M3W1D1 MongoDB Relationships

miguel.garrido@ironhack.com

1:1 Embed

javascript

Copy Download

```javascript
// Owners collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  name: "John Doe",
  email: "john@example.com",
  apartment: {  // Embedded document
    _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
    address: "123 Main St",
    purchase_date: ISODate("2020-01-15")
  }
}
```

1:1 Embed

Pros

Single query retrieves all data

Atomic updates for owner+apartment

Cons

No standalone apartment collection

Hard to query apartments independently

1:1 Reference

javascript Copy Download

```javascript
// Owners collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  name: "John Doe",
  email: "john@example.com",
  apartment_id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b")  // Reference
}

// Apartments collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
  address: "123 Main St",
  owner_id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a")  // Reference back
}
```

1:1 Reference

Pros

Clear separation of concerns

Both collections can be queried independently

Maintains referential integrity

Easy to extend if relationship becomes 1:N later

Cons

Requires two queries or $lookup to get complete data

1:M Embed

javascript Copy Download

```javascript
// Owners collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  name: "John Doe",
  email: "john@example.com",
  apartments: [  // Array of embedded documents
    {
      _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
      address: "123 Main St",
      purchase_date: ISODate("2020-01-15")
    },
    {
      _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a3c"),
      address: "456 Oak Ave",
      purchase_date: ISODate("2021-03-22")
    }
  ]
}
```

1:M Embed

Pros

Single query retrieves complete owner+apartments

Atomic writes for owner+apartments

No joins needed

Cons

Large documents if many apartments

Hard to query apartments independently

Document growth can cause performance issues

1:M Reference

javascript Copy Download

```javascript
// Owners collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  name: "John Doe",
  email: "john@example.com"
}

// Apartments collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
  address: "123 Main St",
  owner_id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a")  // Reference to owner
}
```

1:M Reference

Pros

Clean separation of concerns

No document size limitations

Easy to query apartments independently

Cons

Two queries needed for complete data

M:N - Embedding with Reference Arrays

javascript Copy Download

```javascript
// Owners collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  name: "John Doe",
  email: "john@example.com",
  apartment_ids: [
    ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
    ObjectId("5f8d8a7b2f4a1e3d6c9b8a3c")
  ]
}

// Apartments collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
  address: "123 Main St",
  owner_ids: [
    ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
    ObjectId("5f8d8a7b2f4a1e3d6c9b8a4d")
  ]
}
```

M:N - Embedding with Reference Arrays

Pros

Maintains relationship in both directions

Easy to query (find all apartments for an owner or all owners for an apartment)

Good for moderate-sized relationships

Cons

Requires maintaining both arrays for consistency

Not ideal for extremely large N:M relationships

M:N - Separate Relationship Collection

javascript Copy Download

```javascript
// Owners collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  name: "John Doe",
  email: "john@example.com"
}

// Apartments collection
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
  address: "123 Main St"
}

// Ownership collection (join table)
{
  _id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a5e"),
  owner_id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a1a"),
  apartment_id: ObjectId("5f8d8a7b2f4a1e3d6c9b8a2b"),
  ownership_percentage: 50, // Can add relationship attributes
  since: ISODate("2020-01-15")
}
```

M:N - Separate Relationship Collection

Pros

Handles very large N:M relationships well

Can store additional relationship metadata

More similar to relational approach

Easier to maintain consistency

Cons

Requires three-way joins for complete data

More complex queries