

# Retrieve Data Using Queries

---



**Mark Scott**

AUTHOR

@tripletdad99





## Retrieve documents

### Model methods

- find
- findById
- findOne
- where

## Updating and removing documents

# Documents and Helper Methods

---



# Model Methods



**find**

**findOne**

**findById**

**where**

# Model.find(...)

```
const Standup = require('../models/standup.js');

// No callback... Deferred execution
let query = Standup.find({});
query.exec(function (err, results) { ... });

// With callback... Executes query immediately
Standup.find({}, function (err, results) {
  // handle the error... Or results here
});

// With callback and query conditions
Standup.find({ memberName: 'David' }, function (err, results) {
  // handle the error... Or results here
});

// Limit the returned fields...
Standup.find({ memberName: 'Mary' }, 'memberName impediment', function (err, results) {
  // handle the error... Or results here
});
```

## Parameters

conditions	required
[projection]	optional
[options]	optional
[callback]	optional



# Model.findOne(...)

## Parameters

conditions	required
[projection]	optional
[options]	optional
[callback]	optional

```
const Standup = require('../models/standup.js');

// With callback and query conditions
Standup.findOne({ memberName: 'David' }, function (err, results) {
  // handle the returned docs and/or errors
});

// Limit the returned fields...
Standup.findOne({ memberName: 'Mary' }, 'memberName impediment',
  function (err, results) {
    // handle the error... Or results here
  });
```



# Model.findById(...)

## Parameters

id	required
[projection]	optional
[options]	optional
[callback]	optional

```
const Standup = require('../models/standup.js');

// No callback... Deferred execution
let query = Standup.findById(id);
query.exec(function (err, results) {
    // handle the error... Or results here
});

// Same as above – using chained method calls
Standup.findById(id).exec(function (err, results) {
    // handle the error... Or results here
});

// Return every field EXCEPT impediment
let query = Standup.findById(id, '-impediment');
```



# Comparison Query Operators

**\$gt**            greater than

**\$gte**        greater than or equal to

**\$in**          exists in

**\$lt**          less than

**\$lte**        less than or equal to

**\$ne**         not equal to

**\$nin**        does not exist

## Note:

- These are the same as what is found in MongoDB





```
// Example: find customers with discount >= 10%...
```

```
Customer.find({discount: {$gte: 10}}, function(err,  
results) {  
    if (err) throw err;  
    console.log(results);  
});
```

\$gte

Greater than or equal to example



# Model.where(...)

## Parameters

path	required
[val]	optional

```
const Customer = require('../models/customer.js');
```

```
Customer.find({discount: {$gte: 10, $lt: 20}}, function(err, results) {  
  if (err) throw err;  
  console.log(results);  
});
```

```
Customer.where('discount').gte(10).lt(20).exec(function(err, results) {  
  if (err) throw err;  
  console.log(results);  
});
```

```
// Chain where methods together...  
Customer.where('discount').gte(10).lt(20)  
  .where('zipCode', '12345')  
  .exec(function(err, results) {  
    if (err) throw err;  
    console.log(results);  
  });
```



# Updating and Removing Documents

---



```
Standup.findById(id).exec(function (err, doc) {  
  // handle any errors  
  if (err) return errorHandler(err);  
  
  // update the found document  
  doc.impediment('None');  
  doc.save(function (err) {  
    if (err) return errorHandler(err);  
    console.log('Document updated');  
  });  
});
```

## Updating a Document

One possible approach



# Model Methods



**updateOne**

**updateMany**

**deleteOne**

**deleteMany**

**findByIdAndUpdate**

**findByIdAndDelete**

# Model.updateOne(...)

## Parameters

conditions	required
doc	required
[options]	optional
[callback]	optional

```
const Standup = require('../models/standup.js');

const qry = { memberName: 'Marry' };

const update = {memberName : 'Mary' };

// Only updates the first document matching query criteria
Standup.updateOne(qry, update, function (err, rawResponse) {
  // Handle error or raw results here...
});
```



- [hint](#)

The following options are only for write operations: `update()`, `updateOne()`, `updateMany()`, `replaceOne()`, `findOneAndUpdate()`, and `findByIdAndUpdate()`:

- [upsert](#)
- [writeConcern](#)
- [timestamps](#): If `timestamps` is set in the schema, set this option to `false` to skip timestamps for that particular update. Has no effect if `timestamps` is not enabled in the schema options.

The following options are only for `find()`, `findOne()`, `findById()`, `findOneAndUpdate()`, and `findByIdAndUpdate()`:

- [lean](#)

The following options are only for all operations **except** `update()`, `updateOne()`, `updateMany()`, `remove()`, `deleteOne()`, and `deleteMany()`:

- [maxTimeMS](#)

## The following options are for all operations

- [collation](#)
- [session](#)

# Model.updateMany(...)

```
const Standup = require('../models/standup.js');

const qry = { impediment : '' };
const update = { impediment: 'None' };

// update all documents that match query
Standup.updateMany(qry, update, function (err, rawResponse) {
  // Handle error or raw results here...
});

// same, just using inline parameter objects
Standup.updateMany(
  { impediment : '' },
  { impediment: 'None' },
  function (err, rawResponse) {
    // Handle error or raw results here...
  });
```

## Parameters

conditions	required
doc	required
[options]	optional
[callback]	optional





# Model.deleteOne(...)

## Parameters

conditions  
[callback]

required  
optional

```
const Standup = require('../models/project.js');  
  
const qry = { isActive: false, name: 'Demo test' };  
  
// deletes the FIRST document that matches query  
Standup.deleteOne(qry, function (err) {  
    // Handle any errors here...  
});
```



# Model.deleteMany(...)

## Parameters

conditions	required
[callback]	optional

```
const Standup = require('../models/project.js');

const qry = { isActive: false };

// deletes all documents that match query
Standup.deleteMany(qry, function (err) {
  // Handle any errors here...
});
```



# Model Methods



**Let's multi-task**

**findByIdAndUpdate**

**findByIdAndDelete**

# Model.findByIdAndUpdate(...)

```
const Standup = require('../models/standup.js');

const id = // Object, Number or String value passed in
const update = { project: 'Mongoose Demo' };

// delete one document, found by Id
Standup.findByIdAndUpdate(id, update, function (err, doc) {
  // Handle any errors here...
});
```

## Parameters

id	required
[update]	optional
[options]	optional
[callback]	optional



# Model.findByIdAndDelete(...)

## Parameters

id	required
[options]	optional
[callback]	optional

```
const Standup = require('../models/standup.js');

const id = // Object, Number or String value passed in

// delete one document, found by Id
Standup.findByIdAndDelete(id, function (err, doc) {
  // Handle any errors here...
});
```



placeholder



# Demo



## Review Demo Application

### Querying documents

- 12 newest standup notes
- Team member list
- Project list (active only)

