|  |  |  |  |
| --- | --- | --- | --- |
| **📌 N°** | **Catégorie** | **Commande** | **Explication** |
| **1** | **Création de base & collection** | **use TDI01** | **Crée ou sélectionne la base de données TDI01.** |
|  |  | **db.createCollection("stagiaire")** | **Crée une collection appelée stagiaire.** |
| **2** | **Insertion de documents** | **db.stagiaire.insert({...})** | **Insère un document (objet JSON) dans la collection stagiaire.** |
|  |  | **db.stagiaire.insert([...])** | **Insère plusieurs documents à la fois (liste de JSON).** |
|  |  | **db.stagiaire.insertOne({...})** | **Insère un seul document (équivalent simplifié de insert).** |
| **3** | **Lecture / Consultation** | **db.stagiaire.find({})** | **Récupère tous les documents de la collection stagiaire.** |
|  |  | **db.stagiaire.find({"nom":"saad"})** | **Filtre les documents où nom est "saad".** |
|  |  | **db.stagiaire.find({"option":"Mobile"})** | **Filtre les documents avec l'option "Mobile".** |
|  | **Projection** | **db.stagiaire.find({"nom": "Ahmed"}, {"prenom":1, "option":1, "\_id":0})** | **Affiche uniquement prenom et option, sans afficher \_id.** |
| **4** | **Aide / Exploration** | **db.help()** | **Affiche la liste des commandes disponibles.** |
|  |  | **show dbs** | **Affiche toutes les bases de données.** |
|  |  | **show collections** | **Affiche les collections de la base actuelle.** |
| **5** | **Suppression de collection/db** | **db.Stagiaires.drop()** | **Supprime la collection Stagiaires.** |
|  |  | **db.dropDatabase()** | **Supprime toute la base de données courante.** |
| **6** | **Suppression de documents** | **db.stagiaire.deleteOne({"\_id":"1000"})** | **Supprime un document avec l’\_id spécifié.** |
|  |  | **db.stagiaire.deleteMany({"option":"Mobile"})** | **Supprime tous les documents avec l’option "Mobile".** |
| **7** | **Mise à jour** | **db.stagiaire.update({"\_id":"6000"}, {$set: {"niveau":"1A"}})** | **Met à jour un seul champ (niveau) sans modifier le reste.** |
|  | **Option** | **multi: true (dans les options)** | **Permet de mettre à jour plusieurs documents à la fois (si utilisé avec updateMany).** |
| **8** | **Opérateurs MongoDB** | **$eq, $gt, $gte, $lt, $lte, $ne** | **Opérateurs de comparaison : égal, supérieur, inférieur, etc.** |
|  |  | **$in, $nin** | **Vérifie si une valeur appartient (ou non) à une liste.** |
|  |  | **$or, $and, $nor** | **Opérateurs logiques pour combiner plusieurs conditions.** |
|  |  | **$exists** | **Vérifie si un champ est présent ou non dans un document.** |
| **9** | **Fonctions utiles** | **db.stagiaire.distinct("option")** | **Affiche les valeurs uniques du champ option.** |
|  |  | **db.stagiaire.find().count()** | **Affiche le nombre total de documents.** |
|  |  | **db.stagiaire.find().limit(2)** | **Limite l'affichage à 2 documents.** |
| **10** | **Agrégation avancée** | **db.stagiaire.aggregate([{$match: {"filiere.intitule":"Développement digital"}}])** | **Filtrage avec $match dans une pipeline.** |
|  |  | **db.stagiaire.aggregate([{$group:{\_id:"$filiere.intitule", total:{$sum:"$moy1A"}}}])** | **Regroupement par filière et somme des moyennes ($group + $sum).** |
|  |  | **db.stagiaire.find().sort({"nom": -1})** | **Trie les documents par nom en ordre décroissant.** |

|  |  |  |  |
| --- | --- | --- | --- |
| Opérateur | Symbole | Description | Exemple |
| Égalité | **=, $eq** | **Vérifie si une valeur est égale à celle du champ.** | **{ "age": { "$eq": 25 } } // Trouve les documents où "age" est égal à 25.** |
| Supérieur | **>, $gt** | **Vérifie si une valeur est supérieure à celle du champ.** | **{ "age": { "$gt": 25 } } // Trouve les documents où "age" est supérieur à 25.** |
| Supérieur ou égal | **>=, $gte** | **Vérifie si une valeur est supérieure ou égale à celle du champ.** | **{ "age": { "$gte": 25 } } // Trouve les documents où "age" est supérieur ou égal à 25.** |
| Inférieur | **<, $lt** | **Vérifie si une valeur est inférieure à celle du champ.** | **{ "age": { "$lt": 25 } } // Trouve les documents où "age" est inférieur à 25.** |
| Inférieur ou égal | **<=, $lte** | **Vérifie si une valeur est inférieure ou égale à celle du champ.** | **{ "age": { "$lte": 25 } } // Trouve les documents où "age" est inférieur ou égal à 25.** |
| Différent | **!=, $ne** | **Vérifie si une valeur est différente de celle du champ.** | **{ "age": { "$ne": 25 } } // Trouve les documents où "age" n'est pas égal à 25.** |
| Appartenance | **in, $in** | **Vérifie si une valeur appartient à un tableau donné.** | **{ "age": { "$in": [25, 30, 35] } } // Trouve les documents où "age" est 25, 30, ou 35.** |
| Non-appartenance | **nin, $nin** | **Vérifie si une valeur n'appartient pas à un tableau donné.** | **{ "age": { "$nin": [25, 30, 35] } } // Trouve les documents où "age" n'est ni 25, ni 30, ni 35.** |
| OU logique | **or, $or** | **Vérifie si l'un des critères est vrai.** | **{ "$or": [ { "age": { "$gt": 25 } }, { "name": "John" } ] } // Trouve les documents où "age" est supérieur à 25 ou "name" est John.** |
| ET logique | **and, $and** | **Vérifie si tous les critères sont vrais.** | **{ "$and": [ { "age": { "$gt": 25 } }, { "name": "John" } ] } // Trouve les documents où "age" est supérieur à 25 ET "name" est John.** |
| NI logique | **nor, $nor** | **Vérifie si aucun des critères n'est vrai.** | **{ "$nor": [ { "age": { "$gt": 25 } }, { "name": "John" } ] } // Trouve les documents où ni "age" est supérieur à 25 ni "name" est John.** |
| Existence | **exists** | **Vérifie si un champ existe ou non.** | **{ "name": { "$exists": true } } // Trouve les documents où le champ "name" existe.** |

|  |  |  |
| --- | --- | --- |
| **Méthode** | **Description** | **Exemple complet MongoDB** |
| **$match** | **Filtrer les documents** | **db.stagiaire.aggregate([{ $match: { niveau: "2A" } }])** |
| **$group** | **Regrouper et faire des calculs** | **db.stagiaire.aggregate([{ $group: { \_id: "$option", total: { $sum: "$moy1A" } } }])** |
| **$sort** | **Trier les résultats** | **db.stagiaire.aggregate([{ $sort: { nom: 1 } }])** |
| **$project** | **Sélectionner ou masquer des champs** | **db.stagiaire.aggregate([{ $project: { nom: 1, prenom: 1, \_id: 0 } }])** |
| **$limit** | **Limiter le nombre de résultats** | **db.stagiaire.aggregate([{ $limit: 3 }])** |
| **$skip** | **Ignorer un certain nombre de documents** | **db.stagiaire.aggregate([{ $skip: 2 }])** |
| **$count** | **Compter les documents** | **db.stagiaire.aggregate([{ $match: { niveau: "2A" } }, { $count: "total\_2A" }])** |
| **$sum** | **Somme d’un champ** | **db.stagiaire.aggregate([{ $group: { \_id: "$filiere.intitule", total: { $sum: "$moy1A" } } }])** |
| **$avg** | **Moyenne d’un champ** | **db.stagiaire.aggregate([{ $group: { \_id: "$option", moyenne: { $avg: "$moy1A" } } }])** |
| **$min** | **Valeur minimale** | **db.stagiaire.aggregate([{ $group: { \_id: "$niveau", minNote: { $min: "$moy1A" } } }])** |
| **$max** | **Valeur maximale** | **db.stagiaire.aggregate([{ $group: { \_id: "$niveau", maxNote: { $max: "$moy1A" } } }])** |
| **$first** | **Premier élément du groupe** | **db.stagiaire.aggregate([{ $group: { \_id: "$option", premier: { $first: "$nom" } } }])** |
| **$last** | **Dernier élément du groupe** | **db.stagiaire.aggregate([{ $group: { \_id: "$option", dernier: { $last: "$nom" } } }])** |
| **$addFields** | **Ajouter un champ calculé** | **db.stagiaire.aggregate([{ $addFields: { double\_note: { $multiply: ["$moy1A", 2] } } }])** |
| **$unwind** | **Déplier un tableau** | **db.stagiaire.aggregate([{ $unwind: "$modules" }]) MongoDB stocke des données sous forme de documents** |
| **$lookup** | **Jointure entre deux collections** | **db.stagiaire.aggregate([{ $lookup: { from: "filiere", localField: "filiere\_id", foreignField: "\_id", as: "info\_filiere" } }])** |
| **$facet** | **Pipelines parallèles** | **db.stagiaire.aggregate([{ $facet: { niveau1A: [{ $match: { niveau: "1A" } }], niveau2A: [{ $match: { niveau: "2A" } }] } }])** |
| **$bucket** | **Regrouper les données par intervalles** | **db.stagiaire.aggregate([{ $bucket: { groupBy: "$moy1A", boundaries: [0, 10, 15, 20], default: "Hors intervalle", output: { count: { $sum: 1 } } } }])** |
| **$regex** | **tu veux récupérer tous les événements dont le thème contient le mot "Web"** | **db.evenements.aggregate([**  **{ $match: { theme: { $regex: "Web", $options: "i" } } }, // "i" = insensible à la casse**  **{ $unwind: "$Ateliers" },**  **{ $count: "totalAteliers" }**  **])** |
| **$size** | **L'opérateur $size permet de :**  **🧪 Tester la longueur d’un tableau (dans $match)**  **🧮 Calculer la longueur d’un tableau (dans $project)**  **{ "\_id": 1,**  **"theme": "Développement Web",**  **"Ateliers": ["HTML", "CSS", "JS"] }** | **db.evenements.find({**  **Ateliers: { $size: 3 }**  **})**  **db.evenements.aggregate([**  **{**  **$project: {**  **theme: 1,**  **nombreAteliers: { $size: "$Ateliers" } }}])** |