g·able

MongoDB 3.0

Data Modeling

Topics

- Data Modeling
- Data Structure
- One-to-One Model
- One-to-Many Model

Data Modeling

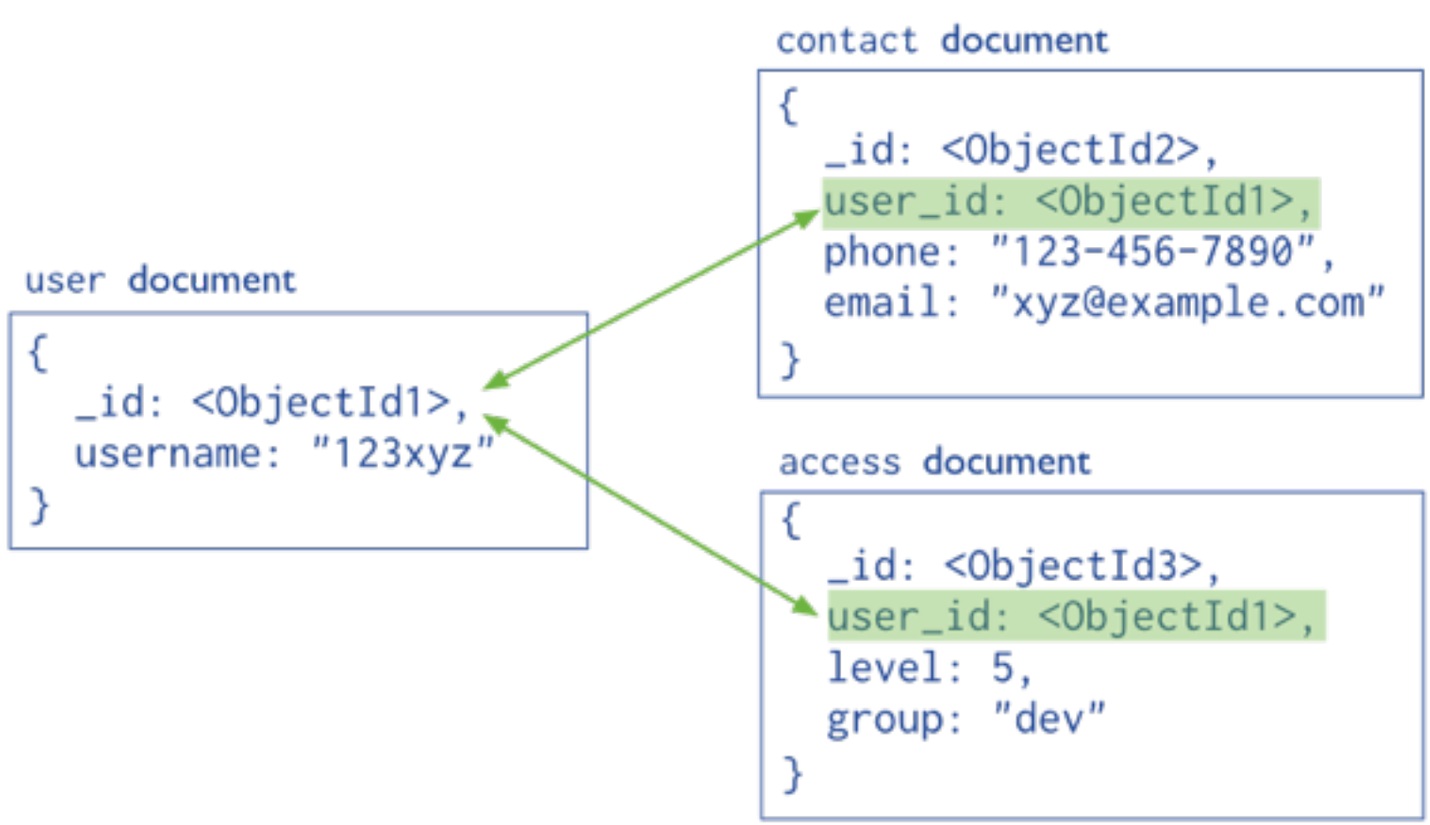
- Data in MongoDB has a *flexible* schema.
- Unlike SQL databases, where you must determine and declare a table's schema before inserting data, MongoDB's collections do not enforce document structure. This flexibility facilitates the mapping of documents to an entity or an object.
- Each document can match the data fields of the represented entity, even if the data has substantial variation.
- In practice, however, the documents in a collection share a similar structure.

Data Structure

- The key decision in designing data models for MongoDB applications revolves around the structure of documents and how the application represents relationships between data.
- There are two tools that allow applications to represent these relationships:
  - References documents.
  - Embedded documents.

References

- References store the relationships between data by including links or references from one document to another. Applications can resolve these references to access the related data. Broadly, these are normalised data models.



Embedded Data

- Embedded documents capture relationships between data by storing related data in a single document structure.
- MongoDB documents make it possible to embed document structures in a field or array within a document. These denormalised data models allow applications to retrieve and manipulate related data in a single database operation.

Atomicity of Write Operations

- In MongoDB, write operations are atomic at the document level, and no single write operation can atomically affect more than one document or more than one collection.

- A denormalized data model with embedded data combines all related data for a represented entity in a single document. This facilitates atomic write operations since a single write operation can insert or update the data for an entity. Normalizing the data would split the data across multiple collections and would require multiple write operations that are not atomic collectively.

- However, schemas that facilitate atomic writes may limit ways that applications can use the data or may limit ways to modify applications. The Atomicity Considerations documentation describes the challenge of designing a schema that balances flexibility and atomicity.

Model One-to-One Relationships with Embedded Documents

- In the normalized data model, the address document contains a reference to the patron document.

```
{
   _id: "joe",
   name: "Joe Bookreader"
}

{
   patron_id: "joe",
   street: "123 Fake Street",
   city: "Faketon",
   state: "MA",
   zip: "12345"
}
```

- If the address data is frequently retrieved with the name information, then with referencing, your application needs to issue multiple queries to resolve the reference. The better data model would be to embed the address data in the patron data, as in the following document:

```
{
   _id: "joe",
   name: "Joe Bookreader",
   address: {
              street: "123 Fake Street",
              city: "Faketon",
              state: "MA",
              zip: "12345"
            }
}
```

Model One-to-Many Relationships with Document References (1/3)

- Embedding the publisher document inside the book document would lead to repetition of the publisher data, as the following documents show:

```
{
    title: "MongoDB: The Definitive Guide",
    author: [ "Kristina Chodorow", "Mike Dirolf" ],
    published_date: ISODate("2010-09-24"),
    pages: 216,
    language: "English",
    publisher: {
                name: "O'Reilly Media",
                founded: 1980,
                location: "CA"
                }
}

{
    title: "50 Tips and Tricks for MongoDB Developer",
    author: "Kristina Chodorow",
    published_date: ISODate("2011-05-06"),
    pages: 68,
    language: "English",
    publisher: {
                name: "O'Reilly Media",
                founded: 1980,
                location: "CA"
                }
}
```

Model One-to-Many Relationships with Document References (2/3)

- To avoid repetition of the publisher data, use references and keep the publisher information in a separate collection from the book collection.

```
{
    name: "O'Reilly Media",
    founded: 1980,
    location: "CA",
    books: [12346789, 234567890, ...]
}

{
     _id: 123456789,
     title: "MongoDB: The Definitive Guide",
     author: [ "Kristina Chodorow", "Mike Dirolf" ],
     published_date: ISODate("2010-09-24"),
     pages: 216,
     language: "English"
}

{
    _id: 234567890,
    title: "50 Tips and Tricks for MongoDB Developer",
    author: "Kristina Chodorow",
    published_date: ISODate("2011-05-06"),
    pages: 68,
    language: "English"
}
```

Model One-to-Many Relationships with Document References (3/3)

- To avoid mutable, growing arrays, store the publisher reference inside the book document:

```
{
    _id: "oreilly",
    name: "O'Reilly Media",
    founded: 1980,
    location: "CA"
}

{
    _id: 123456789,
    title: "MongoDB: The Definitive Guide",
    author: [ "Kristina Chodorow", "Mike Dirolf" ],
    published_date: ISODate("2010-09-24"),
    pages: 216,
    language: "English",
    publisher_id: "oreilly"
}

{
    _id: 234567890,
    title: "50 Tips and Tricks for MongoDB Developer",
    author: "Kristina Chodorow",
    published_date: ISODate("2011-05-06"),
    pages: 68,
    language: "English",
    publisher_id: "oreilly"
}
```

References

- Data Modeling Introduction
  http://docs.mongodb.org/manual/core/data-modeling-introduction/

- Model Relationships between Documents

  - http://docs.mongodb.org/manual/tutorial/model-embedded-one-to-one-relationships-between-documents/

  - http://docs.mongodb.org/manual/tutorial/model-embedded-one-to-many-relationships-between-documents/

  - http://docs.mongodb.org/manual/tutorial/model-referenced-one-to-many-relationships-between-documents/

- Additional Resources :- Thinking in Documents (Presentation)
  http://www.mongodb.com/presentations/webinar-back-basics-1-thinking-documents?_ga=1.8564149.1693324964.1427943686