db.movies.find()

{ \_id: ObjectId("6077d5b1a8b8f1cfa9d32959"),

title: 'Fight Club',

writer: 'Chuck Palahniko',

year: '1999',

actors: [ 'BrandPitt', 'Edward Norton' ] }

{ \_id: ObjectId("6077de90a8b8f1cfa9d3295b"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: '1994',

actors: [ 'John Travolta', 'Uma Thurman' ],

actor: 'Samuel L.Jackson' }

{ \_id: ObjectId("6077e0fda8b8f1cfa9d3295c"),

title: 'inglorious basterds',

writer: 'quentin tarantino',

year: '2009',

actors: [ 'Brand Pitt', 'Diane Krunger', 'Eli Roth' ] }

{ \_id: ObjectId("6077e19aa8b8f1cfa9d3295d"),

title: 'The Hobbit An Unexpected Journey',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit' }

{ \_id: ObjectId("6077e1c8a8b8f1cfa9d3295e"),

title: 'The Hobbit The Desolation of Smaug',

writer: 'J.R.R.Tolkein',

year: '2013',

franchise: 'The Hobbit' }

{ \_id: ObjectId("6077e272a8b8f1cfa9d3295f"),

title: 'The Hobbit The Battle of the Five Armies',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit',

synopsis: 'Bilbo ans Company are forced to engage in a war against an array of combatants and keep the lonely Mountain from falling into the hands of a rising darkness' }

{ \_id: ObjectId("6077e371a8b8f1cfa9d32960"),

title: 'Pee was Hermans Big Advantages' }

{ \_id: ObjectId("6077e38ca8b8f1cfa9d32961"), title: 'Avatar' }

db.movies.find({writer:"Quentin Tarantino"})

{ \_id: ObjectId("6077de90a8b8f1cfa9d3295b"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: '1994',

actors: [ 'John Travolta', 'Uma Thurman' ],

actor: 'Samuel L.Jackson' }

db.movies.find({actors:"Brand Pitt"})

{ \_id: ObjectId("6077e0fda8b8f1cfa9d3295c"),

title: 'inglorious basterds',

writer: 'quentin tarantino',

year: '2009',

actors: [ 'Brand Pitt', 'Diane Krunger', 'Eli Roth' ] }

db.movies.find({franchise:"The Hobbit"})

{ \_id: ObjectId("6077e19aa8b8f1cfa9d3295d"),

title: 'The Hobbit An Unexpected Journey',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit' }

{ \_id: ObjectId("6077e1c8a8b8f1cfa9d3295e"),

title: 'The Hobbit The Desolation of Smaug',

writer: 'J.R.R.Tolkein',

year: '2013',

franchise: 'The Hobbit' }

{ \_id: ObjectId("6077e272a8b8f1cfa9d3295f"),

title: 'The Hobbit The Battle of the Five Armies',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit',

synopsis: 'Bilbo ans Company are forced to engage in a war against an array of combatants and keep the lonely Mountain from falling into the hands of a rising darkness' }

db.movies.find({year:{$gt:"1990",$lt:"2000"}})

{ \_id: ObjectId("6077d5b1a8b8f1cfa9d32959"),

title: 'Fight Club',

writer: 'Chuck Palahniko',

year: '1999',

actors: [ 'BrandPitt', 'Edward Norton' ] }

{ \_id: ObjectId("6077de90a8b8f1cfa9d3295b"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: '1994',

actors: [ 'John Travolta', 'Uma Thurman' ],

actor: 'Samuel L.Jackson' }

db.movies.find({$or:[{year:{$gt:"2010"}},{year: {$lt:"2000"}}]})

{ \_id: ObjectId("6077d5b1a8b8f1cfa9d32959"),

title: 'Fight Club',

writer: 'Chuck Palahniko',

year: '1999',

actors: [ 'BrandPitt', 'Edward Norton' ] }

{ \_id: ObjectId("6077de90a8b8f1cfa9d3295b"),

title: 'Pulp Fiction',

writer: 'Quentin Tarantino',

year: '1994',

actors: [ 'John Travolta', 'Uma Thurman' ],

actor: 'Samuel L.Jackson' }

{ \_id: ObjectId("6077e19aa8b8f1cfa9d3295d"),

title: 'The Hobbit An Unexpected Journey',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit' }

{ \_id: ObjectId("6077e1c8a8b8f1cfa9d3295e"),

title: 'The Hobbit The Desolation of Smaug',

writer: 'J.R.R.Tolkein',

year: '2013',

franchise: 'The Hobbit' }

{ \_id: ObjectId("6077e272a8b8f1cfa9d3295f"),

title: 'The Hobbit The Battle of the Five Armies',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit',

synopsis: 'Bilbo ans Company are forced to engage in a war against an array of combatants and keep the lonely Mountain from falling into the hands of a rising darkness' }

db.movies.update({\_id:ObjectId("5c9f98e5e5c2dfe9b3729bfe")}, {$set:{synopsis:"A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."}})

'DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany or bulkWrite.'

{ acknowledged: true,

insertedId: null,

matchedCount: 0,

modifiedCount: 0,

upsertedCount: 0 }

db.movies.update({\_id:ObjectId("5c9fa42ae5c2dfe9b3729c03")}, {$set:{synopsis:"The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of a mysterious and magical ring."}})

{ acknowledged: true,

insertedId: null,

matchedCount: 0,

modifiedCount: 0,

upsertedCount: 0 }

db.movies.update({\_id:ObjectId("5c9f983ce5c2dfe9b3729bfc")}, {$push:{actors:"Samuel L. Jackson"}})

{ acknowledged: true,

insertedId: null,

matchedCount: 0,

modifiedCount: 0,

upsertedCount: 0 }

db.movies.find({synopsis:{$regex:"Bilbo"}})

{ \_id: ObjectId("6077e272a8b8f1cfa9d3295f"),

title: 'The Hobbit The Battle of the Five Armies',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit',

synopsis: 'Bilbo ans Company are forced to engage in a war against an array of combatants and keep the lonely Mountain from falling into the hands of a rising darkness' }

db.movies.find({synopsis:{$regex:"Gandalf"}})

db.movies.find({synopsis:{$regex:"Gandalf"}})

db.movies.find({$and:[{synopsis:{$regex:"Bilbo"}}, {synopsis:{$not:/Gandalf/}}]})

{ \_id: ObjectId("6077e272a8b8f1cfa9d3295f"),

title: 'The Hobbit The Battle of the Five Armies',

writer: 'J.R.R.Tolkein',

year: '2012',

franchise: 'The Hobbit',

synopsis: 'Bilbo ans Company are forced to engage in a war against an array of combatants and keep the lonely Mountain from falling into the hands of a rising darkness' }

db.movies.find({$or:[{synopsis:{$regex:"dwarves"}}, {synopsis:{$regex:"hobbit"}}]})

db.movies.find({$and:[{synopsis:{$regex:"gold"}}, {synopsis:{$regex:"dragon"}}]})

db.movies.remove({\_id:ObjectId("5c9f992ae5c2dfe9b3729c00")})

'DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany or bulkWrite.'

{ acknowledged: true, deletedCount: 0 }

db.movies.remove({\_id:ObjectId("5c9f9936e5c2dfe9b3729c01")})

{ acknowledged: true, deletedCount: 0 }

db.createCollection("users")

{ ok: 1 }

db.users.insert({\_id:1,username:"GoodGuyGreg", first\_name:"Good Guy", last\_name:"Greg"})

'DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany or bulkWrite.'

{ acknowledged: true, insertedIds: { '0': 1 } }

db.users.insert({\_id:2, username:"ScumbagSteve", fullname:{first: "Scumbag", last:"Steve"}})

{ acknowledged: true, insertedIds: { '0': 2 } }

db.createCollection("posts")

{ ok: 1 }

db.posts.insert({username:"GoodGuyGreg", title:"Passes out at Party", body:"Raises your credit score"})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077fe74a8b8f1cfa9d32962") } }

db.posts.insert({ username:"GoodGuyGreg", title:"Steals your identity", body:"Raises your credit score"})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077fe89a8b8f1cfa9d32963") } }

db.posts.insert({username:"GoodGuyGreg", title:"Reports a bug in your code", body:"Sends you a pull request"})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077fe9ea8b8f1cfa9d32964") } }

db.posts.insert({ username:"ScumbagSteve", title:"Borrows something", body:"Sells it"})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077feaca8b8f1cfa9d32965") } }

db.posts.insert({ username:"ScumbagSteve", title:"Borrows everything", body:"The end"})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077feb7a8b8f1cfa9d32966") } }

db.posts.insert({username:"ScumbagSteve", title:"Forks your repo on github", body:"Sets to private"})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077fec2a8b8f1cfa9d32967") } }

db.createCollection("comments")

{ ok: 1 }

db.comments.insert({ username:"GoodGuyGreg", comment:"Hope you got a good deal!", post:ObjectId("5ca0b7e96435f98b5901f463")})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077feeaa8b8f1cfa9d32968") } }

db.comments.insert({username:"GoodGuyGreg", comment:"What's mine is yours!", post:ObjectId("5ca0b9706435f98b5901f46a")})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077fef6a8b8f1cfa9d32969") } }

db.comments.insert({username:"GoodGuyGreg", comment:"Don't violate the licensing agreement!", post:ObjectId("5ca0b8766435f98b5901f467")})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077ff10a8b8f1cfa9d3296a") } }

db.comments.insert({username:"ScumbagSteve", comment:"It still isn't clean", post:ObjectId("5ca0b8546435f98b5901f466")})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077ff45a8b8f1cfa9d3296b") } }

db.comments.insert({username:"ScumbagSteve", comment:"Denied your PR cause I found a hack", post:ObjectId("5ca0b9256435f98b5901f469")})

{ acknowledged: true,

insertedIds: { '0': ObjectId("6077ff61a8b8f1cfa9d3296c") } }

db.users.find().pretty()

{ \_id: 1,

username: 'GoodGuyGreg',

first\_name: 'Good Guy',

last\_name: 'Greg' }

{ \_id: 2,

username: 'ScumbagSteve',

fullname: { first: 'Scumbag', last: 'Steve' } }

db.posts.find().pretty()

{ \_id: ObjectId("6077fe74a8b8f1cfa9d32962"),

username: 'GoodGuyGreg',

title: 'Passes out at Party',

body: 'Raises your credit score' }

{ \_id: ObjectId("6077fe89a8b8f1cfa9d32963"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score' }

{ \_id: ObjectId("6077fe9ea8b8f1cfa9d32964"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request' }

{ \_id: ObjectId("6077feaca8b8f1cfa9d32965"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it' }

{ \_id: ObjectId("6077feb7a8b8f1cfa9d32966"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end' }

{ \_id: ObjectId("6077fec2a8b8f1cfa9d32967"),

username: 'ScumbagSteve',

title: 'Forks your repo on github',

body: 'Sets to private' }

db.posts.find({username:"GoodGuyGreg"})

{ \_id: ObjectId("6077fe74a8b8f1cfa9d32962"),

username: 'GoodGuyGreg',

title: 'Passes out at Party',

body: 'Raises your credit score' }

{ \_id: ObjectId("6077fe89a8b8f1cfa9d32963"),

username: 'GoodGuyGreg',

title: 'Steals your identity',

body: 'Raises your credit score' }

{ \_id: ObjectId("6077fe9ea8b8f1cfa9d32964"),

username: 'GoodGuyGreg',

title: 'Reports a bug in your code',

body: 'Sends you a pull request' }

db.posts.find({username:"ScumbagSteve"})

{ \_id: ObjectId("6077feaca8b8f1cfa9d32965"),

username: 'ScumbagSteve',

title: 'Borrows something',

body: 'Sells it' }

{ \_id: ObjectId("6077feb7a8b8f1cfa9d32966"),

username: 'ScumbagSteve',

title: 'Borrows everything',

body: 'The end' }

{ \_id: ObjectId("6077fec2a8b8f1cfa9d32967"),

username: 'ScumbagSteve',

title: 'Forks your repo on github',

body: 'Sets to private' }

db.comments.find().pretty()

{ \_id: ObjectId("6077feeaa8b8f1cfa9d32968"),

username: 'GoodGuyGreg',

comment: 'Hope you got a good deal!',

post: ObjectId("5ca0b7e96435f98b5901f463") }

{ \_id: ObjectId("6077fef6a8b8f1cfa9d32969"),

username: 'GoodGuyGreg',

comment: 'What\'s mine is yours!',

post: ObjectId("5ca0b9706435f98b5901f46a") }

{ \_id: ObjectId("6077ff10a8b8f1cfa9d3296a"),

username: 'GoodGuyGreg',

comment: 'Don\'t violate the licensing agreement!',

post: ObjectId("5ca0b8766435f98b5901f467") }

{ \_id: ObjectId("6077ff45a8b8f1cfa9d3296b"),

username: 'ScumbagSteve',

comment: 'It still isn\'t clean',

post: ObjectId("5ca0b8546435f98b5901f466") }

{ \_id: ObjectId("6077ff61a8b8f1cfa9d3296c"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("5ca0b9256435f98b5901f469") }

db.comments.find({username:"GoodGuyGreg"})

{ \_id: ObjectId("6077feeaa8b8f1cfa9d32968"),

username: 'GoodGuyGreg',

comment: 'Hope you got a good deal!',

post: ObjectId("5ca0b7e96435f98b5901f463") }

{ \_id: ObjectId("6077fef6a8b8f1cfa9d32969"),

username: 'GoodGuyGreg',

comment: 'What\'s mine is yours!',

post: ObjectId("5ca0b9706435f98b5901f46a") }

{ \_id: ObjectId("6077ff10a8b8f1cfa9d3296a"),

username: 'GoodGuyGreg',

comment: 'Don\'t violate the licensing agreement!',

post: ObjectId("5ca0b8766435f98b5901f467") }

db.comments.find({username:"ScumbagSteve"})

{ \_id: ObjectId("6077ff45a8b8f1cfa9d3296b"),

username: 'ScumbagSteve',

comment: 'It still isn\'t clean',

post: ObjectId("5ca0b8546435f98b5901f466") }

{ \_id: ObjectId("6077ff61a8b8f1cfa9d3296c"),

username: 'ScumbagSteve',

comment: 'Denied your PR cause I found a hack',

post: ObjectId("5ca0b9256435f98b5901f469") }

mongoimport --db mongo\_practice --collection users --file