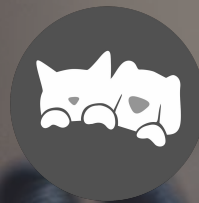


MongoDB





Karl MARQUES BERNARDO

CTO Slyvent
CTO Vetixy

kmarques@vetixy.com
[ESGI] [NODE] [4IW*]

<https://github.com/kmarques>



HISTORIQUE

3

4

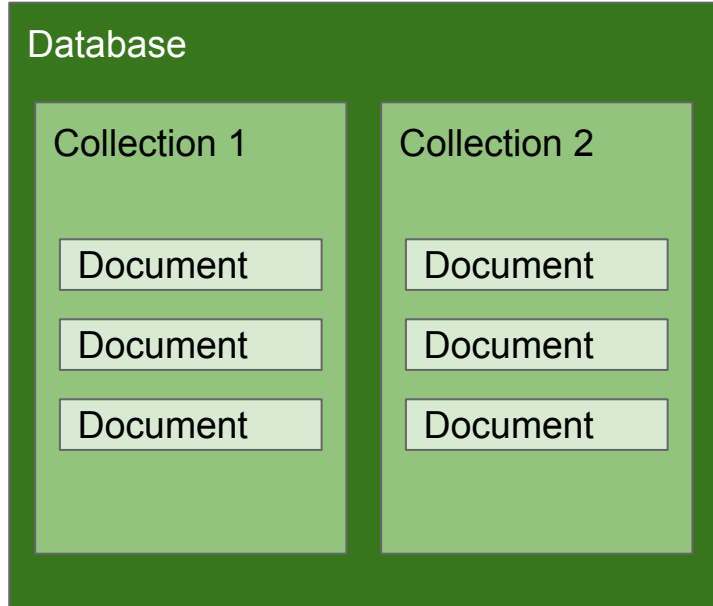
- Créateur : 10gen (MongoDB Inc en 2013)
- Date de début: 2007
- Licence: SSPL
- Dernière version: 4.2
- Type de base: NoSQL
- Type de données: Document JSON
- Language de requête: Javascript



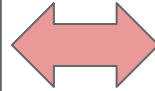
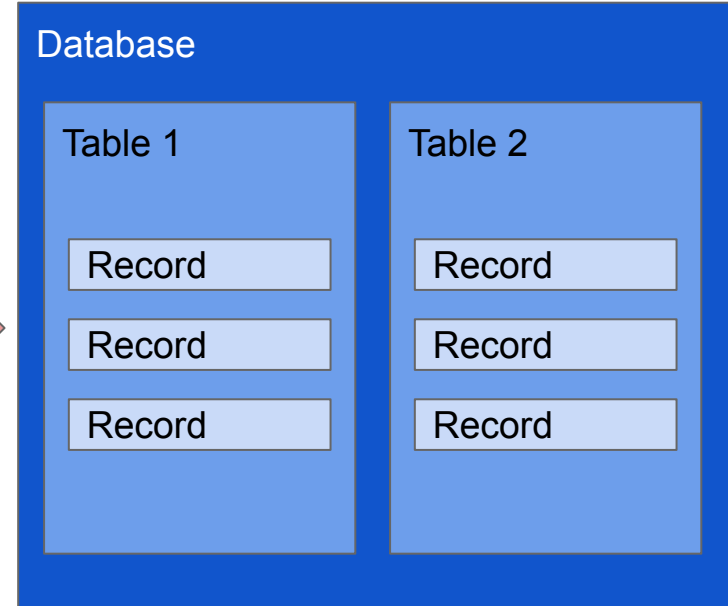
Schématisation

5

MongoDB



SQL Database



6

7

Principes / Avantages

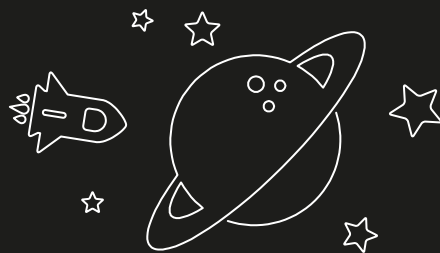
Pas de schéma

Pas de relations entre les documents

Le document correspond à la donnée applicative

Indexation FullText

Hyper-scalable



INSTALLATION

8

1) Télécharger le docker-compose à l'adresse suivante

<https://github.com/kmarques/esgi-cours/blob/master/node/docker-compose.yml>

2) Personnaliser le docker-compose

Ports du service mongo

Credentials du service mongo

3) Télécharger MongoDB Compass à l'adresse suivante

<https://www.mongodb.com/download-center/compass>

4) Lancer les dockers

`docker-compose up -d`

5) Vérifier la connectivité

Lancer l'application MongoDB Compass et insérer le nouveau node



CRUD

10

11

Création

Database

use DB_NAME => Sélectionne la db
Création automatique à la première insertion

Collection

db.createCollection("COLNAME",
{OPTIONS})
Création manuelle

Document

db.COLNAME.insert({DOCUMENT});
Création manuelle
Si collection inconnue, création automatique

12

Suppression

Database

`db.dropDatabase()`

Collection

`db.COLNAME.drop()`

Document

`db.COLNAME.remove([CRITERIA]);`

CRITERIA: Objet de sélection

13

Modification

Document

db.COLNAME.update({CRITERIA},
{NEW_DATA}, multi);

Met à jour le.s document.s selon critères

CRITERIA: Objet de sélection

NEW_DATA: Objet contenant les
nouvelles données

```
{  
  $set: NEW_DATA  
}
```

Multi: Booléen désigne une
modification multiple (défaut 1
document modifié)

db.COLNAME.save(IID, ...NEW_DATA));

Remplace le document désigné par l'ID

14

Sélection

Document

db.COLNAME.find({CRITERIA});

Recherche un ensemble de documents

db.COLNAME.findOne({CRITERIA});

Recherche le premier document
correspondant

CRITERIA: Objet de sélection

15

Critère de recherches 1/2

Combinaison

AND : entrée supplémentaire dans l'objet de sélection

OR : Utilisation de la clé **\$or** dont la valeur est un tableau

{ \$or: [{ CRITERIA1 }, { CRITERIA2 }] }

Numérique/Date

LESS/GREATER THAN

{ key: { \$lt/\$gt: NUMBER } }

LESS/GREATER THAN EQUALS

{ key: { \$lte/\$gte: NUMBER } }

NOT EQUALS

{ key: { \$ne: NUMBER } }

16

Critère de recherches 2/2

Texte

EQUALS

{ key: value }

REGEXP

{ key: /myregexp/ }

{ key: { \$regex: "myregexp" } }

TEXT SEARCH

{ \$text: { \$search: "my text" } }

Recherche sur tous les index de type

FullText

Pagination

LIMIT

```
db.COLNAME.find([CRITERIA])  
  .limit(NUMBER)
```

OFFSET

```
db.COLNAME.find([CRITERIA])  
  .limit(NUMBER)  
  .skip(NUMBER)
```

18

Sort / Filtres

SORT

db.COLNAME.find([CRITERIA])

.sort([{ key: 1, key2: -1}])

1: Asc -1: Desc

Filtres

db.COLNAME.find([CRITERIA], [FILTER])

FILTER: {key: 1, key2: 0}

1: Affiché 0: Masqué

MongoDB

```
db.users.find({  
  name: /jean/i,  
  dob: { $gt: new Date("2001-01-01") }  
}, {  
  name: 1, address: 0, dob: 1  
})  
.sort({  
  name: 1,  
  dob: -1  
})  
.limit(10)  
.skip(10)
```

SQL

```
SELECT  
  name,  
  dob  
FROM users  
WHERE  
  name ILIKE "%jean%"  
AND  
  dob > "2001-01-01"::DATE  
ORDER BY name ASC, dob DESC  
LIMIT 10  
OFFSET 10
```

Collections: Sakila_films

- 1) Rechercher tous les films avec l'acteur ED CHASE
- 2) Rechercher tous les films dont la description comprend "documentary" et de category "horror"
- 3) Donner le nombre de films en rating "G"

Collections: video_movieDetails

- 4) Rechercher tous les films de 2013 ou 2012 dont la durée est entre 60 et 150 minutes
- 5) Rechercher tous les films qui ont une image certified sur tomato

Collections: Sakila_films

- 6) Afficher tous les rated ("PG", "PG-13", ...) et le nombre de films pour chacun