01 Esquemas Introducción

Flexible Schema

Unlike SQL databases, where you must determine and declare a table's schema before inserting data, MongoDB's collections, by default, does not require its documents to have the same schema.

That is:

- The documents in a single collection do not need to have the same set of fields and the data type for a field can differ across documents within a collection.

- To change the structure of the documents in a collection, such as add new fields, remove existing fields, or change the field values to a new type, update the documents to the new 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 document has substantial variation from other documents in the collection.

In practice, however, the documents in a collection share a similar structure, and you can enforce document validation rules for a collection during update and insert operations.

Document 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. MongoDB allows related data to be embedded within a single document.

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 denormalized data models allow applications to retrieve and manipulate related data in a single database operation.

For many use cases in MongoDB, the denormalized data model is optimal.

In general, use embedded data models when:

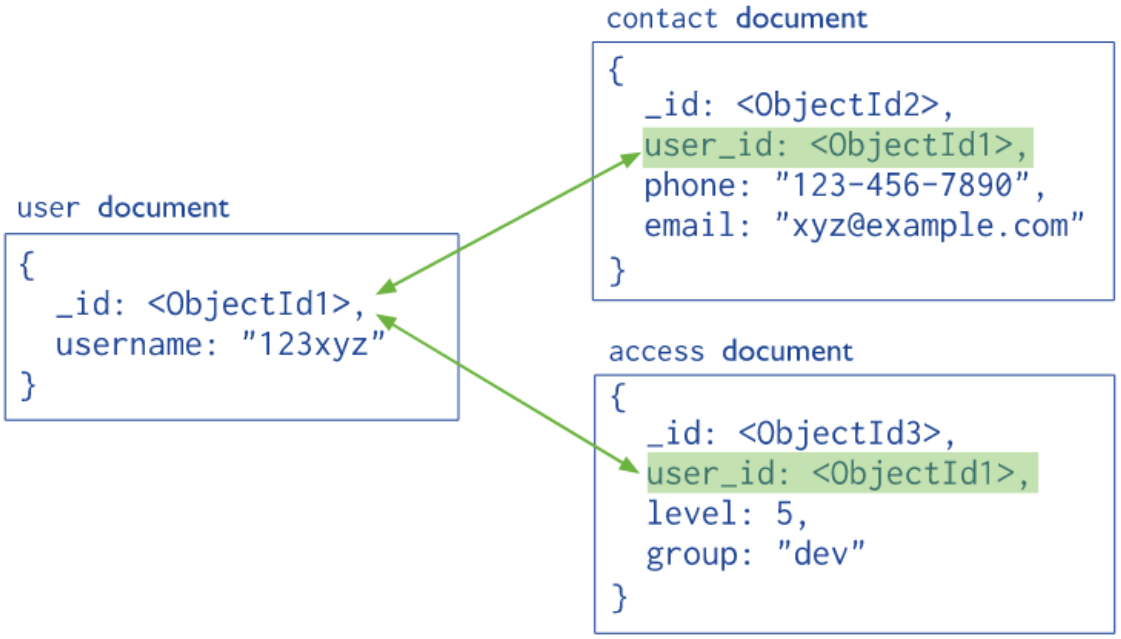
- you have “contains” relationships between entities. See Model One-to-One Relationships with Embedded Documents.

- you have one-to-many relationships between entities. In these relationships the “many” or child documents always appear with or are viewed in the context of the “one” or parent documents. See Model One-to-Many Relationships with Embedded Documents.

In general, embedding provides better performance for read operations, as well as the ability to request and retrieve related data in a single database operation. Embedded data models make it possible to update related data in a single atomic write operation.

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 normalized data models.



In general, use normalized data models:

- when embedding would result in duplication of data but would not provide sufficient read performance advantages to outweigh the implications of the duplication.
- to represent more complex many-to-many relationships.
- to model large hierarchical data sets.

To join collections, MongoDB provides the aggregation stages:

- $lookup (Available starting in MongoDB 3.2)
- $graphLookup (Available starting in MongoDB 3.4)