QUIZ

What's the difference between the MongoDB methods `.updateOne()` and `.replaceOne()`?

With `.replaceOne()`, fields in the existing document not included in the replacement document will be lost. With `.updateOne()` new fields can be added without losing the ones in the old document.

👏 Correct! `replaceOne()` replaces the whole document, so any unused fields will not populate.

`.replaceOne()` deletes a single document from a collection, whereas `.updateOne()` updates certain fields.

`.replaceOne()` only replaces existing fields, whereas `.updateOne()` can remove certain fields in a document by omitting them.

`.replaceOne()` can only add new fields. `.updateOne()` only updates existing fields.

Complete the MongoDB command to replace a document with a field `name` with the value `"Sally Jones"` with the following new document:

```
{
  name: "Sally Jones",
  blocked: true
}
```

```
db.contacts.replaceOne(
  { name: "Sally Jones" },
  { name: "Sally Jones", blocked: true }
)
```

👏 You got it!

Consider a MongoDB collection named `movies` with the following documents:

```
{
  _id: ObjectId("62a0fdb6f2d87d9a48b773c1")
  title: "Fight Club",
  year: 1999
},
{
  _id: ObjectId("62a0fdc0f8145542150299b3")
  title: "The Green Mile",
  year: 1999
 },
{
  _id: ObjectId("62a0fdcc200e65cc029210ce")
  title: "Cast Away",
  year: 2000
 }
```

How many, and which movie(s)/document(s) will remain if we run the following command?

```
db.movies.deleteOne({ year: 1999 });
```

One document: `"Cast Away"`.

Two documents: `"Fight Club"` and `"Cast Away"`.

Three. No changes will be implemented.

Two documents: `"The Green Mile"` and `"Cast Away"`.

That's right! `.deleteOne()` will delete the first matching document from the collection.

What would happen to the `employees` collection if we run the following MongoDB command:

```
db.employees.deleteMany({})
```

No documents will be deleted.

The first document in the collection will be deleted.

It will produce a syntax error.

It will delete all existing documents in the collection.

Correct! Passing an empty filter argument will delete all documents from a collection.

Which of the following MongoDB commands will delete all documents with a field `brand` and the value of `"Kodak"` from a collection named `cameras`?

```
db.cameras.deleteOne({ brand: "Kodak" })
```

```
db.cameras.drop()
```

```
db.cameras.updateMany({ brand: "Kodak" })
```

```
db.cameras.deleteMany({ brand: "Kodak" })
```

Correct! All documents that match a condition can be deleted from a collection by using the `.deleteMany()` method.

Which of the following MongoDB commands would delete the following document from a collection called `superheroes`:

```
{
  _id: ObjectId("62a10022825d7e1a9f096f1a"),
  name: "Spiderman",
  power_level: 85
}
```

```
db.superheroes.drop({ name: "Spiderman" })
```

```
db.superheroes.replaceOne({ name: "Spiderman" }, {})
```

```
db.superheroes.deleteOne("Spiderman")
```

```
db.superheroes.deleteOne({ name: "Spiderman" })
```

Correct! `.deleteOne()` deletes the first document to match the selection criteria, in this case, the name of the superhero.

Consider a MongoDB collection named `movies` with the following documents:

```
{
  _id: ObjectId("62a0fdb6f2d87d9a48b773c1")
  title: "Fight Club",
  year: 1999
},
{
  _id: ObjectId("62a0fdc0f8145542150299b3")
  title: "The Green Mile",
  year: 1999
 },
{
  _id: ObjectId("62a0fdcc200e65cc029210ce")
  title: "Cast Away",
  year: 2000
 }
```

Which command deletes all documents from the above collection from the year 1999?

```
db.movies.deleteMany([ { year: 1999 } ]);
```

```
db.movies.deleteMany({ year: 1999 });
```

Correct! `.deleteMany()` will delete all the documents that match specific criteria in the first parameter. In this case, all the movies with a `year` field with the value `1999`.