Atelier : Modélisation et Implémentation d'une Base de Données pour une Clinique Médicale en MongoDB

1. Modèle physique de données

Patients

Description : Informations sur les patients.

Champs :

- `_id` (String) : Identifiant unique du patient.
- `nom` (String) : Nom du patient.
- `prénom` (String) : Prénom du patient.
- `dateNaissance` (Date) : Date de naissance du patient.
- `sexe` (Enum) : Sexe du patient (`Homme` ou `Femme`).
- `adresse` (String) : Adresse du patient.
- `téléphone` (String) : Numéro de téléphone du patient.
- `email` (String) : Adresse email du patient.
- `historiqueMédical` (Array de String) : Historique médical du patient.

Relations :

- **Un à Plusieurs** avec **Rendez-vous** : Un patient peut avoir plusieurs rendez-vous.
- **Un à Plusieurs** avec **Consultations** : Un patient peut avoir plusieurs consultations.

Médecins

Description : Informations sur les médecins.

Champs :

- `_id` (String) : Identifiant unique du médecin.
- `nom` (String) : Nom du médecin.
- `prénom` (String) : Prénom du médecin.
- `spécialité` (String) : Spécialité du médecin.
- `email` (String) : Adresse email du médecin.
- `téléphone` (String) : Numéro de téléphone du médecin.
- `salleConsultation` (String) : Salle de consultation du médecin.

Relations :

- **Un à Plusieurs** avec **Rendez-vous** : Un médecin peut avoir plusieurs rendez-vous.
- **Un à Plusieurs** avec **Consultations :** Un médecin peut avoir plusieurs consultations.

Rendez-vous

Description : Informations sur les rendez-vous entre patients et médecins.

Champs :

- `_id` (String) : Identifiant unique du rendez-vous.

- `patientId` (String) : Référence à l'identifiant du patient.
- `medecinId` (String) : Référence à l'identifiant du médecin.
- `dateRendezVous` (Date) : Date et heure du rendez-vous.
- `statut` (Enum) : Statut du rendez-vous (`Confirmé`, `Annulé`, `En attente`).

Relations :

- **Plusieurs à Un** avec **Patients** : Un rendez-vous est associé à un patient.
- **Plusieurs à Un** avec **Médecins** : Un rendez-vous est associé à un médecin.

Consultations

Description : Informations sur les consultations médicales, incluant les ordonnances.

Champs :

- `_id` (String) : Identifiant unique de la consultation.
- `patientId` (String) : Référence à l'identifiant du patient.
- `medecinId` (String) : Référence à l'identifiant du médecin.
- `dateConsultation` (Date) : Date et heure de la consultation.
- `diagnostic` (String) : Diagnostic établi lors de la consultation.
- `ordonnance` (Array d'objets) : Liste des médicaments prescrits.
  - `nom` (String) : Nom du médicament.
  - `posologie` (String) : Posologie du médicament.
  - `durée` (String) : Durée du traitement.

Relations :

- **Plusieurs à Un** avec **Patients** : Une consultation est associée à un patient.
- **Plusieurs à Un** avec **Médecins** : Une consultation est associée à un médecin.

2. Création de la Base de Données et des Collections avec MongoDB

2.1. Créer la base de données

- Utilisez la commande suivante pour créer une base de données :

```
test> use cliniqueDB
switched to db cliniqueDB
```

2.2. Créer les Collections avec Validation de Schéma

Patients Collection :

```
db.createCollection("patients", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["_id", "nom", "prénom", "dateNaissance", "sexe", "adresse", "téléphone", "email"],
      properties: {
        _id: { bsonType: "string" },
        nom: { bsonType: "string" },
        prénom: { bsonType: "string" },
        dateNaissance: { bsonType: "date" },
        sexe: { enum: ["Homme", "Femme"] },
        adresse: { bsonType: "string" },
        téléphone: { bsonType: "string" },
        email: { bsonType: "string", pattern: "^.+@.+\\..+$" },
        historiqueMédical: { bsonType: "array", items: { bsonType: "string" } }
      }
    }
  }
})
```

Médecins Collection :

```
db.createCollection("medecins", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["_id", "nom", "prénom", "spécialité", "email", "téléphone"],
      properties: {
        _id: { bsonType: "string" },
        nom: { bsonType: "string" },
        prénom: { bsonType: "string" },
```

```
      spécialité: { bsonType: "string" },
      email: { bsonType: "string", pattern: "^.+@.+\\..+$" },
      téléphone: { bsonType: "string" },
      salleConsultation: { bsonType: "string" }
    }
   }
  }
})
```

Rendez-vous Collection :

```
db.createCollection("rendezvous", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["_id", "patientId", "medecinId", "dateRendezVous", "statut"],
      properties: {
        _id: { bsonType: "string" },
        patientId: { bsonType: "string" },
        medecinId: { bsonType: "string" },
        dateRendezVous: { bsonType: "date" },
        statut: { enum: ["Confirmé", "Annulé", "En attente"] }
      }
    }
  }
})
```

Consultations Collection :

```
db.createCollection("consultations", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["_id", "patientId", "medecinId", "dateConsultation", "diagnostic"],
      properties: {
        _id: { bsonType: "string" },
        patientId: { bsonType: "string" },
        medecinId: { bsonType: "string" },
        dateConsultation: { bsonType: "date" },
        diagnostic: { bsonType: "string" },
        ordonnance: {
          bsonType: "array",
          items: {
            bsonType: "object",
            required: ["nom", "posologie", "durée"],
            properties: {
              nom: { bsonType: "string" },
              posologie: { bsonType: "string" },
              durée: { bsonType: "string" }
            }
          }
        }
      }
    }
  }
})
```

3. **Insertion des Données**

3.1. Insérer des patients

```
db.patients.insertMany([
  {
    _id: "P001",
    nom: "Mohammed",
    prénom: "Ali",
    dateNaissance: new Date("1990-05-15"),
    sexe: "Homme",
    adresse: "Casablanca",
    téléphone: "0600123456",
    email: "mohammed.ali@example.com"
  },
  {
    _id: "P002",
    nom: "Fatima",
    prénom: "Zahra",
    dateNaissance: new Date("1985-07-10"),
    sexe: "Femme",
    adresse: "Rabat",
    téléphone: "0612345678",
    email: "fatima.zahra@example.com"
  }
]);
```

3.2. Insérer des médecins

```
db.medecins.insertMany([
  {
    _id: "M001",
    nom: "Ahmed",
    prénom: "Kamal",
    spécialité: "Cardiologie",
```

```
    email: "ahmed.kamal@example.com",
    téléphone: "0623456789",
    salleConsultation: "101"
  },
  {
    _id: "M002",
    nom: "Leila",
    prénom: "Hassan",
    spécialité: "Dermatologie",
    email: "leila.hassan@example.com",
    téléphone: "0634567890",
    salleConsultation: "202"
  }
]);
```

3.3. Insérer des rendez-vous

```
db.rendezvous.insertOne({
  _id: "R001",
  patientId: "P001",
  medecinId: "M001",
  dateRendezVous: new Date("2025-03-20"),
  statut: "Confirmé"
});
```

3.4. Insérer des consultations

```
db.consultations.insertOne({
  _id: "C001",
  patientId: "P001",
  medecinId: "M001",
  dateConsultation: new Date("2025-03-21"),
  diagnostic: "Hypertension",
  ordonnance: [
```

```
    {
      nom: "Doliprane",
      posologie: "1 comprimé matin et soir",
      durée: "5 jours"
    },
    {
      nom: "Amlodipine",
      posologie: "1 comprimé par jour",
      durée: "30 jours"
    }
  ]
});
```

4. **Vérification des Données Insérées**

Pour vérifier que les données ont été correctement insérées, vous pouvez utiliser les commandes suivantes :

- **Vérifier les patients :**

```
db.patients.find().pretty()
```

- **Vérifier les médecins :**

```
db.medecins.find().pretty()
```

- **Vérifier les rendez-vous :**

```
db.rendezvous.find().pretty()
```

- **Vérifier les consultations :**

```
db.consultations.find().pretty()
```