

Relations IN MONGODB

BY

AKHIL M ANIL

ONE to ONE Relation

```
Command Prompt - mongo
> use book
switched to db book
> db.books.insertOne({Name: "A book", Year: 2020})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd24875f4064862fca225e0")
}
> db.author.insertOne({Name: "Akhil", age: 22, BookWritten: "ObjectId("5fd24875f4064862fca225e0")"})
uncaught exception: SyntaxError: identifier starts immediately after numeric literal :
@(shell):1:69
> db.author.insertOne({Name: "Akhil", age: 22, BookWritten: ObjectId("5fd24875f4064862fca225e0")})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd24923f4064862fca225e1")
}
> db.books.find().pretty()
{
  "_id" : ObjectId("5fd24875f4064862fca225e0"),
  "Name" : "A book",
  "Year" : 2020
}
> db.author.find().pretty()
{
  "_id" : ObjectId("5fd24923f4064862fca225e1"),
  "Name" : "Akhil",
  "age" : 22,
  "BookWritten" : ObjectId("5fd24875f4064862fca225e0")
}
>
```

ONE to MANY Relation

```
Command Prompt - mongo
> use cityData
switched to db cityData
> db.cities.insertOne({name: "Kottayam", coordinates: {lat: 21, lng: 55}})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd1c41718414dfd9c0b3483")
}
> db.cities.findOne()
{
  "_id" : ObjectId("5fd1c41718414dfd9c0b3483"),
  "name" : "Kottayam",
  "coordinates" : {
    "lat" : 21,
    "lng" : 55
  }
}
> db.citizens.insertMany([{name: "Akash", cityId: ObjectId("5fd1c41718414dfd9c0b3483")},{name: "Ashik", cityId: ObjectId("5fd1c41718414dfd9c0b3483")}])
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("5fd1c4b918414dfd9c0b3484"),
    ObjectId("5fd1c4b918414dfd9c0b3485")
  ]
}
> db.citizens.find().pretty()
{
  "_id" : ObjectId("5fd1c4b918414dfd9c0b3484"),
  "name" : "Akash",
  "cityId" : ObjectId("5fd1c41718414dfd9c0b3483")
}
{
  "_id" : ObjectId("5fd1c4b918414dfd9c0b3485"),
  "name" : "Ashik",
  "cityId" : ObjectId("5fd1c41718414dfd9c0b3483")
}
>
```

MANY to MANY Relation

```
Command Prompt - mongo
> db.products.insertOne({title: "A book", price: 12.99})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd1c80818414dfd9c0b3488")
}
> db.customers.insertOne({name: "Akash", age: 32})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd1c83e18414dfd9c0b3489")
}
> db.orders.insertOne({productId: ObjectId("5fd1c80818414dfd9c0b3488"), customerId: ObjectId("5fd1c83e18414dfd9c0b3489")})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd1c88a18414dfd9c0b348a")
}
> db.orders.drop
function(options = {}) {
  const cmdObj = Object.assign({drop: this.getName()}, options);
  ret = this._db.runCommand(cmdObj);
  if (!ret.ok) {
    if (ret.errmsg == "ns not found")
      return false;
    throw _getErrorWithCode(ret, "drop failed: " + toJson(ret));
  }
  return true;
}
> db.orders.drop()
true
> db.customers.updateOne({}, {$set: {orders: [{productId: ObjectId("5fd1c80818414dfd9c0b3488"), quantity: 2}]}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.customers.findOne()
{
  "_id" : ObjectId("5fd1c83e18414dfd9c0b3489"),
  "name" : "Akash",
  "age" : 32,
  "orders" : [
    {
      "productId" : ObjectId("5fd1c80818414dfd9c0b3488"),
      "quantity" : 2
    }
  ]
}
> db.products.findOne()
{
  "_id" : ObjectId("5fd1c80818414dfd9c0b3488"),
  "title" : "A book",
  "price" : 12.99
}
>
```

Joining using \$lookup

```
> db.books.aggregate( [ { $lookup : { from : "authors" , localField : "Authors" , foreignField : "_id" , as: "Authors" } } ]).pretty()
{
  "_id" : ObjectId("5fd1cee75151e8a105d39f43"),
  "Title" : "A book",
  "Price" : 230,
  "Authors" : [
    {
      "_id" : ObjectId("5fd1cf145151e8a105d39f44"),
      "Name" : "Asad"
    },
    {
      "_id" : ObjectId("5fd1cf145151e8a105d39f45"),
      "Name" : "Dev"
    }
  ]
}
```

Validation Schema

```
> db.createCollection("posts", {
...   validator: {
...     $jsonSchema: {
...       bsonType: "object",
...       required: ['title', 'text', 'creator', 'comments'],
...       properties: {
...         title: {
...           bsonType: 'string',
...           description: 'must be a string and it is required'
...         },
...         text: {
...           bsonType: 'string',
...           description: "must be a string and it is required"
...         },
...         creator: {
...           bsonType: 'objectId',
...           description: "Must be an object id and this is required"
...         },
...         comments: {
...           bsonType: 'array',
...           description: 'Must Be an array',
...           required: ['text', 'author'],
...           items: {
...             properties: {
...               text: {
...                 bsonType: 'string',
...                 description: "must be a string and required"
...               },
...               author: {
...                 bsonType: 'objectId',
...                 description: "must be an objectId and required"
...               }
...             }
...           }
...         }
...       }
...     }
...   }
... })
{ "ok" : 1 }
```

Validation Schema Test

```
> db.posts.insertOne({title:"My first post", text:"MongoDB is a NO Sql Database",creator:ObjectId("5fd216655151e8a105d39f50"),comments:[{text:"I will try MongoDB",author:123}]})
WriteError({
  "index" : 0,
  "code" : 121,
  "errmsg" : "Document failed validation",
  "op" : {
    "_id" : ObjectId("5fd217a25151e8a105d39f52"),
    "title" : "My first post",
    "text" : "MongoDB is a NO Sql Database",
    "creator" : ObjectId("5fd216655151e8a105d39f50"),
    "comments" : [
      {
        "text" : "I will try MongoDB",
        "author" : 123
      }
    ]
  }
}) :
WriteError({
  "index" : 0,
  "code" : 121,
  "errmsg" : "Document failed validation",
  "op" : {
    "_id" : ObjectId("5fd217a25151e8a105d39f52"),
    "title" : "My first post",
    "text" : "MongoDB is a NO Sql Database",
    "creator" : ObjectId("5fd216655151e8a105d39f50"),
    "comments" : [
      {
        "text" : "I will try MongoDB",
        "author" : 123
      }
    ]
  }
})
```

writeConcern

```
Command Prompt - mongo
> db.persons.insertOne({name: "Akhil", age: 25}, {writeConcern: {w:1}})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd20f0418414dfd9c0b34a3")
}
> db.persons.insertOne({name: "Sree", age: 22}, {writeConcern: {w:1, j: false}})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd20f3018414dfd9c0b34a4")
}
> db.persons.insertOne({name: "Sree", age: 22}, {writeConcern: {w:1, j: true}})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd20f3918414dfd9c0b34a5")
}
> db.persons.insertOne({name: "balu", age: 25}, {writeConcern: {w:1, j: true, wtimeout: 2000}})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd20fb418414dfd9c0b34a6")
}
> db.persons.insertOne({name: "balu", age: 25}, {writeConcern: {w:1, j: true, wtimeout: 1}})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5fd20fba18414dfd9c0b34a7")
}
>
```