



MASTER
INFORMATIQUE

Conversion du MCD MERISE en des objets MongoDB

Membres

Fayssal EL ANSARI
Jonathan DI MARTINO

15/06/2023

1. Spécification des modèles de documents à mettre dans chaque collection

Appartement : Cette entité serait un document avec les attributs suivants : `_id`, `price`, `location`, `numberOfRooms`, `numberOfBathrooms`, `_class`.

Exemple de document en JSON :

```
json Copy code

{
  "_id": "app1",
  "_class": "Appartement",
  "price": 500,
  "location": "Paris",
  "numberOfRooms": 3,
  "numberOfBathrooms": 2
}
```

Clients : Cette entité serait un document avec les attributs suivants : `_id`, `email`, `password`, `phoneNumber`, `countryOfOrigin`, `isVerified`, `_class`.

Exemple de document en JSON :

```
json Copy code

{
  "_id": "client1",
  "_class": "Client",
  "email": "john.doe@example.com",
  "password": "securePassword",
  "phoneNumber": "0123456789",
  "countryOfOrigin": "France",
  "isVerified": true
}
```

Reservation : Cette entité serait un document avec les attributs suivants : `_id`, `apartmentId`, `userId`, `startDate`, `endDate`, `price`, `_class`.

Exemple de document en JSON :

```
json Copy code

{
  "_id": "res1",
  "_class": "Reservation",
  "apartmentId": "app1",
  "userId": "client1",
  "startDate": "2023-06-15",
  "endDate": "2023-06-20",
  "price": 500
}
```

Transaction : Cette entité serait un document avec les attributs suivants : `_id`, `clientId`, `reservationId`, `amount`, `transactionType`, `transactionStatus`, `_class`.

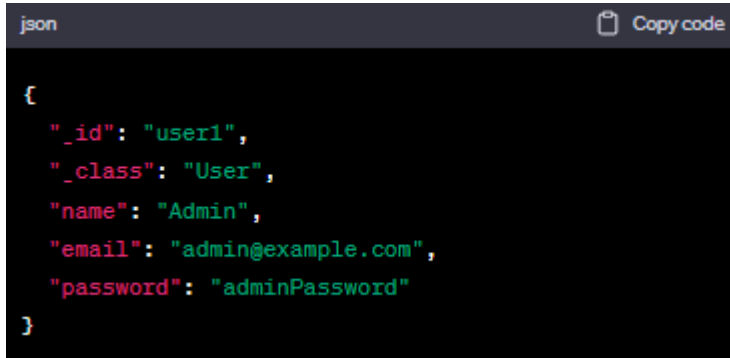
Exemple de document en JSON :

```
json Copy code

{
  "_id": "trans1",
  "_class": "Transaction",
  "clientId": "client1",
  "reservationId": "res1",
  "amount": 500,
  "transactionType": "debit",
  "transactionStatus": "completed"
}
```

Utilisateur : Cette entité serait un document avec les attributs suivants : `_id`, `name`, `email`, `password`, `_class`.

Exemple de document en JSON :



```
json Copy code
{
  "_id": "user1",
  "_class": "User",
  "name": "Admin",
  "email": "admin@example.com",
  "password": "adminPassword"
}
```

2. Spécification des classes et des méthodes JAVA

Chaque classe Java correspondante aurait les méthodes CRUD suivantes :

`create()` : pour insérer un nouveau document dans la base de données.

`creates()` : pour insérer un ou plusieurs nouveau documents dans la base de données.

`get()` : pour lire un document à partir de la base de données en utilisant l'ID du document.

`getAll()` : pour lire un ou plusieurs documents à partir de la base de données en utilisant l'ID du document.

`update()` : pour modifier un document existant dans la base de données.

`updates()` : pour modifier un ou plusieurs documents existant dans la base de données.

`delete()` : pour supprimer un document de la base de données.

`deletes()` : pour supprimer un ou plusieurs documents de la base de données.

Pour les index secondaires, avec MongoDB, il n'est pas nécessaire de créer un index primaire. Par contre, il serait judicieux de créer des index secondaires pour accélérer certaines requêtes. Voici les commandes pour créer des index secondaires sur les champs appropriés :

```
db.Apartments.createIndex({ ownerId: 1 });
db.Clients.createIndex({ clientId: 1 });
db.Reservation.createIndex({ apartmentId: 1 });
db.Transactions.createIndex({ clientId: 1 });
db.Users.createIndex({ userId: 1 });
```

En ce qui concerne les méthodes applicatives de consultation, on va utiliser le langage de requête de MongoDB. Donc les méthodes vont être répartis entre les opérations sans requête et avec requête. Voici les opérations.

Pour les opérations sans N1QL:

- ✓ Recherche d'appartements par utilisateur

```
db.Apartments.find({ ownerId: userId })
```

- ✓ Recherche de transactions par client

```
db.Transactions.find({ clientId: clientId })
```

- ✓ Recherche de réservations par appartement

```
db.Reservation.find({ apartmentId: apartmentId })
```

- ✓ Recherche d'appartements par nombre de chambres

```
db.Apartments.find({ numberOfRooms: numberOfRooms })
```

- ✓ Recherche de clients par pays d'origine

```
db.Clients.find({ countryOfOrigin: country })
```

- ✓ Trouver tous les appartements dans une certaine gamme de prix

```
db.Apartments.find({ defaultPrice: { $gte: minPrice, $lte:
maxPrice } })
```

Pour les opérations utilisant N1QL:

- ✓ Trouver tous les utilisateurs qui n'ont pas encore vérifié leur compte

```
db.Users.find({ isVerified: false })
```

- ✓ Trouver toutes les réservations pour un appartement spécifique

```
db.Reservation.find({ apartmentId: apartmentId })
```

- ✓ Trouver tous les clients d'un pays d'origine spécifique

```
db.Clients.find({ countryOfOrigin: country })
```

- ✓ Trouver tous les appartements ayant un certain nombre de salles de bains

```
db.Apartments.find({ numberOfBathrooms: numberOfBathrooms })
```