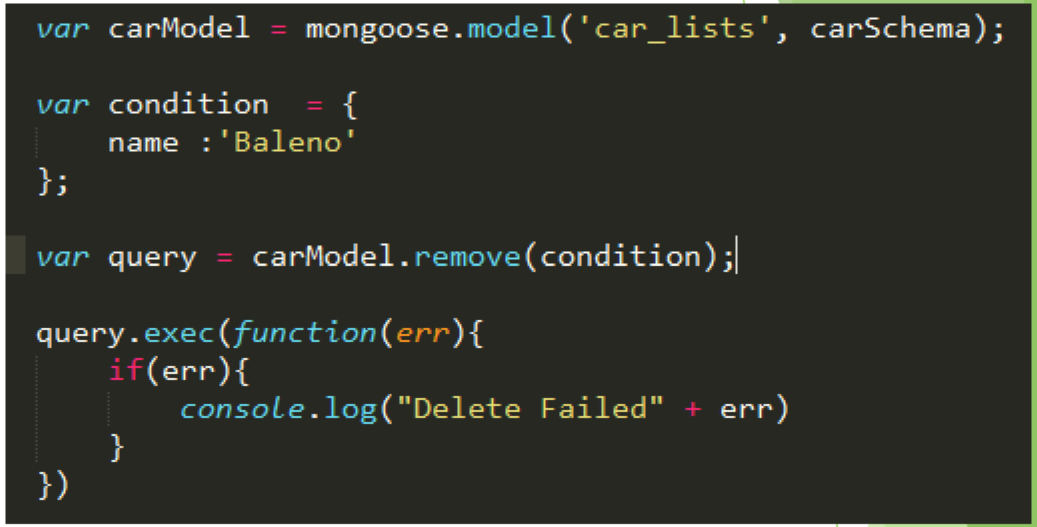


```
{ _id: 596b7817bf2c61073814bb25,
  name: 'Baleno',
  brand: 'Suzuki' }
```

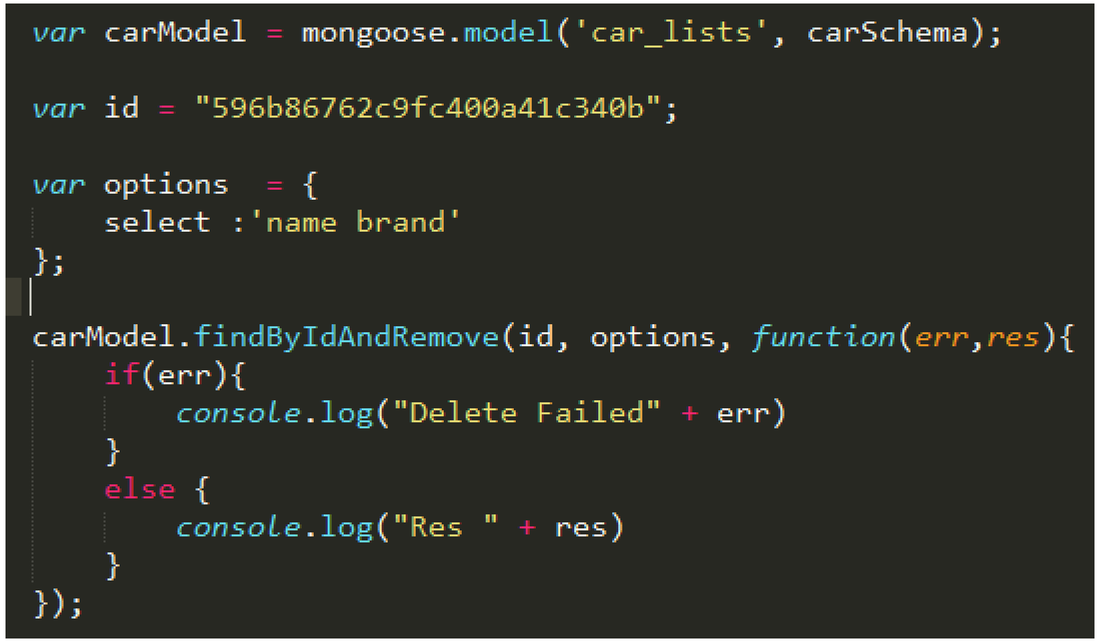
Remove



```
carSchema.add({variant : String});

var carModel = mongoose.model('car_lists', carSchema);

var condition  = {
    name :'Celerio'
};

carModel.remove(condition, function(err){
    if(err){
        console.log("Delete Failed" + err)
    }
})
```

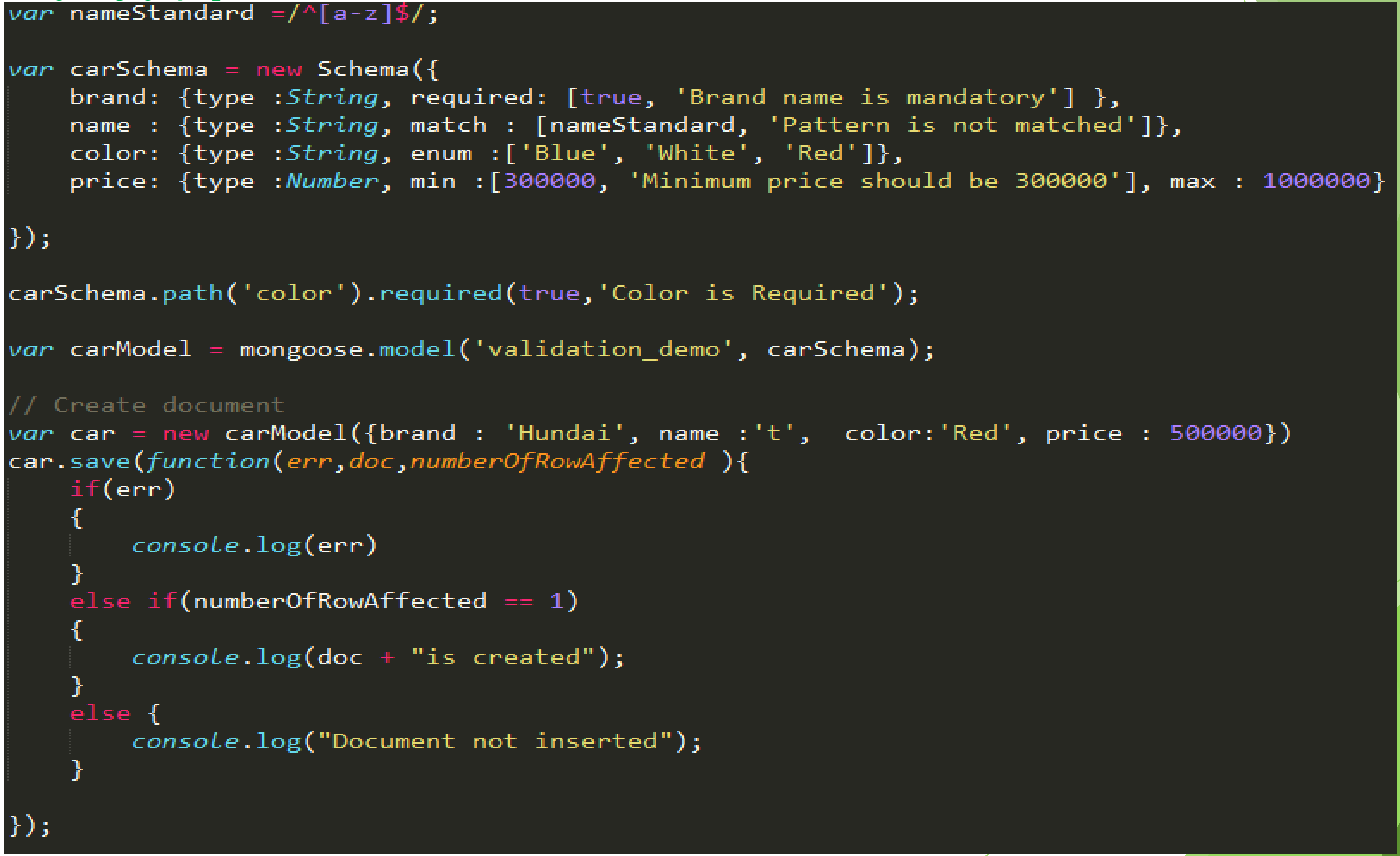


```
var carModel = mongoose.model('car_lists', carSchema);

var condition  = {
    name :'Baleno'
};

var query = carModel.remove(condition);

query.exec(function(err){
    if(err){
        console.log("Delete Failed" + err)
    }
})
```



```
var carModel = mongoose.model('car_lists', carSchema);

var id = "596b86762c9fc400a41c340b";

var options  = {
    select :'name brand'
};

carModel.findByIdAndRemove(id, options, function(err,res){
    if(err){
        console.log("Delete Failed" + err)
    }
    else {
        console.log("Res " + res)
    }
});
```

Validation



```
var nameStandard =/^[a-z]$/;

var carSchema = new Schema({
    brand: {type :String, required: [true, 'Brand name is mandatory'] },
    name : {type :String, match : [nameStandard, 'Pattern is not matched']},
    color: {type :String, enum :['Blue', 'White', 'Red']},
    price: {type :Number, min :[300000, 'Minimum price should be 300000'], max : 1000000}

});

carSchema.path('color').required(true,'Color is Required');

var carModel = mongoose.model('validation_demo', carSchema);

// Create document
var car = new carModel({brand : 'Hundai', name :'t',  color:'Red', price : 500000})
car.save(function(err,doc,numberOfRowAffected ){
    if(err)
    {
        console.log(err)
    }
    else if(numberOfRowAffected == 1)
    {
        console.log(doc + "is created");
    }
    else {
        console.log("Document not inserted");
    }

});
```

Thank You

RUTHRAN.KUMAR@GMAIL.COM
91-9715531887