**DDL**

**mongosh => показать список БД => использовать(неявно создать) БД =>**

**явно создать коллекцию => переименовать в коллекцию “users” => добавить документ в коллекцию (неявно создав коллекция) => список доментов коллекции => список коллекций => статистика БД => статистика коллекции => очистить коллекцию => удалить коллекцию => удалить БД**

mongosh => show dbs => use newdb => явно: db.createCollection("users1") =>

db.users1.renameCollection("users") => неявно: db.users.insertOne({"name":"Tom"}) => db.users.find() => show collections => db.stats() => db.users.stats() => db.users.deleteMany({}) => db.users.drop() => db.dropDatabase()

**DML (insert)**

**Добавили документы (name,age) в коллекцию:**

*db.users.insertOne({"name": "Bill", "age": 32})*

**Добавили документы (name,age) в коллекцию (сохранив в переменную)**

*doc=({"name": "Bill", "age": 32}) db.users.insertOne(doc)*

**Удаление документов (name:Bill; age>30):**

*db.users.deleteOne/*deleteMany*({name : "Tom"})*

*db.users.deleteOne({age: {$lt : 30}})*

**Добавили 3 документа с влож.массивом(languages:[])/ влож.объектом(адрес:{,})/ влож.списоком\_объектов(friends:[{name,age},{,}])**

*db.users.insertOne({"name": "Tom", "age": 18, "languages": ["english", "spanish"]})*

*db.users.insertOne({"name": "Bob", "age": 25, friends: [{"name": "Tim"}, {"name": "Tom"}]})*

*db.users.insertOne({"name": "Bob", "age": 28, "address": {"city": "Kiev", "street": "Sheva 1"}})*

**Добавили несколько документов (name, age, languages):**

*db.users.insertMany([{"name": "Donald", "age": 26, "languages": ["english", "french"]}, {"name": "Alice", "age": 31, "languages":["german", "english"]}])*

**Добавим документ с явным \_id как число:**

*db.users.insertOne({"\_id": 123457, "name": "Alla", "age": 27})*

**Добавим документ с явным \_id****как ObjectId** (24 символа):

*db.users.insertOne({"\_id": ObjectId("62e27b1b06adfcddf4619fc6"), "name": "Maria", "age": 22})*

**Загрузка данных из файла:** D:/trash/users\_load.txt: db.users.insertMany([

{"name": "Alice", "age": 31, "languages": ["english", "french"]},

{"name": "Lene", "age": 29, "languages": ["english", "spanish"]},

{"name": "Kate", "age": 30, "languages": ["german", "russian"]}])

load("D:/trash/users\_load.txt")

**Импортировать json-файл** [{},{}]: Mongo Compass

**DML (update/delete)**

**Полностью заменить один документ (name:Donald) другим:**

db.users.replaceOne({name: "Donald"}, {name: "Donald", age: 25})

**Изменим age в документе/ах с name="Tom" ($set):**

db.users.updateOne/updateMany({name : "Bob"}, {$set: {age : 27}}

**Увеличить все age на 1 ($inc):**

db.users.updateMany({}, {$inc: {age:1}})

**Добавить новое поле salary ($set):**

db.users.updateOne({name: "Donald"}, {$set: {salary:300}})

**Удалить поле salary ($unset):**

db.users.updateMany({name : "Donald"}, {$unset: {salary: 1}})

**Добавить значение в масcив (c/без дублей: $push/$addToSet):**

db.users.updateOne({name : "Donaldin"}, {$push: {languages: "russian"}})

db.users.updateOne({name : "Donaldin"}, {$addToSet: {languages: "russian"}})

**Удаление значения с массива (последний/ первый/ каждое попадание/ каждый из списка: $pop1/$pop-1/$pull/$pullAll):**

db.users.updateOne({name : "Tom"}, {$pop: {languages: 1}})

db.users.updateOne({name : "Tom"}, {$pop: {languages: -1}})

db.users.updateOne({name : "Tom"}, {$pull: {languages: "english"}}) db.users.updateOne({name : "Tom"}, {$pullAll: {languages: ["english", "german", "french"]}})

**DQL**

**SELECT \* => skip/limit => sort{name:1/-1} => поиск одиночного документа => WHERE Name='…' AND Age=31 => {friends.name:Tom} => WHERE NULL (отсутствует поле) == {age: {$eq:30}} => ($ne (!=), $gt (>), $lt (<), $gte (>=), $lte (<=), $in, $nin (not in)) => {$or: [{},{}]) => (*$and,* $not, $nor) => {поле: {$exists:true}} => {age: {$type:"number"} => {name: {$regex:"b"}**

db.users.find()

db.users.find().skip(3).limit(3)

db.users.find().sort({name: 1})

db.users.find().sort({name: -1})

db.users.findOne({name: "Tom"})

db.users.find({name: "Alice", age: 31})

db.users.find({"friends.name": "Tom"})

db.users.find({languages: null})

db.users.find ({age: {$gt : 30, $lt: 50}})

db.users.find ({age: {$in : [22, 32]}})

db.users.find ({$and : [{name: "Tom"}, {age: 22}]})

db.users.find ({company: {$exists:true}})

db.users.find ({age: {$type:"number"}})

db.users.find ({name: {$regex:"b"}})

**Фильтрация по элементам массива:**

**{lang: "en"} => {"lang.0": "en"} => {lang: "en", “de”} => {lang: {$all : ["de", "en"]}} => {lang: {$size:2}} => {$elemMatch: {friend\_age: {$gt: 18}}}} => {lang: {$slice : 1/-1}}**

db.users.find({languages: "english"})

db.users.find({"languages.0": "english

db.users.find({languages: ["english", "german"]})

db.users.find ({languages: {$all : ["german", " english"]}})

db.users.find ({languages: {$size:2}})

db.users.find({friends: {$elemMatch: {age: {$gt: 18}}}})

db.users.find ({name: "Tom"}, {languages: {$slice : -1}})

**ПРОЕКЦИЯ**

**Вывести только свойства {age:1}, где name=Tom**

db.users.find({name: "Bob"}, {age: true})

2сп: db.users.find({name: "Bob"}, {age: 1})

**Вывести только свойства age (без \_id: 0), где name=Tom**

db.users.find({name: "Tom"}, {age: true, \_id: false})

2сп: db.users.find({name: "Bob"}, {age: 1, \_id: 0})

**Вывести все свойства кроме age (age: 0), где name=Tom**

db.users.find({name: "Bob"}, {age: 0})

**Вывести только name всех документов**

db.users.find({}, {name: 1, \_id: 0})

**Вывести только name всех документов по возрастанию**

db.users.find({}, {name:1, \_id: 0}).sort({name: 1})

**Агрегатные функции и индексы (для ф-ций min/max(age)):**

**countDocuments() => count() => skip(2).count(true) => distinct => createIndex({"age" : 1}) => min/max({age:30}).hint({age:1}) => getIndexes() => dropIndex**

db.users.countDocuments()

db.users.find({name: "Tom"}).count()

db.users.find({name: "Tom"}).skip(2).count(true)

db.users.distinct("name")

db.users.createIndex({"age" : 1})

db.users.find().min({age:30}).hint({age:1})

db.users.find().max({age:30}).hint({age:1})

db.users.getIndexes()

db.users.dropIndex("age\_1")

**Использование JavaScript:**

**function sqrt => age:sqrt(5); => resultset => while(rs.hasnext()) => rs.forEach**

function sqrt(n) { return n\*n; }

db.users.find({age: sqrt(5)+3})

var resultset = db.users.find() // сохранили резалтсет в переменную

while(resultset.hasNext()){obj = resultset.next(); print(obj["name"]);}

2сп: resultset.forEach(function(obj){ print(obj.name); })

**Ручное и автоматическое связывание коллекций:**

**db.companies.insertOne(\_id) => db.users.insertOne(…company) => user1=db.users.findOne => db.companies.findOne({\_id: user.company})**

db.companies.insertOne({"\_id" : "microsoft", year: 1974})

db.users.insertOne({name: "Tom", age: 28, company: "microsoft"})

user = db.users.findOne() // { \_id: ObjectId('…'), name: 'Tom', age: 28, company: 'microsoft' }

db.companies.findOne({\_id: user.company}) // { \_id: 'microsoft', year: 1974 }

**comp1=comp.insertOne => user1=(name,age,comp:{ "$ref":”comp”, "$id":comp1. insertedId}}) =>users.insertOne(user1) => comp.findOne(\_id:user1.comp.$id)**

1\ google = db.companies.insertOne({name : "google", year: 1998}) // получаем результат добавления компании в переменную google

2\ sam = ({name: "Sam", age: 25, company: { "$ref" : "companies", "$id" : google.insertedId}}) // в переменную сохраним проект юзера и его ключ company свяжем с id добавленнной компании google

3\ db.users.insertOne(sam) \\ добавим юзера

4\ db.companies.findOne({\_id: sam.company.$id}) // { \_id: ObjectId(…), name: 'google', year: 1998 }