

## Rapport : atelier5 exercice1 et 2

**Module** : Base de Données MongoDB

**Nom**: Aya ouazzani akkar hammoudi

### Exercice 1 :

#### SOMMAIRE :

1. Création de la Base de Données et des Collections:
2. Afficher les salariés triés par ordre croissant des noms
3. Afficher le nombre des salariés ayant la fonction "Technicien"
4. Supprimer le salarié ayant l'\_id "s3"
5. Afficher le nombre de salariés par fonction

#### 1. Création de la Base de Données et des Collections:

### Create Database ×

Database Name

DBSalaries

Collection Name

salaries

☐ Time-Series

Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

Additional preferences (e.g., Custom collection, Sharded collection)

Cancel

Create Database

```
DBSalaries> db.salaries.insertOne({
  _id: "s1",
  nomsal: "Alami",
  prenomsal: "Sara",
  fonction: "Technicien",
  service: 1,
  codeser: "1",
  nomser: "informatique"
});
```

```
< {
  acknowledged: true,
  insertedId: 's1'
}
```

2. Afficher les salariés triés par ordre croissant des noms

```
DBSalaries> db.salaries.find().sort({ nomsal: 1 });
```

```
< {
  _id: 's1',
  nomsal: 'Alami',
  prenomsal: 'Sara',
  fonction: 'Technicien',
  service: 1,
  codeser: '1',
  nomser: 'informatique'
}
DBSalaries>
```

3. Afficher le nombre des salariés ayant la fonction "Technicien"

```
DBSalaries> db.salaries.countDocuments({ fonction: "Technicien" });
> db.salaries.countDocuments({ fonction: "Technicien" });
< 1
```

4. Supprimer le salarié ayant l'\_id "s3"

```
> db.salaries.deleteOne({ _id: "s3" });  
< {  
  acknowledged: true,  
  deletedCount: 0  
}  
DBSalaries> |
```

5. Afficher le nombre de salariés par fonction

```
DBSalaries> db.salaries.aggregate([  
  {  
    $group: {  
      _id: "$fonction",  
      nombre: { $sum: 1 }  
    }  
  }  
]);
```

```
< {  
  _id: 'Technicien',  
  nombre: 1  
}
```

## Exercice 2 :

### SOMMAIRE :

1. Création de la Base de Données et des Collections:
2. Afficher les notaires travaillant à "Rabat", triés par nom croissant
- 3 Afficher le nombre total des terrains pour chaque type
4. Afficher le prix maximum de tous les terrains de type "agricole"
5. Augmenter de 1% le prix de tous les terrains agricoles des propriétaires cin=P1 et cin=P4
6. Supprimer l'adresse et le téléphone du notaire numn = "N2"

## 1. Création de la Base de Données et des Collections:

### Create Database ✕

Database Name

DBTerrains

Collection Name

terrains

☐ Time-Series

Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

☐ Additional preferences (e.g. Ordered collection, Clustered collection)

Cancel

Create Database

```
DBTerrains> db.terrains.insertMany([
  {
    numT: "T1",
    type: "agricole",
    prix: 100000,
    titre_foncier: "TF101",
    cin: "P1",
    adresse: "Rabat",
    superficie: 500,
    notaire: {
      numn: "N1",
      nom: "Benali",
      prenom: "Mohamed",
      adresse: "Rabat",
      tel: "0537123456"
    }
  },
])
```

```
{
  numT: "T2",
  type: "lotissement",
  prix: 150000,
  titre_foncier: "TF102",
  cin: "P2",
  adresse: "Casablanca",
  superficie: 400,
  notaire: {
    numn: "N2",
    nom: "El Fassi",
    prenom: "Fatima",
    adresse: "Casablanca",
    tel: "0522123456"
  }
},
```

```
{
  numT: "T3",
  type: "agricole",
  prix: 200000,
  titre_foncier: "TF103",
  cin: "P4",
  adresse: "Marrakech",
  superficie: 600,
  notaire: {
    numn: "N3",
    nom: "Chikhi",
    prenom: "Ahmed",
    adresse: "Marrakech",
    tel: "0524123456"
  }
}
]);
```

```
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('691900d9d6bea68f5c828070'),
    '1': ObjectId('691900d9d6bea68f5c828071'),
    '2': ObjectId('691900d9d6bea68f5c828072')
  }
}
```

2. Afficher les notaires travaillant à "Rabat", triés par nom croissant

```
DBTerrains> db.terrains.find(
  { "notaire.adresse": "Rabat" },
  { _id: 0, "notaire.nom": 1, "notaire.prenom": 1, "notaire.adresse": 1 }
).sort({ "notaire.nom": 1 });
```

```
< {
  notaire: {
    nom: 'Benali',
    prenom: 'Mohamed',
    adresse: 'Rabat'
  }
}
```

3. Afficher le nombre total des terrains pour chaque type

```
DBTerrains> db.terrains.aggregate([
  {
    $group: {
      _id: "$type",
      nombreTerrains: { $sum: 1 }
    }
  }
]);
```

```
< {
  _id: 'agricole',
  nombreTerrains: 2
}
{
  _id: 'lotissement',
  nombreTerrains: 1
}
```

4. Afficher le prix maximum de tous les terrains de type "agricole"

```
DBTerrains> db.terrains.aggregate([
  { $match: { type: "agricole" } },
  {
    $group: {
      _id: null,
      prixMax: { $max: "$prix" }
    }
  }
]);
```

```
< {
  _id: null,
  prixMax: 200000
}
```

5. Augmenter de 1% le prix de tous les terrains agricoles des propriétaires cin=P1 et cin=P4

```
DBTerrains> db.terrains.updateMany(
  { type: "agricole", cin: { $in: ["P1", "P4"] } },
  { $mul: { prix: 1.01 } } |
);
```

```
< {  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 2,  
  modifiedCount: 2,  
  upsertedCount: 0  
}
```

6. Supprimer l'adresse et le téléphone du notaire numn = "N2"

```
DBTerrains> db.terrains.updateMany(  
  { "notaire.numn": "N2" },  
  { $unset: { "notaire.adresse": "", "notaire.tel": "" } }  
);
```

```
< {  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 1,  
  modifiedCount: 1,  
  upsertedCount: 0  
}
```