# MongoDB tutorial

1. **Create an online Altas cluster – cloud MongoDB to access free Sample Databases as instructed in the lecture**
2. **Connect to cluster and bring those sample databases and practise some queries**
3. **let’s set up some own data to play with.**

## Copy and paste this into a MongoShell that is connected to a MongoDB What happen when you execute the following code?

db.unicorns.insert({name: 'Horny',

dob: new Date(1992,2,13,7,47),

loves: ['carrot','papaya'],

weight: 600,

gender: 'm',

vampires: 63});

db.unicorns.insert({name: 'Aurora',

dob: new Date(1991, 0, 24, 13, 0),

loves: ['carrot', 'grape'],

weight: 450,

gender: 'f',

vampires: 43});

db.unicorns.insert({name: 'Unicrom',

dob: new Date(1973, 1, 9, 22, 10),

loves: ['energon', 'redbull'],

weight: 984,

gender: 'm',

vampires: 182});

db.unicorns.insert({name: 'Roooooodles',

dob: new Date(1979, 7, 18, 18, 44),

loves: ['apple'],

weight: 575,

gender: 'm',

vampires: 99});

db.unicorns.insert({name: 'Solnara',

dob: new Date(1985, 6, 4, 2, 1),

loves:['apple', 'carrot',

'chocolate'],

weight:550,

gender:'f',

vampires:80});

db.unicorns.insert({name:'Ayna',

dob: new Date(1998, 2, 7, 8, 30),

loves: ['strawberry', 'lemon'],

weight: 733,

gender: 'f',

vampires: 40});

db.unicorns.insert({name:'Kenny',

dob: new Date(1997, 6, 1, 10, 42),

loves: ['grape', 'lemon'],

weight: 690,

gender: 'm',

vampires: 39});

db.unicorns.insert({name: 'Raleigh',

dob: new Date(2005, 4, 3, 0, 57),

loves: ['apple', 'sugar'],

weight: 421,

gender: 'm',

vampires: 2});

db.unicorns.insert({name: 'Leia',

dob: new Date(2001, 9, 8, 14, 53),

loves: ['apple', 'watermelon'],

weight: 601,

gender: 'f',

vampires: 33});

db.unicorns.insert({name: 'Pilot',

dob: new Date(1997, 2, 1, 5, 3),

loves: ['apple', 'watermelon'],

weight: 650,

gender: 'm',

vampires: 54});

db.unicorns.insert({name: 'Nimue',

dob: new Date(1999, 11, 20, 16, 15),

loves: ['grape', 'carrot'],

weight: 540,

gender: 'f'});

db.unicorns.insert({name: 'Dunx',

dob: new Date(1976, 6, 18, 18, 18),

loves: ['grape', 'watermelon'],

weight: 704,

gender: 'm',

vampires: 165});

## Now execute the following queries one by one and note down your results. What do these queries do?

db.unicorns.find({gender: 'm',

weight: {$gt: 700}})

db.unicorns.find({

vampires: {$exists: false}})

db.unicorns.find({

loves: {$in:['apple','orange']}})

db.unicorns.find({gender: 'f',

$or: [{loves: 'apple'},

{weight: {$lt: 500}}]})

## Update your records by running following queries one by one. What is the purpose of the following $ operator ,upsert ,muti keywords?

db.unicorns.update({name: 'Pilot'},

{$inc: {vampires: -2}})

db.unicorns.update({name: 'Aurora'},

{$push: {loves: 'sugar'}})

db.hits.update({page: 'unicorns'},

{$inc: {hits: 1}}, {upsert:true});

db.hits.find();

db.unicorns.update({},

{$set: {vaccinated: true }},

{multi:true});

db.unicorns.find({vaccinated: true});

## Ordering - What is the purpose of these following queries?

db.unicorns.find().sort({weight: -1})

db.unicorns.find()

.sort({weight: -1})

.limit(2)

.skip(1)

db.unicorns.find({vampires: {$gt: 50}})

.count()

## No Joins – Create the following documents one by one. What does the code do?

db.employees.insert({\_id: ObjectId(

"4d85c7039ab0fd70a117d730"),

name: 'Leto'})

db.employees.insert({\_id: ObjectId(

"4d85c7039ab0fd70a117d731"),

name: 'Duncan',

manager: ObjectId(

"4d85c7039ab0fd70a117d730")});

db.employees.insert({\_id: ObjectId(

"4d85c7039ab0fd70a117d732"),

name: 'Moneo',

manager: ObjectId(

"4d85c7039ab0fd70a117d730")});

db.employees.find({manager: ObjectId(

"4d85c7039ab0fd70a117d730")})

## Arrays of collections – Create the following documents one by one. What does the code do?

db.employees.insert({\_id: ObjectId(

"4d85c7039ab0fd70a117d733"),

name: 'Siona',

manager: [ObjectId(

"4d85c7039ab0fd70a117d730"),

ObjectId(

"4d85c7039ab0fd70a117d732")] })

db.employees.find({manager: ObjectId(

"4d85c7039ab0fd70a117d730")})

## Aggregation Pipeline, what answer do you get when you run the following pipeline? Explain $ operator words and their purpose?

db.unicorns.aggregate([{$unwind:'$loves'},

{$group: {\_id:'$loves', total:{$sum:1},

unicorns:{$addToSet:'$name'}}},

{$sort:{total:-1}},

{$limit:1} ])