MongoDB Query Language (MQL)

Lei Lei

What is MQL

- **FLEXIBLE query language to interact with MongoDB database**
- **Allows users to retrieve, update, delete, and manipulate documents stored in MongoDB collections.**

Most Basic Structure of query

- in Json like syntax (with a key and value pair)
  - {

    "field1": value1,

    "field2": value2,

    ...

    }

```
{
  _id: ObjectId('66027277bb01876e2445d849'),
  name: 'Alicia',
  interests: [ 'reading', 'traveling' ],
  address: { street: 'Lancaster', 'zip code': 19041 },
  class: 5
},
```

- Value could be string, int, float, boolean, array, object

Insert() method

```
db.students.insertOne({'name': 'Alicia',
                       'age': 17
                       'interests':["reading", "traveling"],
                       'address':{'street': 'Lancaster', 'zip code': 19041},
                       'class': 5})
db.students.insertMany([
  { name: 'Ryan', 'age': 15, 'interests':["reading", "traveling"] },
  { name: 'Joanna', 'age': 16, 'interest': ["photography", "hiking"],
    'course': ['physics', 'math']}
]);
```

```
mycompiler_mongodb> ... {
  acknowledged: true,
  insertedId: ObjectId('6602747c0e79b9be7202d453')
}
mycompiler_mongodb> ... ... ... ... {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6602747d0e79b9be7202d454'),
    '1': ObjectId('6602747d0e79b9be7202d455')
  }
}
```

- insertOne({field1:value, field2:value, ...})
- insertMany([{ , , ..}, { , , ...}, ...])

Flexibility: do not expect exch data has same fields

Find() method

- find(query filter) can can set a filter for the subset of results we gonna return
- if **empty** query filter, then the output show all the data in the collection

db.students.find()

```
  {
    _id: ObjectId('6602787a0fa7306e6e054c96'),
    name: 'Alicia',
    age: 17,
    interests: [ 'reading', 'traveling' ],
    address: { street: 'Lancaster', 'zip code': 19041 },
    class: 5
  },
  {
    _id: ObjectId('6602787a0fa7306e6e054c97'),
    name: 'Ryan',
    age: 15,
    interests: [ 'reading', 'traveling' ]
  },
  {
    _id: ObjectId('6602787a0fa7306e6e054c98'),
    name: 'Joanna',
    age: 16,
    interest: [ 'photography', 'hiking' ],
    course: [ 'physics', 'math' ]
  }
]
```

```
db.students.find({'age': 15});
db.students.find({'age': {$lt: 17}})
```

Query Operator

used to specify conditions

- $lt (Less Than): <
- $lte (Less Than or Equal To): <=
- $gt (Greater Than): >
- $gte (Greater Than or Equal To): >=
- $ne (Not Equal To): ≠
- $exists (Exist):
- $in (In):{ field: { $in: [value1, value2, ...] } }
- $nin (Not In):
- $and, $or

```
mycompiler_mongodb> [
  {
    _id: ObjectId('660279e38bd58ea10ad3f003'),
    name: 'Ryan',
    age: 15,
    interests: [ 'reading', 'traveling' ]
  }
]
mycompiler_mongodb> [
  {
    _id: ObjectId('660279e38bd58ea10ad3f003'),
    name: 'Ryan',
    age: 15,
    interests: [ 'reading', 'traveling' ]
  },
  {
    _id: ObjectId('660279e38bd58ea10ad3f004'),
    name: 'Joanna',
    age: 16,
    interest: [ 'photography', 'hiking' ],
    course: [ 'physics', 'math' ]
  }
]
```

```
db.students.findOne({ name: 'Alicia' });
db.students.findOneAndUpdate(
  { 'name': 'Ryan' }, // filter criteria
  { $set: { age: 16 } }, // update operation
);
db.students.findOneAndDelete({ name: 'Joanna' });
db.students.find()
```

```
  {
    _id: ObjectId('660280b65aaad730b652f984'),
    name: 'Alicia',
    age: 17,
    interests: [ 'reading' ],
    address: { street: 'Lancaster', 'zip code': 19041 },
    class: 5
  },
  {
    _id: ObjectId('660280b65aaad730b652f985'),
    name: 'Ryan',
    age: 16,
    interests: [ 'reading', 'traveling', 'cooking' ]
  }
]
```

Little Discussion

- Who would this find() output

```
db.students.find({
  $and: [
    { 'interests': 'reading' },
    {
      $or: [
        { 'class': 5 },
        { 'age': { $gte: 16}}
      ]
    }
  ]
})
```

```
  _id: ObjectId('66028418cc2da750acd71162'),
  name: 'Alicia',
  age: 17,
  interests: [ 'reading' ],
  address: { street: 'Lancaster', 'zip code': 19041 },
  class: 5
},
{
  _id: ObjectId('66028419cc2da750acd71163'),
  name: 'Ryan',
  age: 15,
  interests: [ 'reading', 'traveling', 'cooking' ],
  class: 5
},
{
  _id: ObjectId('66028419cc2da750acd71164'),
  name: 'Joanna',
  age: 16,
  interests: [ 'photography', 'hiking' ],
  course: [ 'physics', 'math' ],
  class: 4
}
```

Delete()

- Syntax
    - deleteOne/Many({ filter query})

```
db.students.deleteOne({ name: 'Alicia' });
db.students.deleteMany({ age: { $lt: 16 } });
db.students.deleteMany({}); //no criteria, then delete all
```

Update()

- Syntax

  updateOne/Many(

  { filter query to apply update} ,

  { $set: { : }

  );

```
db.students.updateOne(
  { name: 'Alicia' }, // Filter criteria
  { $set: { age: 18 } } // Update operation
);
db.students.updateMany(
  { age: { $lt: 16 } }, // Filter criteria
  { $set: { status: 'inactive' } } // Update operation
);
db.students.updateMany(
  {}, // No filter criteria to update all documents
  { $inc: { age: 1 } } // Increment operation
);
db.students.find()
```

Wrap-up

- Basic structure of a query
- Insert()
- Find()
- Update()
- Delete()

Any Questions?

Thank you!!