Comandos do MongoDB

Insere estados

* db.estados.insert( {nome: "Acre", sigla: "AC", regiao: "Norte"} )
* db.estados.save({ nome: "Alagoas", sigla: "AL", regiao: "Nordeste", populacao: 3.2 })
* db.estados.insert({

nome: "Rio de Janeiro",

sigla: "RJ",

regiao: "Sudeste",

cidades: [{

\_id: ObjectId(),

nome: "Niteroi",

area: 133.9,

populacao: 450.0

}]

})

* db.estados.insert({

nome: "São Paulo",

sigla: "SP",

regiao: "Sudeste",

cidades: [{

\_id: ObjectId(),

nome: "Campinas",

area: 795.7,

prefeito: "Jonas Donizette",

populacao: 1081000

}, {

\_id: ObjectId(),

nome: "Garulhos",

populacao: 1325000

}, {

\_id: ObjectId(),

nome: "Sorocaba",

distanciaCapital: 87,

populacao: 644919

}]

})

Retorna todos os dados

* db.estados.find()
* db.estados.find().pretty()

Retorna somente um dado que adeque aos parâmetros da consulta

* db.estados.findOne({ sigla:”RJ” })

Retorna os dados que se adeque aos parâmetros da consulta

* db.estados.find({ regiao: “Sudeste” }).pretty()

Retorna os dados que se adeque aos parâmetros da consulta, usando o operador $or, $and, $not

* db.estados.find({$or: [{ sigla: “RJ” }, { sigla: “AC” }]}).pretty()
* db.estados.find({$and: [{ sigla: “RJ” }, { sigla: “AC” }]}).pretty()
* db.estados.find({$not: [{ sigla: “RJ” }, { sigla: “AC” }]}).pretty()

Retorna os dados que adeque aos parâmetros da consulta, usando o operador $exists e $notexists

* db.estados.find({populacao: { $exists: true }}).pretty()
* db.estados.find({populacao: { $exists: false }}).pretty()

Consulta Paginada e com Limite

* db.estados.find().skip(1).limit(2).pretty()

Consulta para retornar alguns atributos

* db.estados.find({ sigla: “SP” }, { nome: 1, sigla: 1 }).pretty()
* db.estados.find({ sigla: “SP” }, { “cidades.nome”: 1, \_id: 0 }).pretty()

Consultas agregadas

$match: Extrair dados com base em um filtro que é passado

$project: Extrair atributos desejados

$group: Agrupa os dados

$sum: Soma os dados

$avg: Média dos dados

$unwind: Desenrola o documento, onde cada elemento do array será transformado em um documento

$lookup:

$sort: Ordena os documentos com base em um critério

* db.estados.aggregate([

{ $project: { nome: 1, “cidades.nome”: 1, \_id: 0 } }

])

* db.estados.aggregate([

{ $project: { populacao: { $sum: “$cidades.populacao” }, sigla: 1 }},

{ $group: { \_id: null, populacaoTotal: { $sum: “$populacao” }}},

{ $project: { \_id: 0, populacaoTotal: 1 }}

])

* db.estados.aggregate([

{ $match: { “cidades.nome”: “Sorocaba” }},

{ $unwind: “$cidades” },

{ $match: { “cidades.nome”: “Sorocaba” }},

{ $project: { \_id: “$cidades.\_id” }}

]).pretty()

Realizando updates

* db.estados.update({ sigla: "AL" }, { $set: { populacao: 30200000 }})
* db.estados.update({ sigla: "SP" }, { $set: { populacao: 45000000 }})
* db.estados.update({ sigla: “RJ” }, { $set: { populacao: 16000000 }})
* db.estados.update({ sigla: "AL" },{ $set: { cidades: [{ nome: "Sergipe" }]}})
* db.estados.update(

{ sigla: “SP” },

{ $push: { cidades: { nome: “Santos”, populacao: 433966 }}}

)

Removendo dados

* db.estados.remove({ sigla: “AC” })
* db.estados.remove({ populacao: {$exists: false), 1)
* db.estados.remove({ populacao: { $lt: 20000000 }})

Inserindo empresas

* db.empresas.insert({

nome: "Bradesco",

estadoId: ObjectId("65a84a293a4db1fabe6424fd")

})

* db.empresas.insert({

nome: “Vale do Rio Doce”,

cidadeId: ObjectId(“65a84a293a4db1fabe6424fc”)

})

Consulta com LookUp

* db.empresas.aggregate([

{ $match: { nome: "Bradesco" }},

{ $lookup: {

from: "estados",

localField: "estadoId",

foreignField: "\_id",

as: "estado"

}},

{ $unwind: "$estado"},

]).pretty()

* db.empresas.aggregate([

{ $match: { nome: "Vale do Rio Doce" }},

{ $lookup: {

from: "estados",

localField: "cidadeId",

foreignField: "cidades.\_id",

as: "estado"

}},

{ $unwind: "$estado" },

{ $unwind: "$estado.cidades" },

{ $addFields: { mesmaCidade: { $cmp: [ "$estado.cidades.\_id", "$cidadeId" ]}}},

{ $match: { mesmaCidade: 0 }}

]).pretty()