No SQL
Part 2

CRUD oporations

- Insert
- Find
- Update
- Delete
- nesting

MongoDB shell Basic first Commands review

```
show dbs
```

```
admin> use sampledb
```

```
sampledb> db.dropDatabase()
```

```
sampledb> db.sample_collection.insertOne({"key":"value"})
```

```
sampledb> db.sample_collection.insertMany({},{},{})
```

```
sampledb> db.sample_collection.drop()
```

Inserting options

{ordered :< true/false>} determines the behavior with all the documents in case of a few illegal ones

{writeConcern: { w: <value>, j: <boolean>, wtimeout: <number> }} determines the acknowledgement behavior

```
db.abc.insertMany([{"k1":"v1"},{"k2":"v2"}],{writeConcern :{w:3, j:true,wtimeout:200}, ordered:true})
```

Mongosh Commands- fetch

findOne

db.collection_name.findOne() — No filter

db.collection_name.findOne({*key:value*}) — filter

find

db.collection_name.find() — all

db.collection_name.find().limit(*num of results*) — limit

db.collection_name.find ({*key:value*}) — filter

db.collection_name.find ({},{key:true/false, key:0/1}) — Choose fields

In this example the filter is empty,
there is no filter

SQL Query	MongoDB Query
SELECT * FROM products	db.products.find({})
SELECT * FROM products WHERE price > 100	db.products.find({ "price": { "$gt": 100 } })
SELECT name, price FROM products WHERE price < 50	db.products.find({ "price": { "$lt": 50 } }, { "name": 1, "price": 1 })
SELECT name FROM products WHERE LOWER(name) LIKE ‘%a%’	db.products.find({“name”: /a/i},{“name”:1}

CRUD- read

findOne

db.collection_name.findOne() — No filter

db.collection_name.findOne({*key:value*}) — filter

find

db.collection_name.find() — all

db.collection_name.find().limit(*num of results*) — limit

db.collection_name.find ({*key:value*}) — filter

db.collection_name.find ({},{key:true/false, key:0/1}) — Choose fields

In this example the filter is empty,
there is no filter

CRUD- Update

updateOne

db.collection_name.updateOne({filter},{update})

db.collection_name.updateOne({*key:value*},{$set:{key:<new value>})

updateMany

db.collection_name.updateMany ({filter},{update})

db.collection_name.updateMany({*key:value*},{$set:{key:<new value>})

update

db.collection_name.update ({filter},{update},{options})

db.collection_name.update({*key:value*},{$set:{key:<new value>},{multy:true})

Add to update multiple documents

CRUD- DELETE

deleteOne **db.collection_name.deleteOne({filter})**

Will delete the first document that fits the condition
If no filter is specified- the first document found -will be deleted

deleteMany **db.collection_name.deleteMany ({filter})**

Will delete all documetns that fit the condition
If no filter is specified- all documents will be deleted form the collection