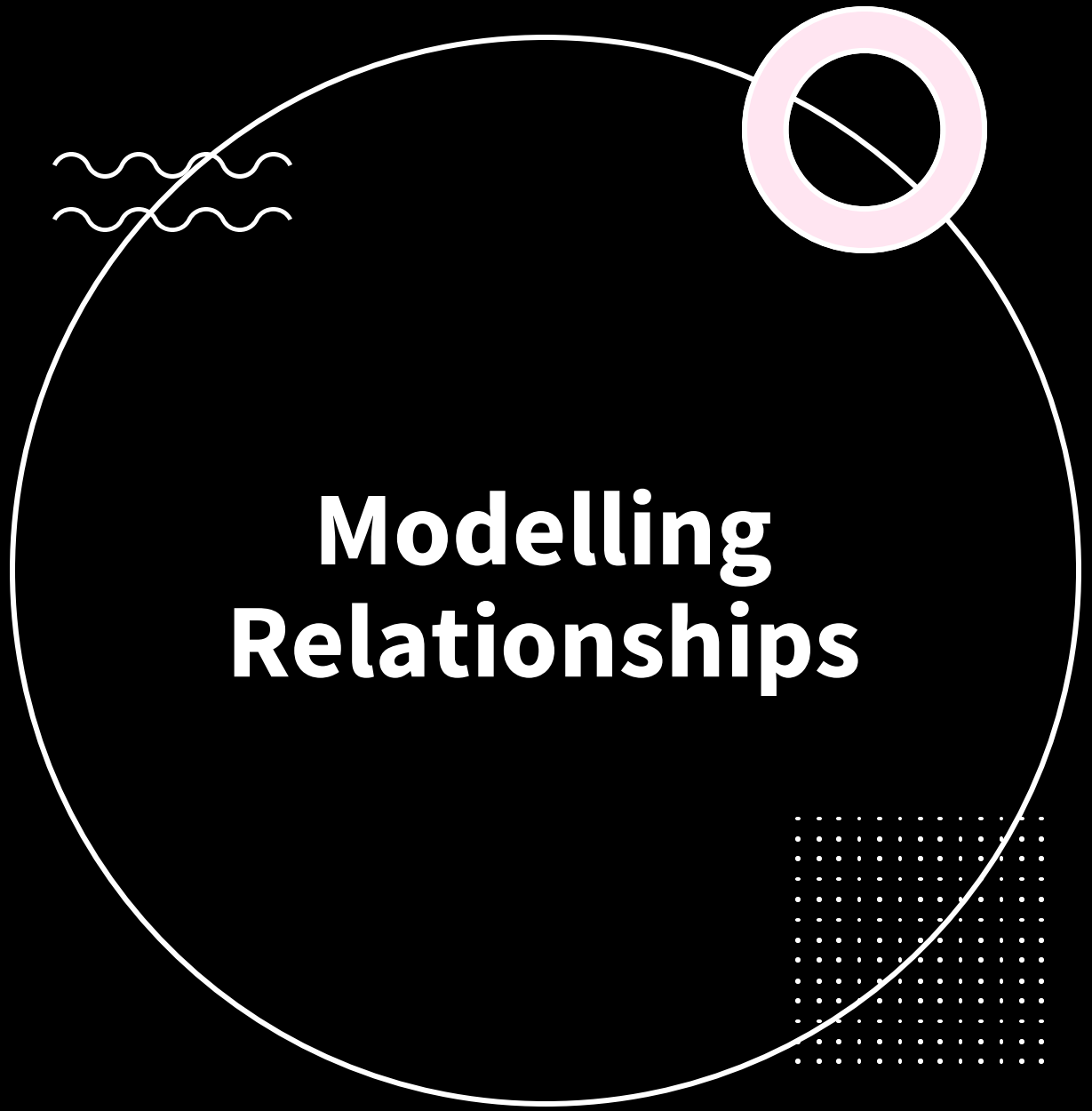
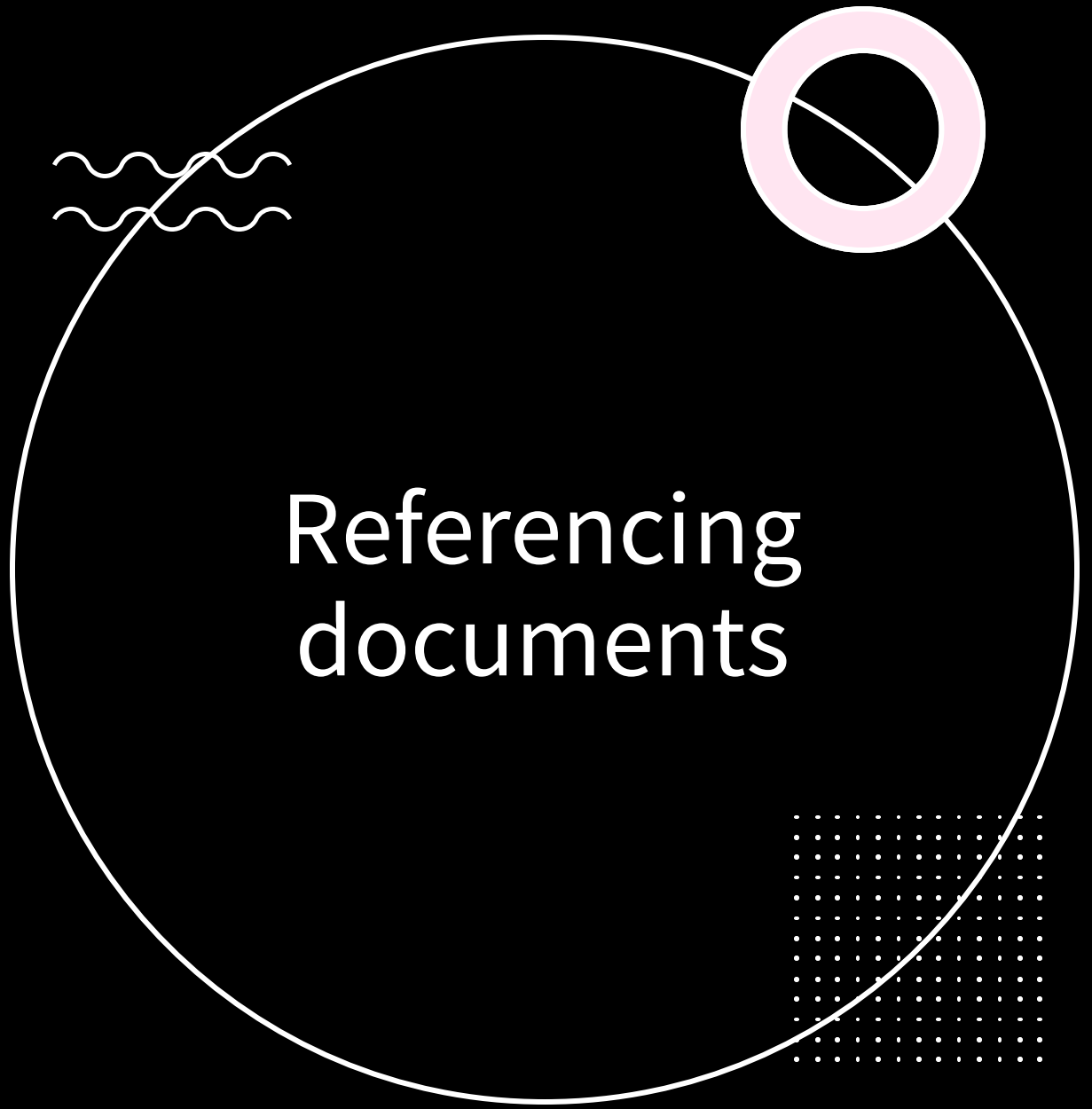
MONGOOSE-MODELING RELATIONSHIPS BETWEEN CONNECTED DATA



Modelling Relationships

To model relationships between connected data, we can either reference a document or embed it in another document

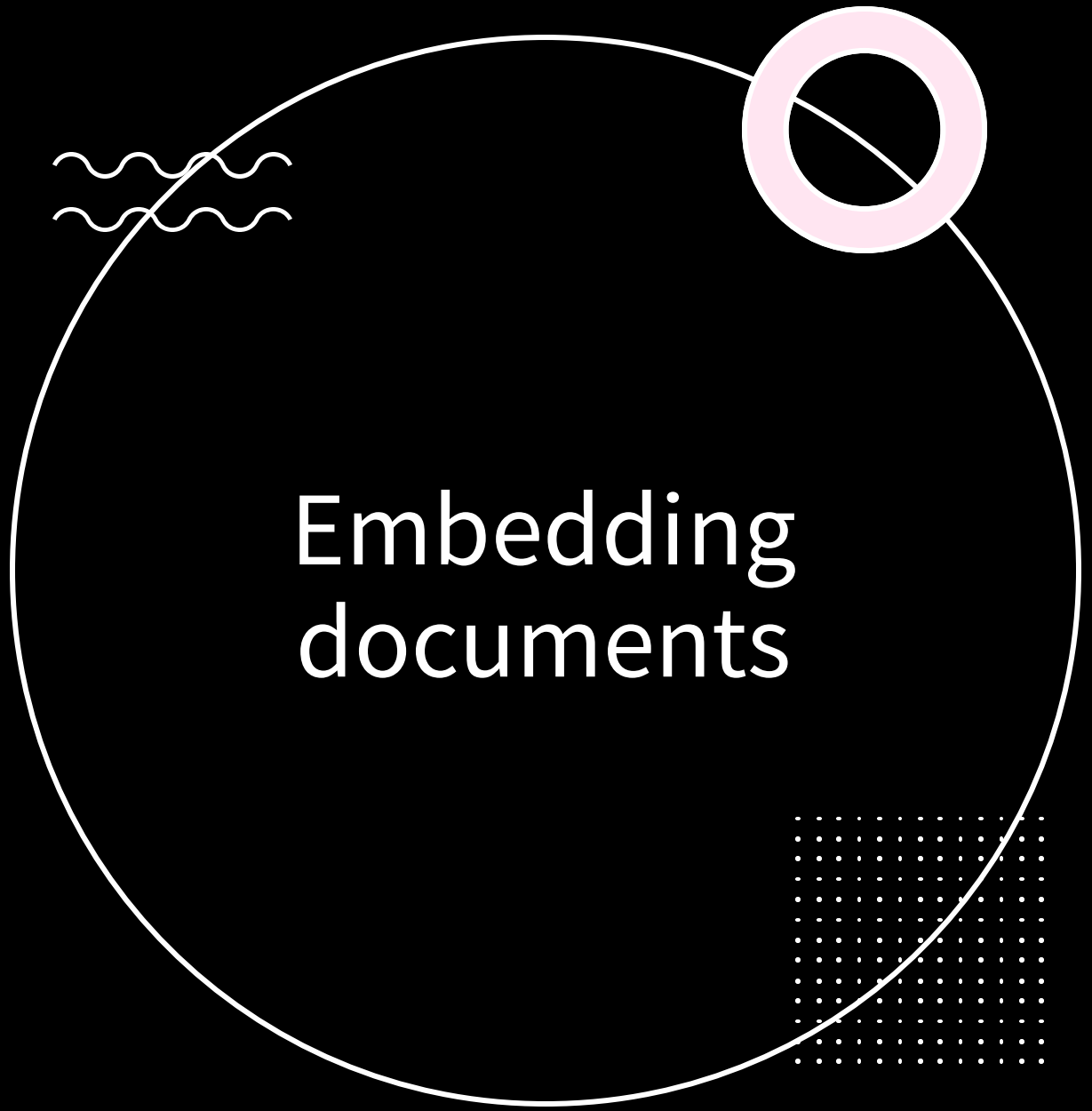


Referencing documents

Referencing documents (normalization) is a good approach when you want to enforce data consistency. Because there will be a single instance of an object in the database.

But this approach has a negative impact on the performance of your queries because in MongoDB we cannot JOIN documents as we do in relational databases.

So, to get a complete representation of a document with its related documents, we need to send multiple queries to the database



Embedding documents

Embedding documents (denormalization) solves this issue.

We can read a complete representation of a document with a single query.

All the necessary data is embedded in one document and its children.

But this also means we'll have multiple copies of data in different places.

While storage is not an issue these days, having multiple copies means changes made to the original document may not propagate to all copies.

If the database server dies during an update, some documents will be inconsistent.

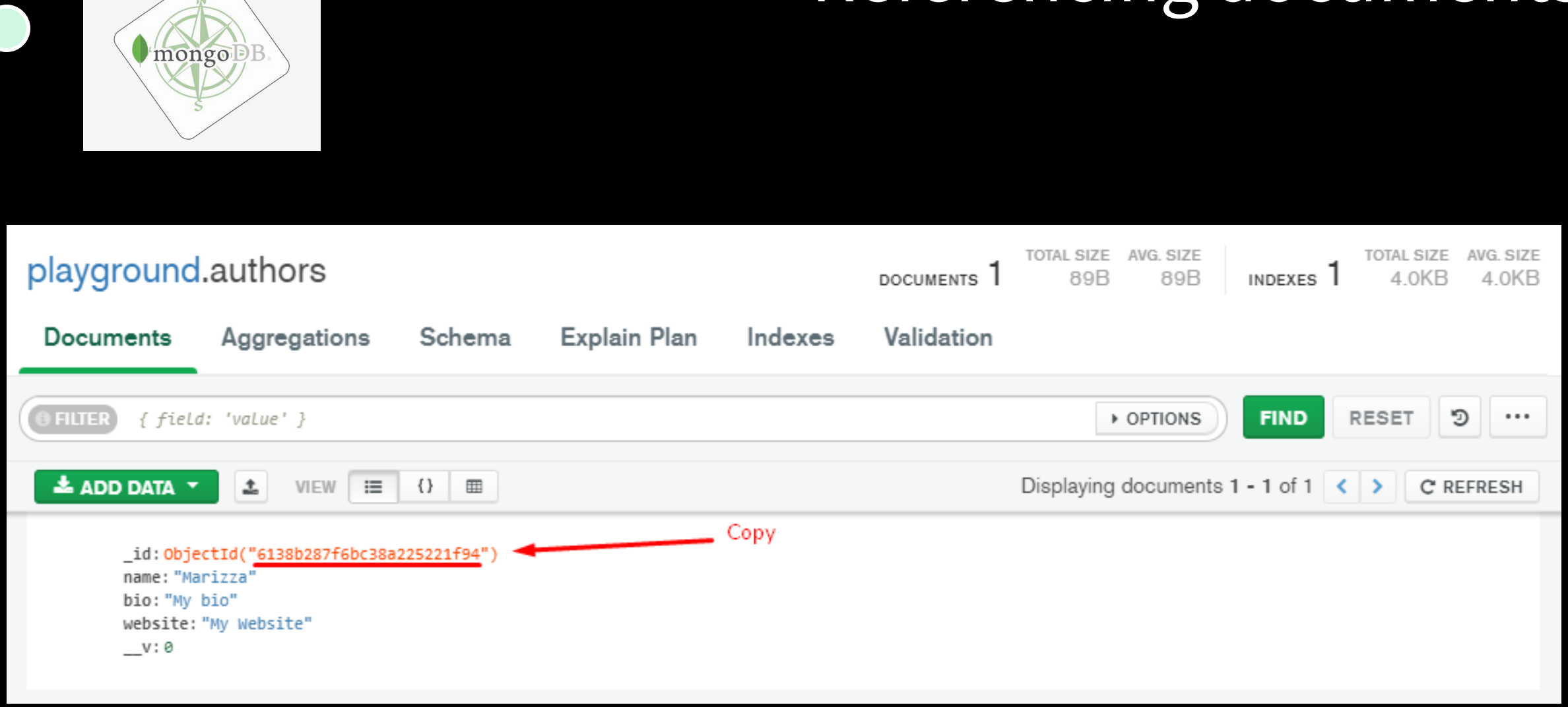
For every business, for every problem, you need to ask this question: "can we tolerate data being inconsistent for a short period of time?" If not, you'll have to use references. But again, this means that your queries will be slower

population.js

Referencing documents

```
const mongoose = require('mongoose');
mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));
const Author = mongoose.model('Author', new mongoose.Schema({
  name: String,
  bio: String,
  website: String
}));
const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
}));
async function createAuthor(name, bio, website) {
  const author = new Author({
    name,
    bio,
    website
  });
  const result = await author.save();
  console.log(result);
}
async function createCourse(name, author) {
  const course = new Course({
    name,
    author
  });

  const result = await course.save();
  console.log(result);
}
async function listCourses() {
  const courses = await Course
    .find()
    .select('name');
  console.log(courses);
}

createAuthor('Marizza', 'My bio', 'My Website');
// createCourse('Node Course', 'authorId')
// listCourses();
```

Referencing documents



Referencing documents

population.js

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const Author = mongoose.model('Author', new mongoose.Schema({
  name: String,
  bio: String,
  website: String
}));

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
}));

async function createAuthor(name, bio, website) {
...
}

async function createCourse(name, author) {
...
}

async function listCourses() {
...
}

// createAuthor('Marizza', 'My bio', 'My Website');

createCourse('Node Course', '6138b287f6bc38a225221f94')

// listCourses();
```

Paste

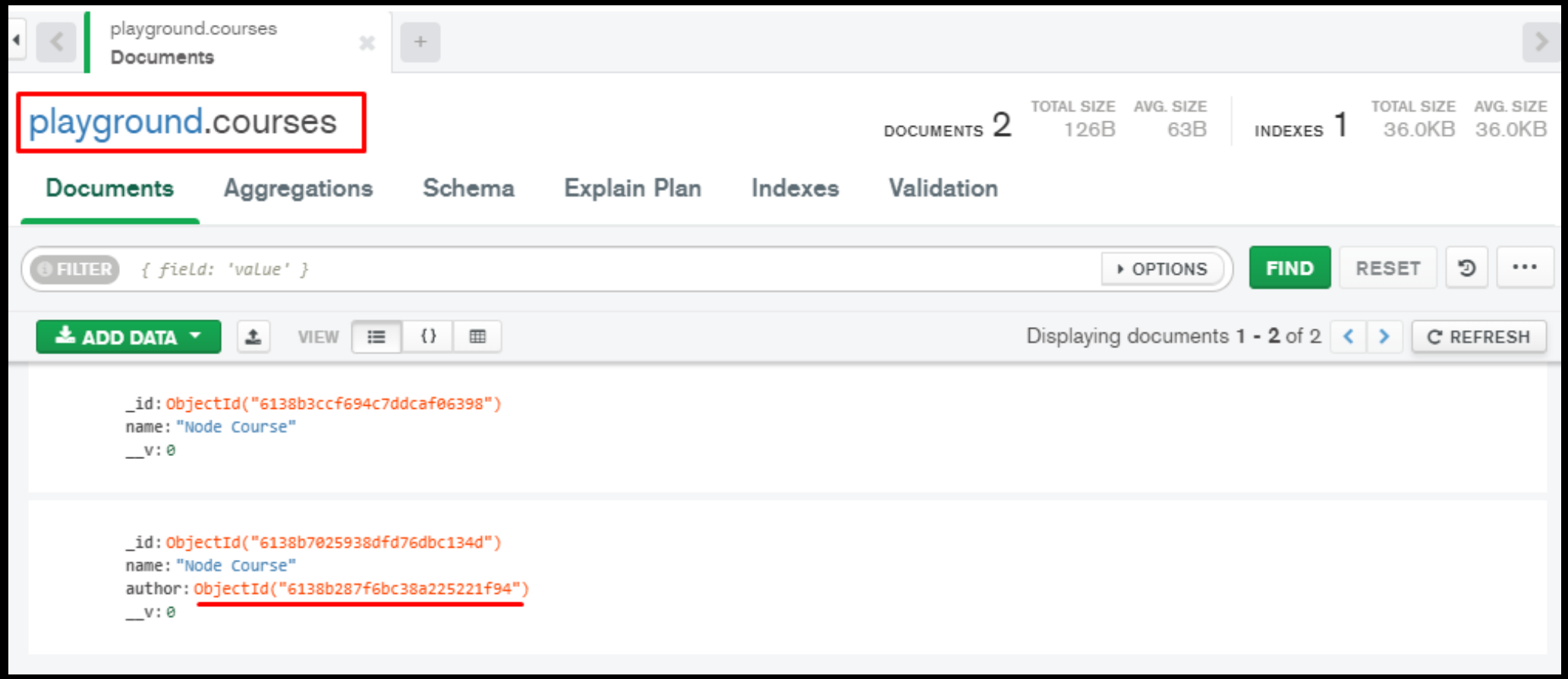
population.js

Referencing documents

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const Author = mongoose.model('Author', new mongoose.Schema({
  name: String,
  bio: String,
  website: String
}));

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  author: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Author'
  }
}));

async function createAuthor(name, bio, website) {
...
}
async function createCourse(name, author) {
...
}
async function listCourses() {
...
}

// createAuthor('Marizza', 'My bio', 'My Website');
createCourse('Node Course', '6138b287f6bc38a225221f94')
// listCourses();
```

Insert code

Referencing documents



Population

population.js

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const Author = mongoose.model('Author', new mongoose.Schema({
  name: String,
  bio: String,
  website: String
}));

const Course = mongoose.model('Course', new mongoose.Schema({
...
}));

async function createAuthor(name, bio, website) {
...
async function createCourse(name, author) {
...
}
async function listCourses() {
  const courses = await Course
    .find()
    .populate('author', 'name -_id')
    .select('name author');
  console.log(courses);
}

// createAuthor('Marizza', 'My bio', 'My Website');
// createCourse('Node Course', '6138b287f6bc38a225221f94')
listCourses();
```

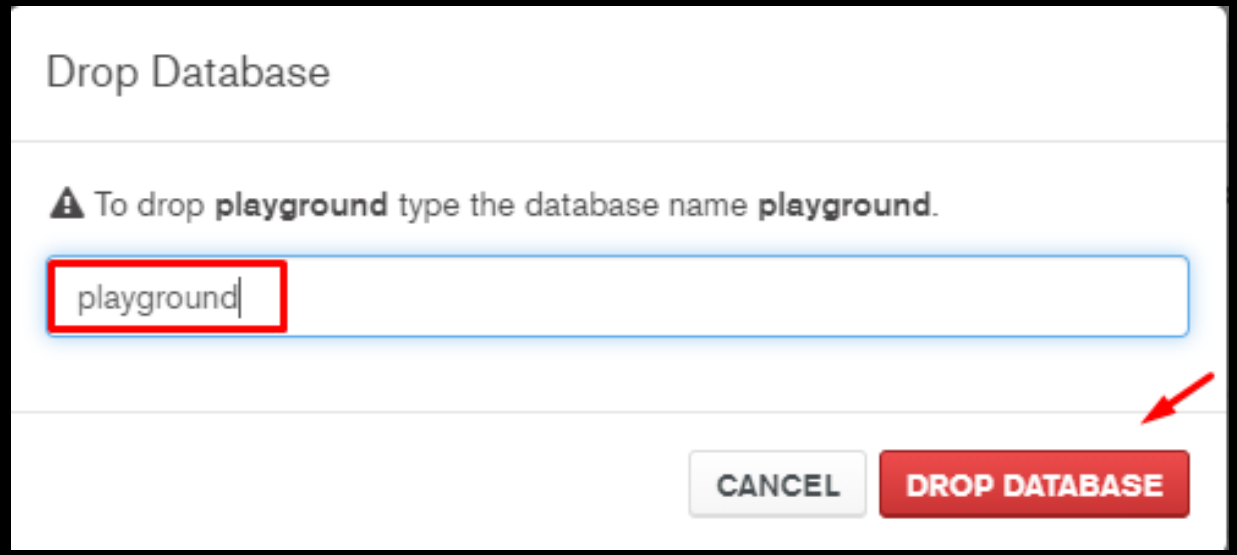
Insert code

Embedding documents

embedding.js

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});

const Author = mongoose.model('Author', authorSchema);

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  author: authorSchema
}));

async function createCourse(name, author) {
  const course = new Course({
    name,
    author
  });

  const result = await course.save();
  console.log(result);
}

async function listCourses() {
  const courses = await Course.find();
  console.log(courses);
}

createCourse('Node Course', new Author({ name: 'Marizza' }));
```

Embedding documents



Terminal

nodemon embedding.js

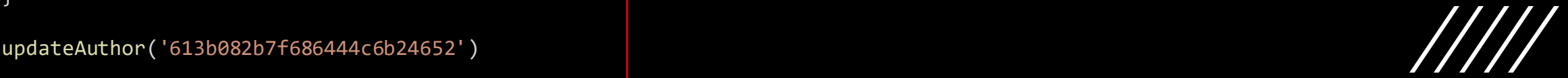
```
PROBLEMS 1  OUTPUT  TERMINAL  DEBUG CONSOLE

[nodemon] starting `node embedding.js`
Connected to MongoDB...
{
  name: 'Node Course',
  author: { name: 'Marizza', _id: new ObjectId("613b082b7f686444c6b24651") },
  _id: new ObjectId("613b082b7f686444c6b24652"),
  __v: 0
}
```

Embedding documents

embedding.js

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});

const Author = mongoose.model('Author', authorSchema);

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  author: authorSchema
}));

async function createCourse(name, author) {
...
}

async function listCourses() {
...
}

async function updateAuthor(courseId){
    const course = await Course.findById(courseId);
    course.author.name = 'Marizza Mill';
    course.save();
}

updateAuthor('613b082b7f686444c6b24652')
```

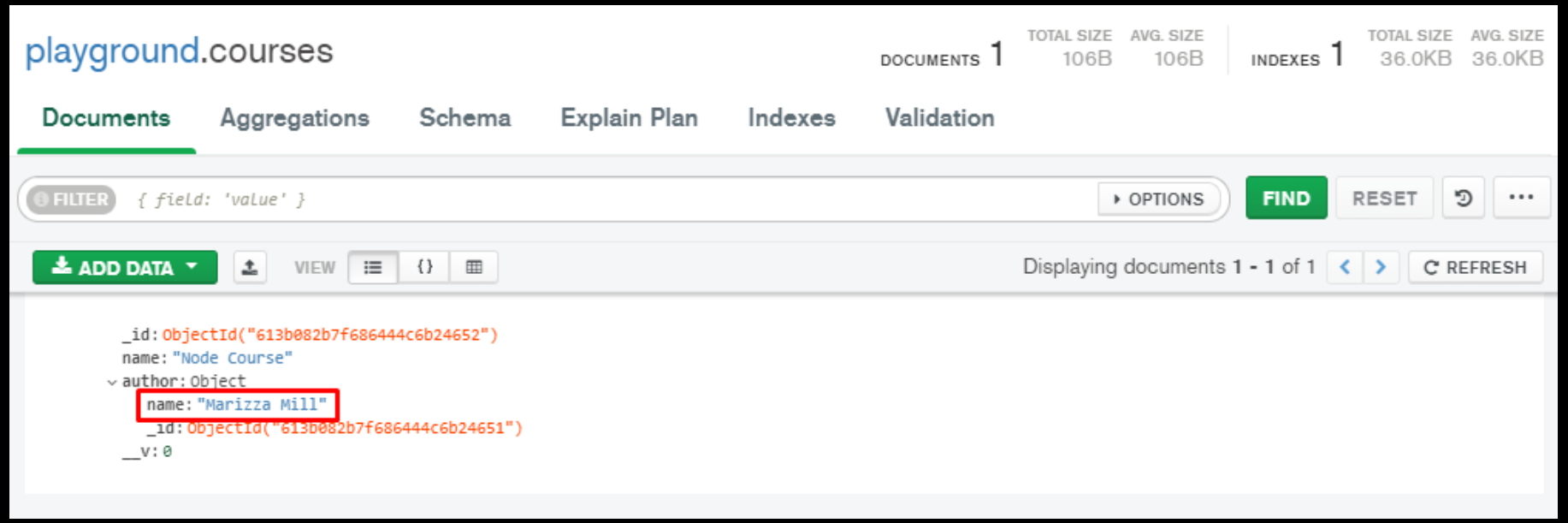
Insert code



Embedding documents

Terminal

nodemon embedding.js



Embedding documents

embedding.js

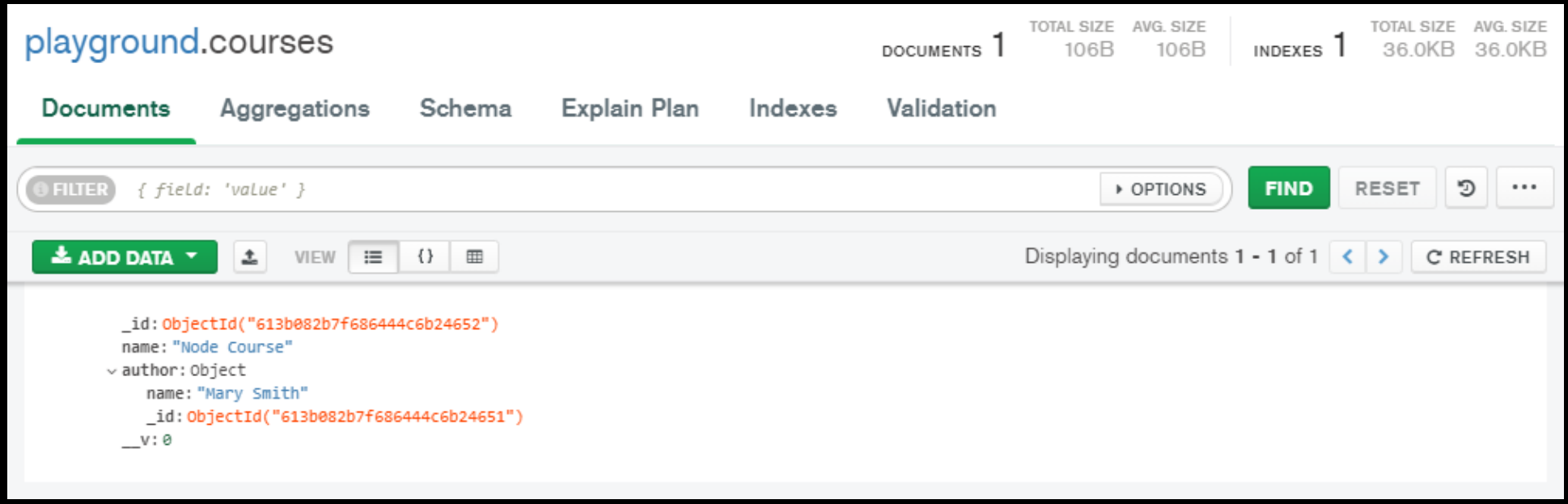
```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});

const Author = mongoose.model('Author', authorSchema);

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  author: authorSchema
}));

async function createCourse(name, author) {
...
}
async function listCourses() {
...
}
async function updateAuthor(courseId){
    const course = await Course.updateOne({_id: courseId}, {
        $set: {
            'author.name': 'Mary Smith'
        }
    });
}
updateAuthor('613b082b7f686444c6b24652')
```

Replace code

Embedding documents

Terminal

nodemon embedding.js



Embedding documents

embedding.js

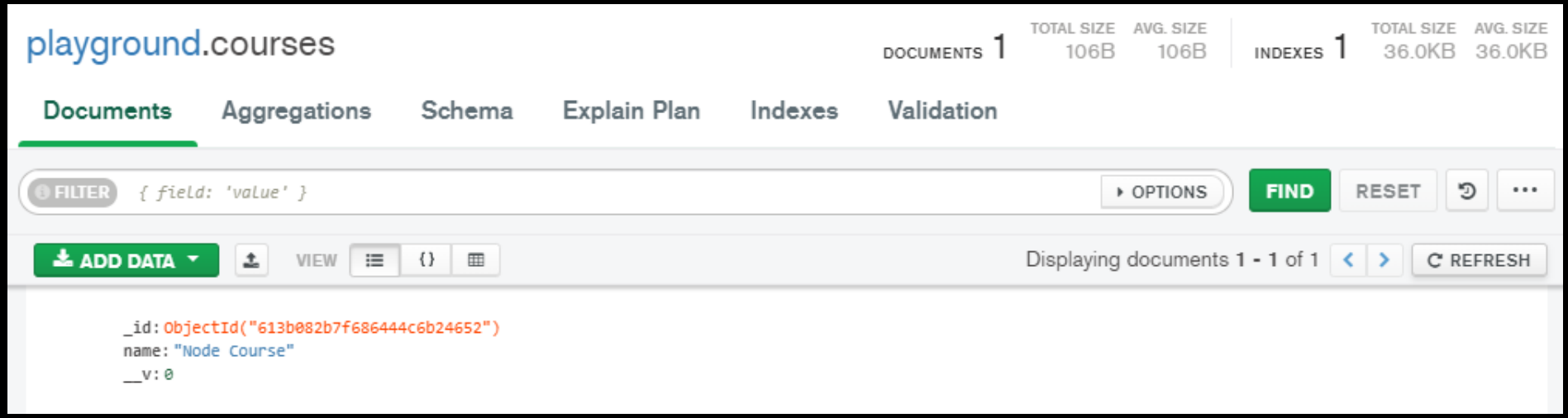
```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});

const Author = mongoose.model('Author', authorSchema);

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  author: authorSchema
}));

async function createCourse(name, author) {
...
}
async function listCourses() {
...
}
async function updateAuthor(courseId){
    const course = await Course.updateOne({_id: courseId}, {
        $unset: {
            'author': ''
        }
    });
}
updateAuthor('613b082b7f686444c6b24652')
```

Replace code

Embedding documents

Terminal

nodemon embedding.js



Using an Array of Sub-documents

embedding.js

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});
const Author = mongoose.model('Author', authorSchema);
const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  authors: [authorSchema]
}));

async function createCourse(name, authors) {
  const course = new Course({
    name,
    authors
  });
  const result = await course.save();
  console.log(result);
}
async function listCourses() {
...
}

createCourse('Angular Course', [
    new Author ({name: 'Marizza'}),
    new Author ({name: 'Alex'}),
])
```

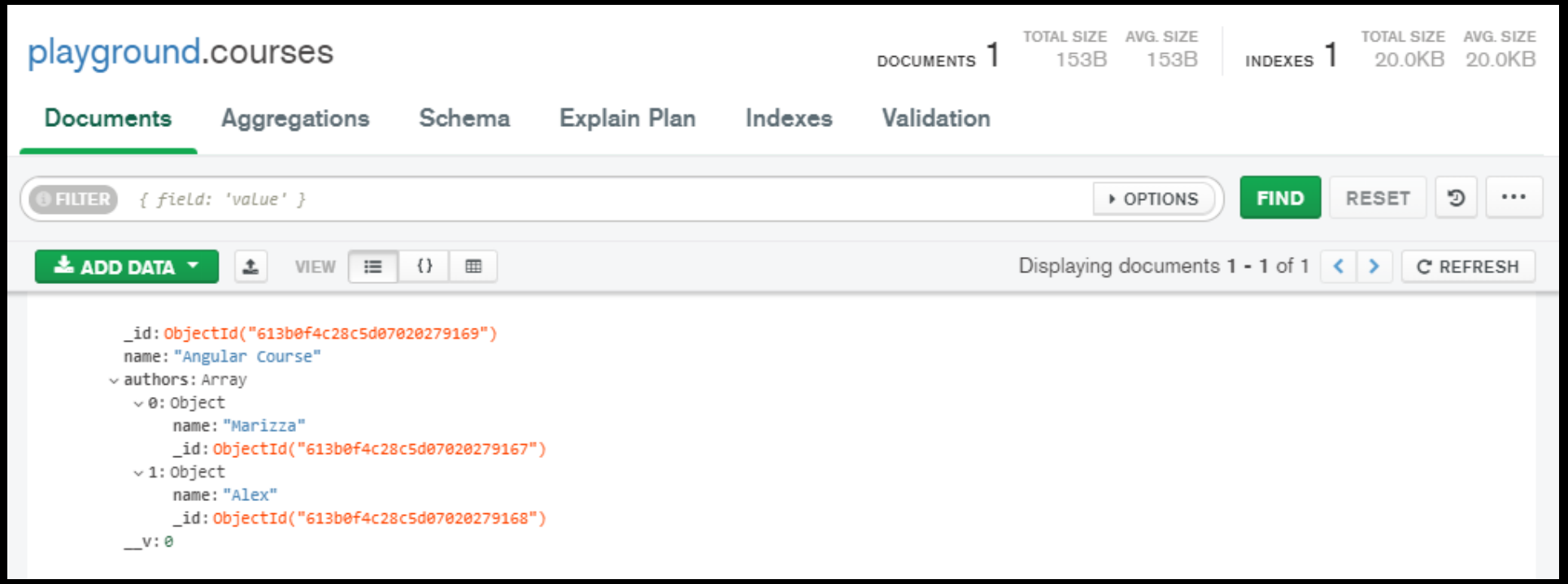
replace code

Insert code

Using an Array of Sub-documents

Terminal

nodemon embedding.js



Using an Array of Sub-documents

embedding.js

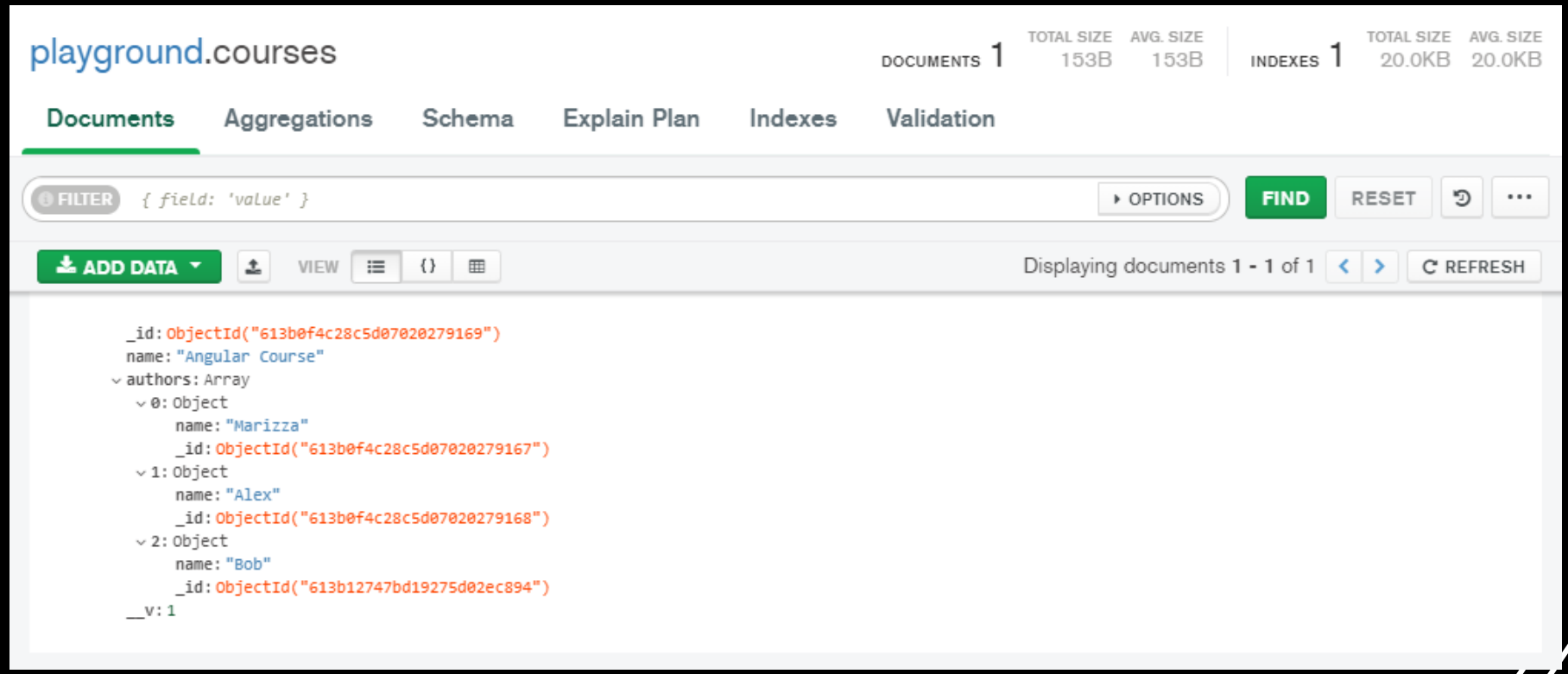
```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});

const Author = mongoose.model('Author', authorSchema);

const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  authors: [authorSchema]
}));

async function createCourse(name, authors) {
...
}
async function listCourses() {
...
}

async function addAuthor(courseId, author) {
    const course = await Course.findById(courseId);
    course.authors.push(author);
    course.save();
}

addAuthor('613b0f4c28c5d07020279169', new Author({ name: 'Bob'}));
```

Insert code

Using an Array of Sub-documents

Terminal

nodemon embedding.js



Using an Array of Sub-documents

embedding.js

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/playground')
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));

const authorSchema = new mongoose.Schema({
  name: String,
  bio: String,
  website: String
});
const Author = mongoose.model('Author', authorSchema);
const Course = mongoose.model('Course', new mongoose.Schema({
  name: String,
  authors: [authorSchema]
}));
async function createCourse(name, authors) {
...
}
async function listCourses() {
...
}
async function addAuthor(courseId, author) {
...
}
```

```
async function removeAuthor(courseId, authorId) {
    const course = await Course.findById(courseId);
    const author = course.authors.id(authorId);
    author.remove();
    course.save();
}
removeAuthor('613b0f4c28c5d07020279169', '613b12747bd19275d02ec894');
```

Insert code

Using an Array of Sub-documents

Terminal

nodemon embedding.js