MongoDB
Data Modeling

Team Emertxe

Data Modeling

Data Model

Data in mongoDB has a flexible schema in a collection.

MongoDB does not need any predefined schema.

Every document in a collection could have different data.

ΣMERTXE

Scheme Free

```
{ name : “Smith” ,
  Id      : 01 ,
  Course : “Java” ,
  City     : “Banglore”,
  State    : “Karnataka” }
```

```
{ name : “Mac” ,
  Id      : 02 ,
  Course : “HTML” ,
  State    :  “UP” }
```

```
{ name : “Allen” ,
  Id      : 03 ,
  City    : “Mumbai” }
```

```
{ name : “John” ,
  Id      : 04
  }
```

The above example in one collection four documents of different structure.

ΣMERTXE

Design
Data Model

There are two tools to represents relationship between data.

- References
- Embedded Documents

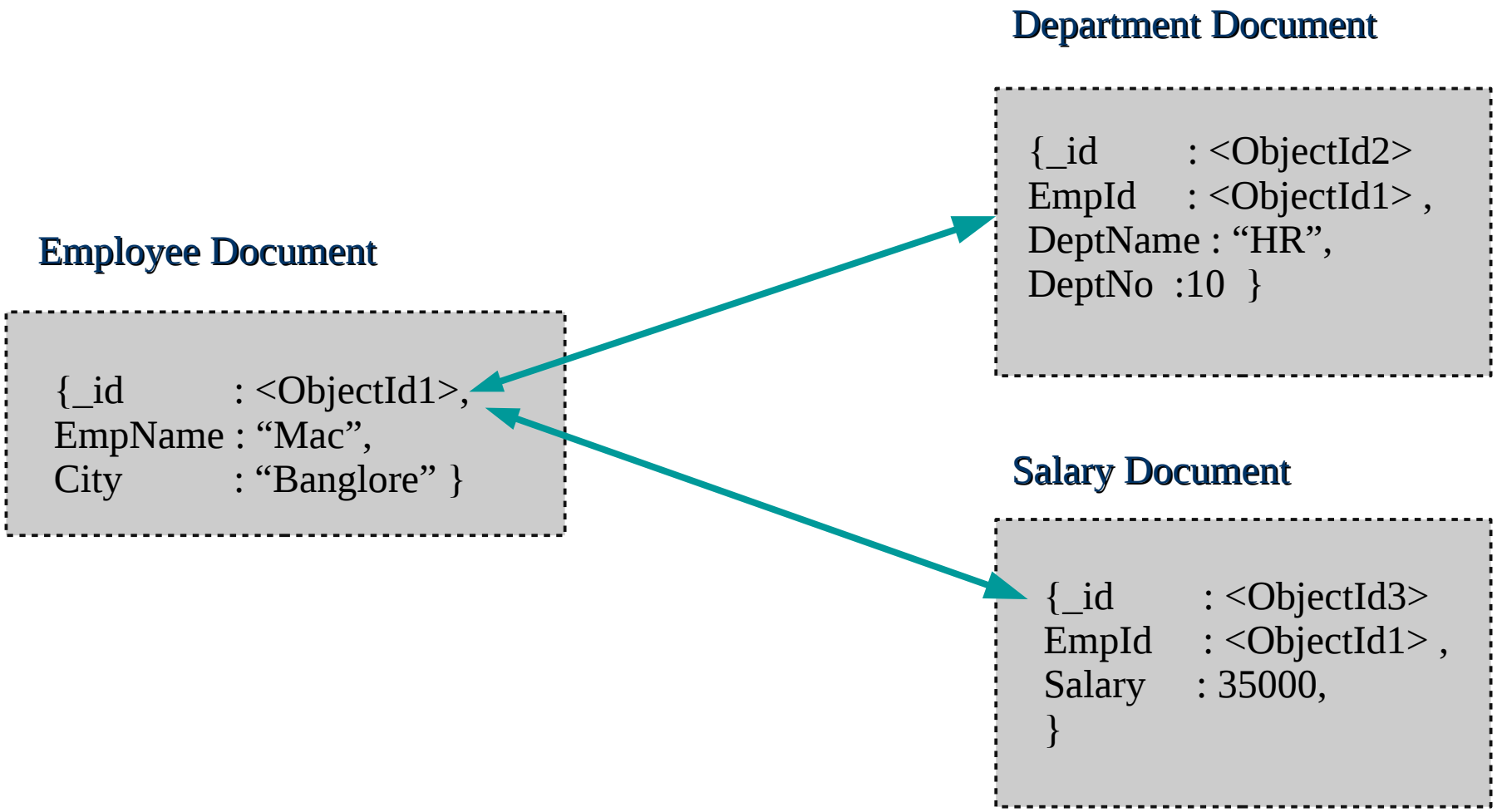
ΣMERTXE

References

- MongoDB does not support Joins.
- References store the relationships between data by including link or references to relate one document to another .
- These Schemas are normalized data.

ΣMERTXE

References
(Example)



EMERTXE

Embedded Documents

In MongoDB related data can be embedded in a single structure or document (like subset).

These Schemas are known as “denormalized” model.

ΣMERTXE

Embedded documents
Example

```
{
  _id          : <ObjectId1> ,
EmpName    : “Mac” ,
City           : “Banglore”,
Department  :  {  DeptName     : “HR” ,
                  DeptNo         :  10   } ,
Salary     :  {   Salary          : 350000 }


}
```

Embedded Sub documents

Embedded Sub documents

EMERTXE

Document Validation

Document Validation

MongoDB provides the capability to validate documents during updates and insertions.

Validation rules are specified using validator option.

The collMod command is used to add validation document to an existing collection with the validator option .

MongoDB handles existing documents using the validationLevel option.

By default validation level is strict and validation rule is applied to all inserts and update .

The validation level is moderate then validation rule is applied to inserts and updates to existing document that fulfill the validation criteria.

ΣMERTXE

Document validation
(Example)

```
> use db1
switched to db db1
> db.createCollection("order");
{ "ok" : 1 }
> db.order.insert( { _id : 1 , item : "mobile" , price : 15000 , qty :2 });
WriteResult({ "nInserted" : 1 })
> db.order.insert( { _id : 2 , item : "camera" , price : 8000 , qty :3 });
WriteResult({ "nInserted" : 1 })
```

ΣMERTXE

Document validation
(Example)

```
> db.runCommand( { collMod : "order" , validator : { $or :
 [ { price : { $exists :true } } ,
 { qty : { $exists :true }} ]
 validationLevel : "moderate" });
```

EMERTXE

Data Modeling Concepts

Operational factors and Data Model

There are some operational factors that arises outside of the application but impact the performance of MongoDB itself.

- Document growth
- Atomicity
- Sharding
- Indexes

Document Growth

The document size can be increased due to some updates to documents.

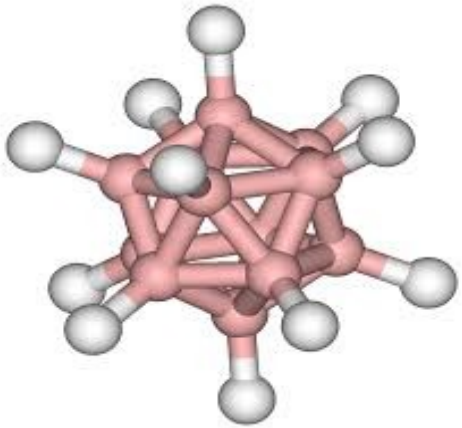
These updates includes adding new fields to a document and pushing elements to array.

ΣMERTXE

Atomicity

MongoDB write operations (db.collection.update() and db.collection.remove()) are atomic at the level of a single document .

Single write operation can change more than one document.



Sharding

- MongoDB uses sharding to provide horizontal scaling.
- Sharding is the method for distributing data across multiple machines.
- MongoDb uses a shard key to distribute data .

ΣMERTXE

Indexes

- MongoDB uses indexes to improve performances for common queries.
- MongoDB automatically creates a unique index on the _id field.
- Each index requires at least 8kb of data space.

ΣMERTXE

Indexes

- MongoDB uses indexes to improve performances for common queries.
- MongoDB automatically creates a unique index on the _id field.
- Each index requires at least 8kb of data space.

ΣMERTXE

Model Relationship between documents

Relationships in MongoDB

Relationship in MongoDB are used to relate one or more documents.

The relationship can be of following forms :

- One to One .
- One to Many.
- Many to Many.

ΣMERTXE

One to One relationships with embedded documents

```
> db.student.insert( { _id : 01 ,
                       name : "peter" ,
                      course :
                {           courseId : "B.E" ,
                            duration   : “4 year” ,
                                  }
                             } );
```

The above example student database of a college , where the student can enroll in only one course .

ΣMERTXE

Exercise

- Write a MongoDB query to create a collection name person(field names are : Name ,Address ,Phone Number) , Embed the person collection document into bank document (field name : Bank Name , Account Number , Address).

One to Many relationships with embedded documents

```
> db.student.insert ( {          _id : 01 ,
                          name : "Peter" ,
            address :  [ { street  : "Mg Road" ,
                           city    : "Blr  " ,
                           state   : "Karnataka" } ,
                         {  street  : "Brigade Road" ,
                           city    : "Blr" ,
                           state   : "Karnatka" } ]
                         } );
```

The above example student have two address one current and one permanent .

ΣMERTXE

Exercise

- Write a MongoDB query to to create collection customer and customer have account in two different banks using one to many relationship with embedded documents.

One to Many relationships with document references

```
(  { _id : 1 ,
state :"karnataka" ,
address_ids :
[ ObjectId("58c62ffa97a24d8a5d676495") ,
ObjectId("58d241da0f338ce9cbaa6fad")
] } )
```

EMERTXE

One to Many relationships with document references

```
> var res = db.Student.find( { name : "john" },
{address_ids :1 });
> var adds =db.StudAddress.find( { _id :
{ "$in" : res [ "address_ids"] }})
> print (adds)
```

ΣMERTXE

References

- https://www.wikimedia.org/
- https://docs.mongodb.com/manual/

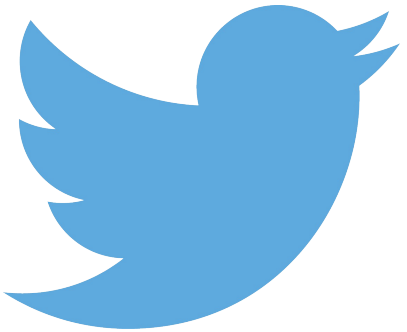
EMERTXE

Stay connected

About us: Emertxe is India's one of the top IT finishing schools & self learning kits provider. Our primary focus is on Embedded with diversification focus on Java, Oracle and Android areas

Emertxe Information Technologies,
No-1, 9th Cross, 5th Main,
Jayamahal Extension,
Bangalore, Karnataka 560046
T: +91 80 6562 9666
E: training@emertxe.com

https://www.facebook.com/Emertxe



https://twitter.com/EmertxeTweet

https://www.slideshare.net/EmertxeSlides

EMERTXE

Thank You