TP 1 — Premières manipulations avec MongoDB

Prérequis : aucune connaissance préalable de MongoDB. Juste un terminal. Durée indicative : 45 minutes à 1 heure.

Objectif : se familiariser avec mongosh et manipuler les opérations CRUD de base (Create, Read, Update, Delete), à travers des exemples concrets et simples.

Étape 1 — Démarrer MongoDB

Le plus simple pour débuter, sans rien installer sur sa machine :

```
docker run -d --name mon-mongo -p 27017:27017 mongo:7
```

Vérifier que le conteneur tourne :

```
docker ps
```

Se connecter avec le shell interactif :

```
docker exec -it mon-mongo mongosh
```

Vous devez voir un prompt de ce type :

```
test>
```

À partir de maintenant, toutes les commandes du TP se tapent directement dans ce prompt.

Premier test :

```
test> db.version()
```

Si une version s'affiche (ex. 7.0.x), tout fonctionne.

Étape 2 — Créer une base et une collection

```
test> use bibliotheque
switched to db bibliotheque
```

use bibliotheque sélectionne (ou crée) la base bibliotheque — elle n'apparaîtra dans show dbs qu'une fois qu'elle contiendra au moins un document.

```
bibliotheque> show dbs
```

Étape 3 — Créer un document (Create)

```
bibliotheque> db.livres.insertOne({ titre: "Les Misérables",
auteur: "Victor Hugo", annee: 1862 })
```

Résultat attendu (extrait) :

```
{
  acknowledged: true,
  insertedId: ObjectId("665f1a2b3c4d5e6f7a8b9c0d")
}
```

db.livres fait référence à la collection livres — elle est créée implicitement au premier insert, sans déclaration préalable. insertedId est l'identifiant unique généré automatiquement pour ce document.

Exercice 1 : insérez un deuxième livre avec les champs titre, auteur, annee de votre choix.

Insérer plusieurs documents d'un coup

```
bibliotheque> db.livres.insertMany([
  { titre: "1984", auteur: "George Orwell", annee: 1949, genre:
"science-fiction" },
  { titre: "Madame Bovary", auteur: "Gustave Flaubert", annee:
1857, genre: "roman" },
  { titre: "Fondation", auteur: "Isaac Asimov", annee: 1951, genre:
"science-fiction" }
])
```

Étape 4 — Lire des documents (Read)

Tout afficher

```
bibliotheque> db.livres.find()
```

Pour un affichage plus lisible, ajouter .pretty() :

```
bibliotheque> db.livres.find().pretty()
```

Lire un seul document

```
bibliotheque> db.livres.findOne({ titre: "1984" })
```

Filtrer

```
bibliotheque> db.livres.find({ genre: "science-fiction" })
```

N'afficher que certains champs (projection)

```
bibliotheque> db.livres.find({ genre: "science-fiction" },
{ titre: 1, annee: 1, _id: 0 })
```

1 = inclure le champ, 0 = l'exclure. _id est inclus par défaut, sauf exclusion explicite.

Trier et limiter

```
bibliotheque> db.livres.find().sort({ annee: -1 })   // du plus
récent au plus ancien
bibliotheque> db.livres.find().limit(2)              // les 2
premiers résultats
```

Compter

```
bibliotheque> db.livres.countDocuments({ genre: "science-
fiction" })
```

Exercice 2 : affichez uniquement les titres des livres publiés après 1900, triés par année croissante.

Étape 5 — Modifier un document (Update)

Modifier un champ

```
bibliotheque> db.livres.updateOne(
  { titre: "1984" },
  { $set: { editeur: "Secker & Warburg" } }
)
```

Résultat attendu :

```
{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }
```

Vérifier le résultat :

```
bibliotheque> db.livres.findOne({ titre: "1984" })
```

Modifier plusieurs documents à la fois

```
bibliotheque> db.livres.updateMany(
  { genre: "science-fiction" },
  { $set: { rayon: "SF" } }
)
```

Incrémenter un nombre

```
bibliotheque> db.livres.updateOne({ titre: "1984" }, { $inc:
{ nombre_lectures: 1 } })
bibliotheque> db.livres.updateOne({ titre: "1984" }, { $inc:
{ nombre_lectures: 1 } })
bibliotheque> db.livres.findOne({ titre: "1984" },
{ nombre_lectures: 1 })
```

Le champ nombre_lectures n'existait pas au départ : $inc le crée à 1 au premier appel, puis l'incrémente normalement ensuite.

Exercice 3 : ajoutez un champ disponible: true à tous les livres de la collection avec updateMany, puis marquez un livre précis comme emprunté (disponible: false).

Étape 6 — Supprimer un document (Delete)

Supprimer un seul document

```
bibliotheque> db.livres.deleteOne({ titre: "Madame Bovary" })
```

Résultat attendu :

```
{ acknowledged: true, deletedCount: 1 }
```

Supprimer plusieurs documents

```
bibliotheque> db.livres.deleteMany({ genre: "roman" })
```

Attention : deleteMany({}) (filtre vide) supprime tous les documents de la collection — à utiliser avec prudence.

Étape 7 — Explorer ce qui existe

Lister les collections de la base actuelle :

```
bibliotheque> show collections
```

Compter tous les documents d'une collection :

```
bibliotheque> db.livres.countDocuments()
```

Voir la structure d'un document au format lisible :

```
bibliotheque> db.livres.findOne()
```

Étape 8 — Tout nettoyer

Supprimer tous les documents d'une collection :

```
bibliotheque> db.livres.deleteMany({})
```

Supprimer complètement la collection (structure comprise) :

```
bibliotheque> db.livres.drop()
```

Quitter le shell :

```
bibliotheque> exit
```

Arrêter le conteneur MongoDB (optionnel) :

```
docker stop mon-mongo
```

Récapitulatif des commandes vues

Commande	Rôle
`use <base>`	Sélectionner ou créer une base de données
`insertOne` / `insertMany`	Créer un ou plusieurs documents
`find` / `findOne`	Lire des documents (tous ou un seul)
Projection `{ champ: 1 }`	Choisir les champs retournés
`sort` / `limit`	Trier et limiter les résultats
`countDocuments`	Compter les documents correspondant à un filtre
`updateOne` / `updateMany`	Modifier un ou plusieurs documents
`$set` / `$inc`	Définir un champ / incrémenter une valeur
`deleteOne` / `deleteMany`	Supprimer un ou plusieurs documents
`show collections`	Lister les collections d'une base
`drop()`	Supprimer une collection entière

Corrigé des exercices

Exercice 1 :

```
db.livres.insertOne({ titre: "Votre titre", auteur: "Votre
auteur", annee: 2020 })
```

Exercice 2 :

```
db.livres.find({ annee: { $gt: 1900 } }, { titre: 1, annee: 1,
_id: 0 }).sort({ annee: 1 })
```

Exercice 3 :

```
db.livres.updateMany({}, { $set: { disponible: true } })
db.livres.updateOne({ titre: "1984" }, { $set: { disponible: false
} })
```

Pour la suite

Une fois ces manipulations à l'aise, la suite logique est le module 6 du cours (requêtes avancées : $gt, $in, $or, recherche dans les tableaux) puis le module 8 (framework d'agrégation) pour calculer des statistiques sur vos données — par exemple compter le nombre de livres par genre.