Atelier : Maîtrise de l'Agrégation dans MongoDB pour les Étudiants en Licence Informatique

---

Objectif de l'Atelier :

- Comprendre le **pipeline d'agrégation** dans MongoDB.
- Manipuler les étapes courantes comme `$match`, `$group`, `$sort`, `$project`, `$unwin`$unwind` et `$limit`.
- Résoudre d

1. Introduction Théorique

A. Qu'est-ce que l'agrégation ?

- MongoDB permet de transformer et analyser des données grâce au **pipeline d'agrégation**.
- Les données passent par plusieurs **étapes (stages)** pour produire un résultat fi

Exemple d'un pipeline :

```
Entrée → $match → $group → $sort → $project → Sortie
```

B. Présentation des étapes essentielles :

1. **`$match`** : Filtrer les document
2. **`$group`** : Regrouper les documents et appliquer des calculs comme `$sum`, `$a`$avg`, etc.
3. **`$sort`** : Trier les documents selon un ou plusieur
4. **`$project`** : Sélectionner ou renommer des champs.
5. **`$unwind`** : Décomposer les tableau
6. **`$limit`** et **`$skip`** : Limiter ou sauter un nombre de d

---

2. Mise en Place des Données pour la Pratique

Fournir un fichier **`employes.json`** avec la collection `empl`employes` pour les exercices.

Exemple de structure des documents :

```
 [
   { "_id": 1, "nom": "Alice", "sexe": "F", "age": 25, "salaire": 3000,
"departement": "IT", "enfants": ["Sam", "Lily"] },
```

```
   { "_id": 2, "nom": "Bob", "sexe": "M", "age": 30, "salaire": 4000,
"departement": "Finance", "enfants": ["John"] },
   { "_id": 3, "nom": "Clara", "sexe": "F", "age": 35, "salaire": 5000,
"departement": "IT", "enfants": [] },
   { "_id": 4, "nom": "David", "sexe": "M", "age": 40, "salaire": 6000,
"departement": "HR", "enfants": ["Nina"] }
]
```

Exercice 1 : Filtrer les données avec `$match`

1. Listez tous les employés du département **IT**.

```
db.employes.aggregate([
   { $match: { departement: "IT" } }
]);
```

2. Affichez les employés dont le salaire est **supérieur ou égal à 4000**.

Exercice 2 : Grouper les données avec `$group`

1. Calculez le **salaire moyen** par département.

```
db.employes.aggregate([
   { $group: { _id: "$departement", salaire_moyen: { $avg: "$salaire" }
} }
]);
```

2. Comptez le nombre d'employés par **sexe**.

Exercice 3 : Trier et limiter les résultats avec `$sort` et `$limit`

1. Listez les employés triés par **salaire décroissant**.

```
db.employes.aggregate([
   { $sort: { salaire: -1 } }
]);
```

2. Affichez les **2 employés les mieux payés**.

Exercice 4 : Transformer les documents avec `$project`

1. Affichez uniquement les champs `nom`, `departement` et `salaire`.

```
db.employes.aggregate([
   { $project: { _id: 0, nom: 1, departement: 1, salaire: 1 } }
]);
```

2. Renommez le champ `salaire` en `revenu`.

Exercice 5 : Manipuler les tableaux avec `$unwind`

1. Décomposez le tableau `enfants` pour afficher chaque enfant dans un document séparé.

```
db.employes.aggregate([
   { $unwind: "$enfants" }
]);
```

2. Comptez le nombre total d'enfants dans l'ensemble de la collection.

Exercice 6 : Pipeline complet

Objectif : Trouver le salaire moyen des employés du département **IT** ayant au moins un enfant.

```
db.employes.aggregate([
   { $match: { departement: "IT", enfants: { $ne: [] } } },
   { $group: { _id: "$departement", salaire_moyen: { $avg: "$salaire" } } }
]);
```

---

4. Activité de Groupe : Mini-Projet (30 minutes)

Objectif : Les étudiants travailleront en groupe pour résoudre le problème suivant :

- Affichez les **3 employés les mieux payés**, en indiquant leur nom, salaire, et nombre d'enfants.

Solution Possible :

```
db.employes.aggregate([
   { $project: { nom: 1, salaire: 1, nombre_enfants: { $size: "$enfants" } }
},
   { $sort: { salaire: -1 } },
   { $limit: 3 }
]);
```
