Mongo DB

What is NoSQL database?

NoSQL databases are non-tabular databases and store data differently than traditional row-column relational databases. NoSQL databases are designed to handle large volumes of structured, semi-structured and unstructured data. Some popular NoSQL databases are MongoDB, CouchDB, Redis, etc.

MongoDB is a document database model. It stores data in JSON-like documents. The documents are stored in collections and can contain different fields. MongoDB supports dynamic schemas, which means you can insert documents with different fields in one collection.

Some key benefits of MongoDB are:

•Schema less: As mentioned earlier, MongoDB has dynamic schema. This makes it very easy to adapt to changes. Since documents can have different fields, you can add fields without modifying the schema. This flexibility makes it suitable for unstructured data.

•Scalability: MongoDB scales horizontally by using sharding. Sharding distributes the data across multiple machines. This helps in scaling the database to handle very large data sets and high loads. MongoDB also supports auto-sharding which automatically shards data based on a shard key.

•Replication: MongoDB supports Master-Slave replication which increases redundancy and availability. Even if one machine goes down, the data remains accessible. MongoDB also has replica sets which contain multiple master and slave nodes for redundancy.

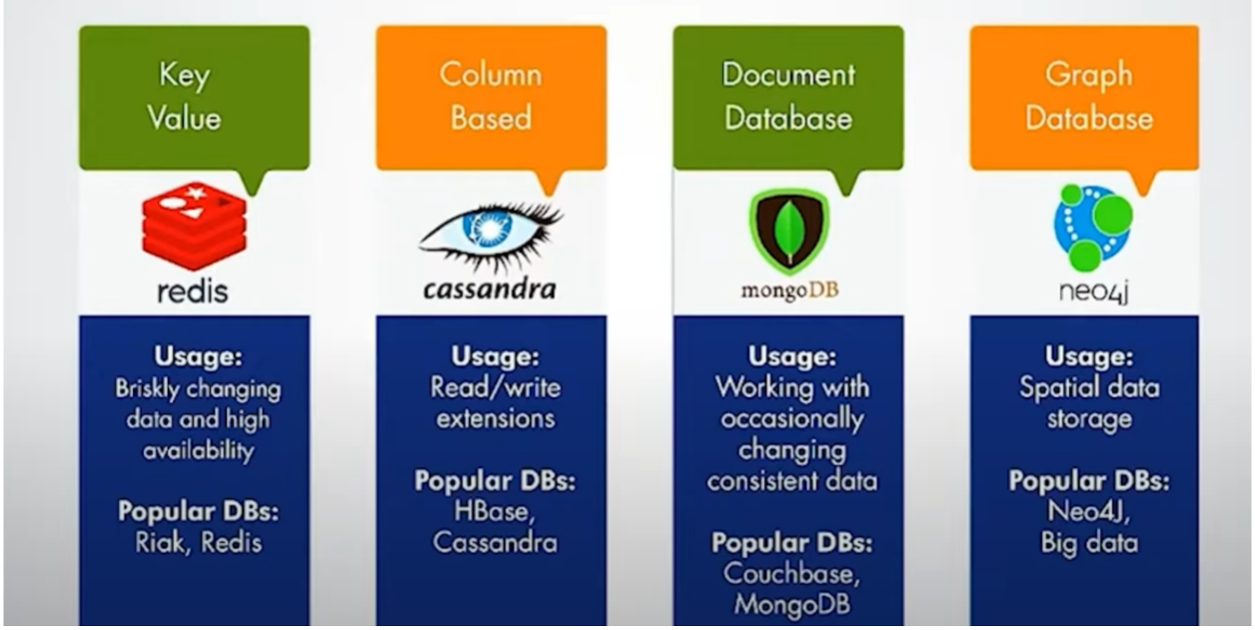
•Fast reads and writes: MongoDB provides high performance for reads and writes. The document model maps naturally to JSON-like objects in programming languages which makes it easy to work with.

•Geospatial Indexing: MongoDB has in-built geospatial indexing which is useful for location-based services. It has various geospatial query operators to query data based on location.

•Easy to use: MongoDB is very easy to learn and use. It has a simple query language

and the document model fits very naturally with object-oriented and dynamic languages. The documents are stored in JSON-like format which is easy for humans to read and write.
•ACID Transactions: MongoDB supports ACID (Atomicity, Consistency, Isolation, Durability) transactions on replica sets to ensure data integrity.
To summarize, MongoDB is a scalable, high-performance, open source, dynamic schema, document-oriented database. It is suitable for unstructured and semi-structured data and provides flexibility, scalability and high availability. It is a great choice for many modern applications.

Types of NoSQL DB



Key Value

Usage: Briskly changing data and high availability

Popular DBs: Riak, Redis

Column Based

Usage: Read/write extensions

Popular DBs: HBase, Cassandra

Document Database

Usage: Working with occasionally changing consistent data

Popular DBs: Couchbase, MongoDB

Graph Database

Usage: Spatial data storage

Popular DBs: Neo4J, Big data

Why to use MongoDB?

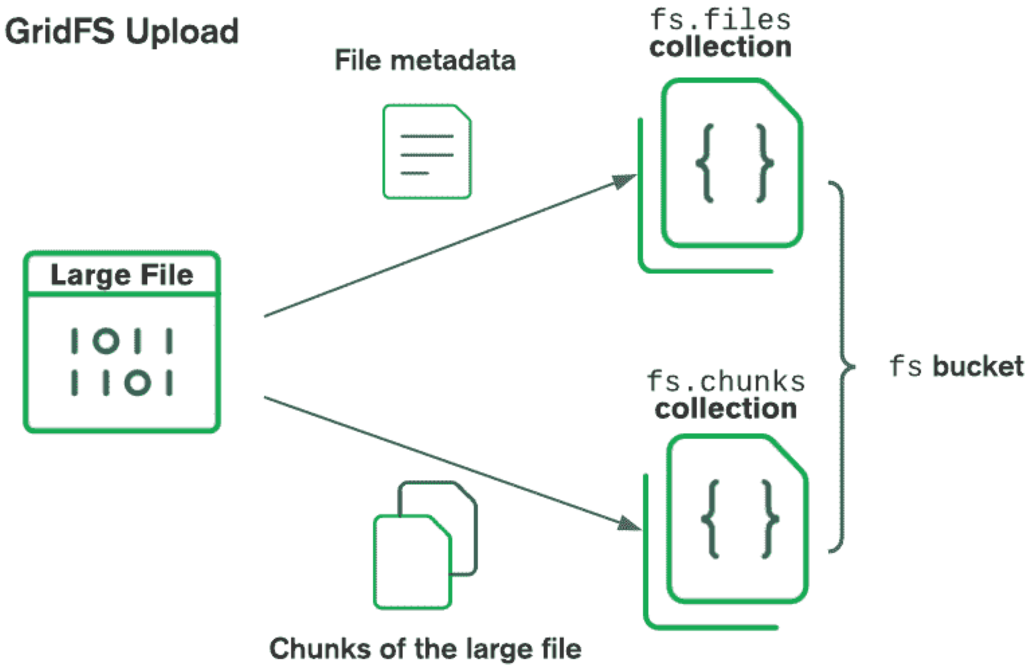
MongoDB is a popular document-oriented database management system that offers a number of advantages over traditional relational databases. Here are some reasons why you might want to use MongoDB:

1. Flexible Data Model: MongoDB has a flexible document data model that allows you to store data in JSON-like documents that can be nested and varied in structure. This makes it easier to work with data that has changing or evolving requirements.

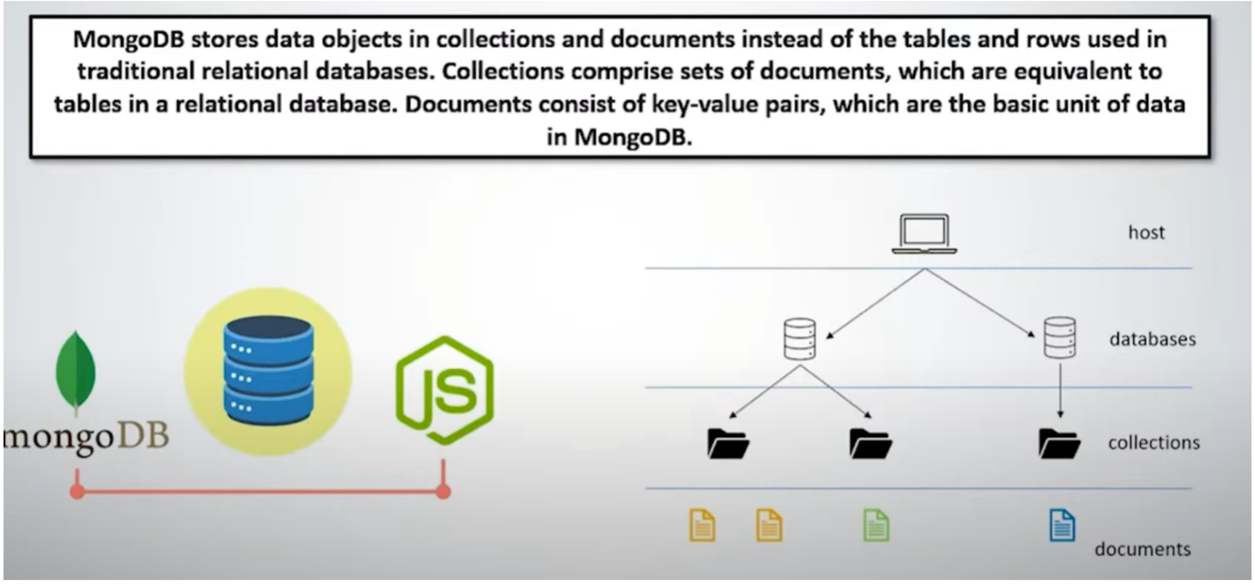
2. Scalability: MongoDB is highly scalable and can handle large volumes of data and high traffic workloads with ease. It can be scaled horizontally across multiple

servers to distribute the load and increase performance.

3. High Availability: MongoDB provides high availability through replication, allowing you to automatically maintain multiple copies of your data across multiple servers.
4. Performance: MongoDB is designed for high performance and can handle complex queries and large volumes of data with ease. It supports indexing and aggregation operations that can help improve query performance.
5. Easy to use: MongoDB is easy to use and can be deployed quickly with minimal setup and configuration. It also provides a simple and intuitive interface for managing and querying data.
6. Open Source: MongoDB is open source and free to use, which makes it an attractive option for many developers and organizations.
7. GridFS: GridFS is a file storage system in MongoDB that allows you to store and retrieve large files such as images, audio, video, and other binary data. It is a specification for storing and retrieving files that exceed the BSON-document size limit of 16 MB, which is the maximum size for a single document in MongoDB. GridFS stores files in two collections: one for metadata and another for the actual data. The metadata collection stores information about the files such as filename, content type, and file size, while the data collection stores the actual data in small chunks called "chunks". GridFS is useful for storing large files that are too big to fit into a single document. It also provides a way to stream large files and enables you to access parts of a file without loading the entire file into memory. GridFS is commonly used in web applications to store and serve files such as images, videos, and other multimedia content.

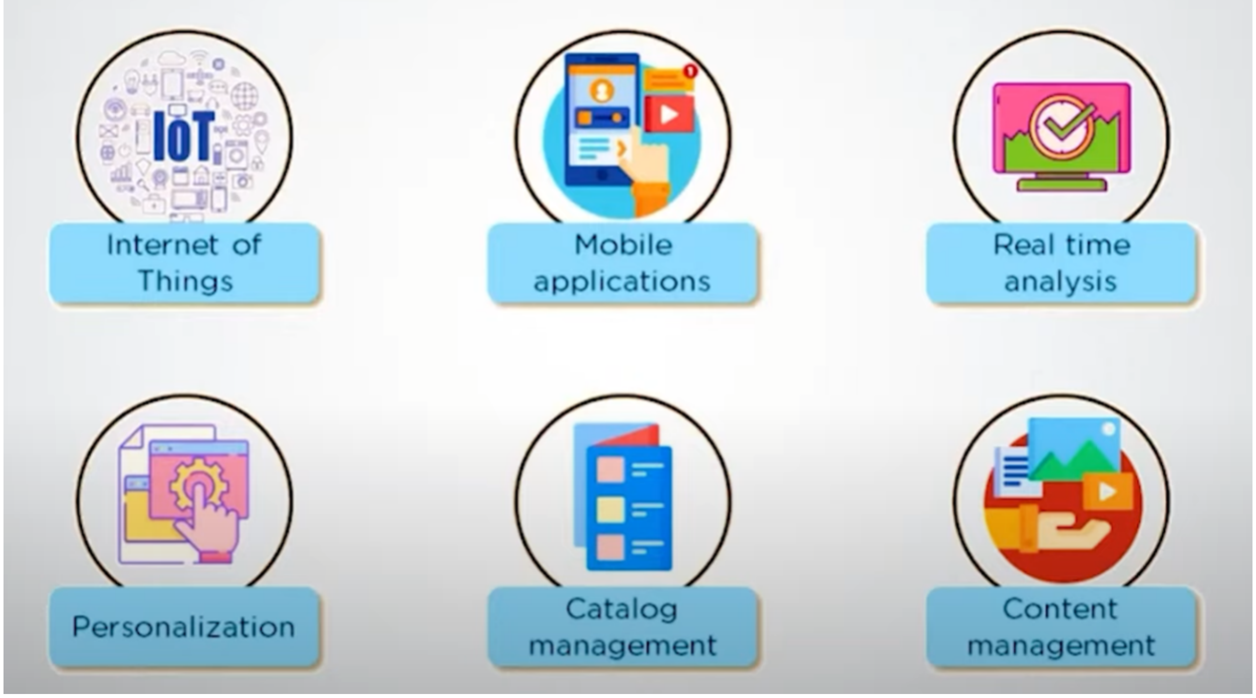


How MongoDB works

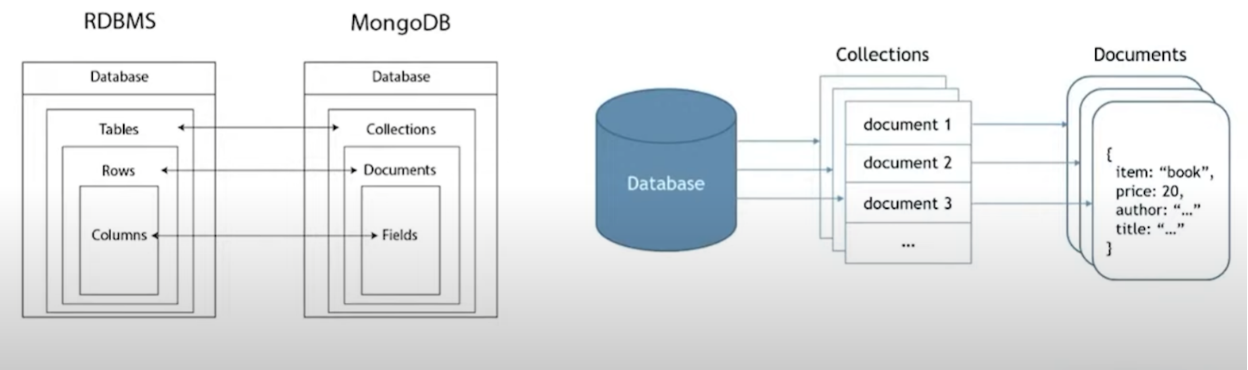


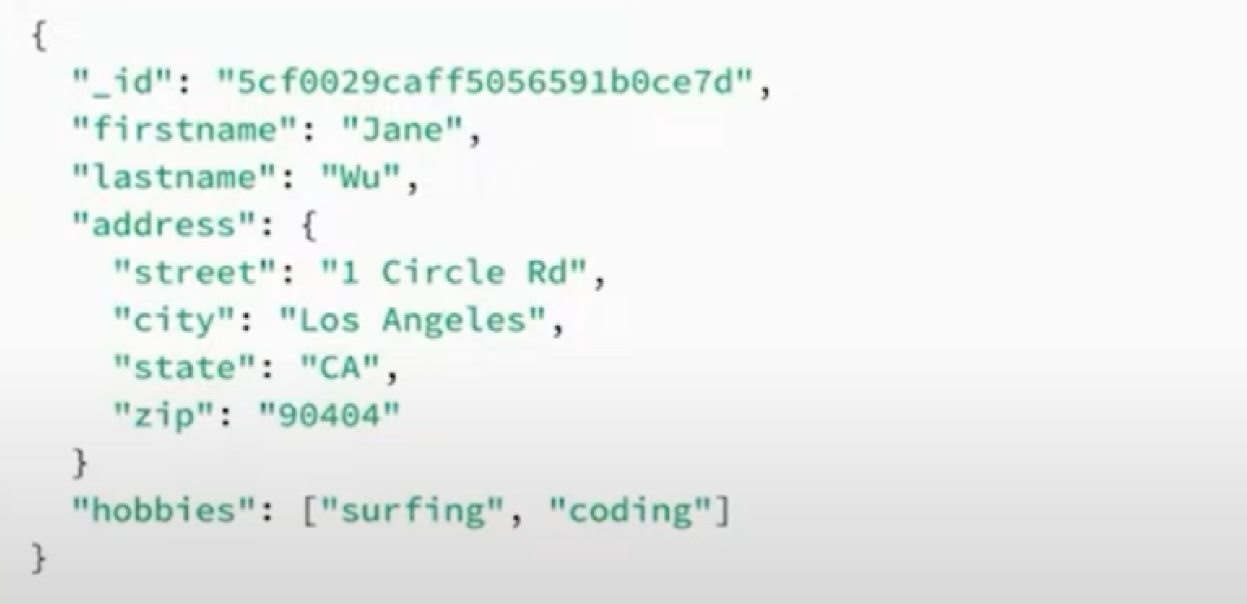
MongoDB stores data objects in collections and documents instead of the tables and rows used in traditional relational databases. Collections comprise sets of documents, which are equivalent to tables in a relational database. Documents consist of key-value pairs, which are the basic unit of data in MongoDB.

Applications of MongoDB



RDBMS mapped to MongoDB





```
{
  "_id": "5cf0029caff5056591b0ce7d",
  "firstname": "Jane",
  "lastname": "Wu",
  "address": {
    "street": "1 Circle Rd",
    "city": "Los Angeles",
    "state": "CA",
    "zip": "90404"
  }
  "hobbies": ["surfing", "coding"]
}
```

Downloading MongoDB

Download Link

While installed go to location “C:\Program Files\MongoDB\Server\6.0\bin” and add the path to “User Variable”

Data Modeling

In MongoDB, data modeling is the process of designing the structure of the database to efficiently and effectively store and retrieve data. This involves defining the document schema, selecting appropriate data types, and identifying relationships between documents.

The data model in MongoDB is based on a flexible document structure, which means that the schema is not predefined, and documents in the same collection can have different structures. This makes it easy to store and retrieve complex data structures.

When designing a data model in MongoDB, it is important to consider the following:

1. Data Access Patterns: It is important to identify the most common queries and operations that will be performed on the data to design a data model that can efficiently handle those operations.
2. Relationships: Identify the relationships between documents and how they will be queried to optimize the schema.

3. Indexing: Proper indexing can greatly improve query performance, so it is important to identify the fields that will be queried most frequently and create indexes on them.
4. Data Growth: Consider the expected growth of the data to design a schema that can handle a large volume of data over time.

Data modeling has several advantages, including:

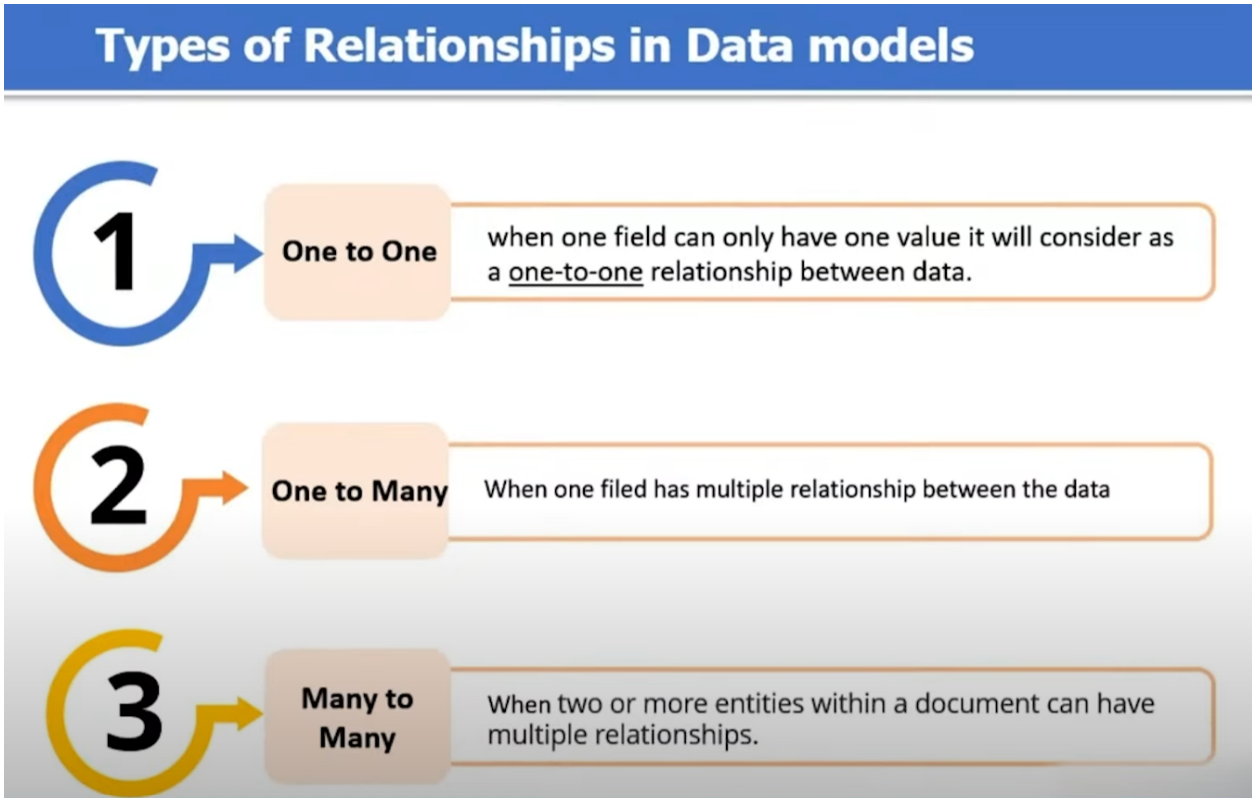
1. Improved Data Quality: Data modeling helps to identify and eliminate inconsistencies and errors in the data, resulting in improved data quality. By defining the structure of the data, data modeling ensures that the data is accurate, complete, and consistent.
2. Better Data Integration: Data modeling helps to integrate data from different sources and systems. By defining the structure of the data, data modeling ensures that data from different sources is compatible and can be integrated seamlessly.
3. Improved Decision Making: Data modeling helps to organize and structure data in a way that makes it easy to analyze and make decisions based on the data. By creating a clear and logical representation of the data, data modeling enables users to understand and interpret the data more easily.
4. Improved System Performance: By optimizing the database schema and improving query performance, data modeling can improve system performance and reduce the time required to retrieve and process data.
5. Scalability: Data modeling enables the database to be scaled up or down as required to handle changing data volumes and user loads. By designing a data model that can accommodate future growth, data modeling ensures that the database can scale up as the data volume increases.

How data modeling works in MongoDB?

Unlike in RDBMS where we have to define the schema before hand, MongoDB documents need not require to have to same schema. Documents in the single collection do not need to have the same set of fields and the datatype of a field can differ across documents within a collection. To change the structure of the document in a collection such as adding a new field or removing the existing field in the document.

Based on the level of detail and specificity data models are categorized into three

- Conceptual Data Model → It explains what the system should with regard to and how it is related. This model is usually built with the help of the user and the stakeholders. It represents the business logic and is often used as the basis for the following model.
- Logical Data Model → It model describes how the data will be structured. In this model the relationship between the entities are established at a high level.
- Physical Data Model → It represents how the data will be stored in a specific database.

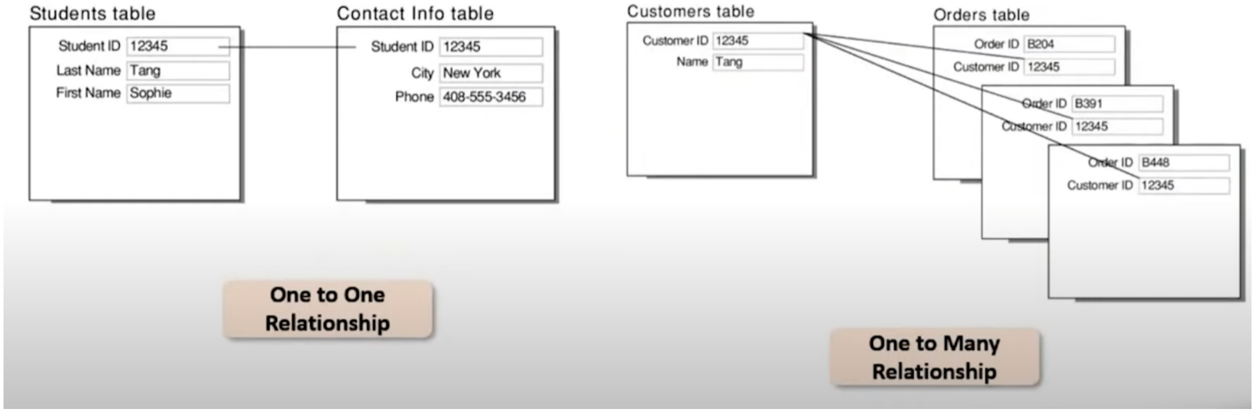


Types of Relationships in Data models

1 One to One — when one field can only have one value it will consider as a one-to-one relationship between data.

2 One to Many — When one filed has multiple relationship between the data

3 Many to Many — When two or more entities within a document can have multiple relationships.



Students table

Student ID 12345
Last Name Tang
First Name Sophie

Contact Info table

Student ID 12345
City New York
Phone 408-555-3456

Customers table

Customer ID 12345
Name Tang

Orders table

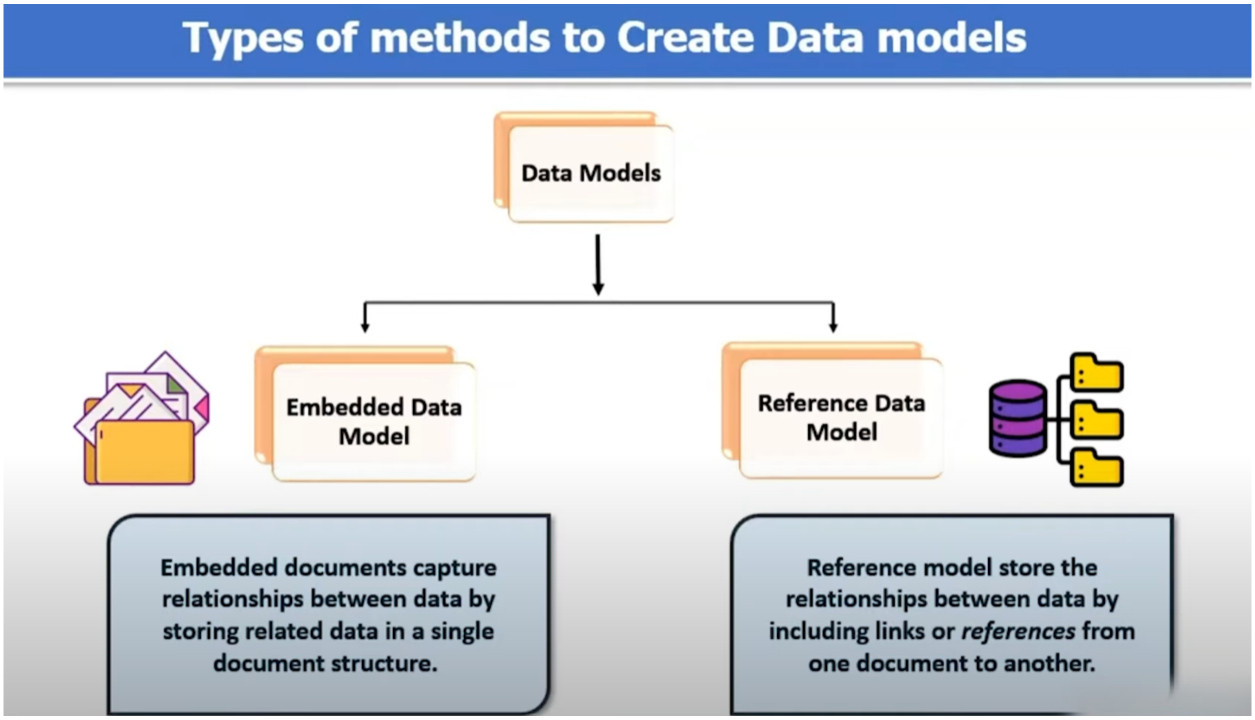
Order ID B204
Customer ID 12345

Order ID B391
Customer ID 12345

Order ID B448
Customer ID 12345

One to One Relationship

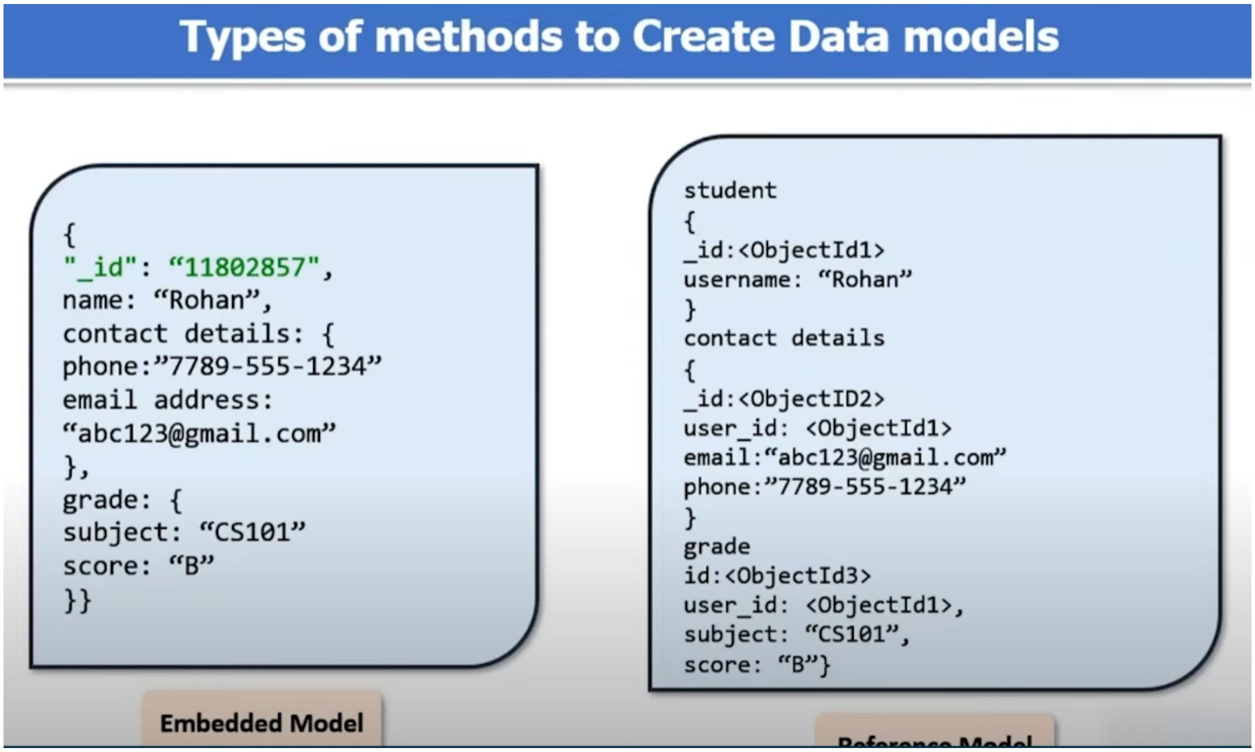
One to Many Relationship



Types of methods to Create Data models

Embedded documents capture relationships between data by storing related data in a single document structure.

Reference model store the relationships between data by including links or *references* from one document to another.



Types of methods to Create Data models

```
{
"_id": "11802857",
name: "Rohan",
contact details: {
phone:"7789-555-1234"
email address:
"abc123@gmail.com"
},
grade: {
subject: "CS101"
score: "B"
}}
```

Embedded Model

```
student
{
_id:<ObjectId1>
username: "Rohan"
}
contact details
{
_id:<ObjectID2>
user_id: <ObjectId1>
email:"abc123@gmail.com"
phone:"7789-555-1234"
}
grade
id:<ObjectId3>
user_id: <ObjectId1>,
subject: "CS101",
score: "B"}
```

Reference Model