

# Mettre en place une base de données MongoDB

# Installation de mongoDB

- Nous allons installer le serveur mongoDB localement
- Rendez-vous sur la page:

<https://www.mongodb.com/try/download/community>

- On va également installer le shell

<https://www.mongodb.com/products/shell>

- On va installer l'outils compass aussi

# Les étapes d'installation

- Choisir une installation complète
- Installer mongoDB comme un service
  - Run service as Network Service User
- Ajouter le répertoire bin de mongoDB dans le **PATH**
- Lancer la commande **mongod** dans l'invite de commande
- Créer un répertoire **c:/data/db**

# Création d'une base de données mongoDB

- Pour créer une base de données, on utilise la commande **use**:

```
use NomBase
```

- **Remarque:** une base de données qui est vide n'est pas affichée parmi les bases du serveur, elle doit contenir au moins une collection pour l'afficher
- Afficher les bases disponibles:

```
show dbs
```

# Supprimer une base de données

- La commande **dropDatabase** est utilisée pour supprimer une base de données. Il supprime également les fichiers de données associés. Il fonctionne sur la base de données actuelle.

```
db.dropDatabase()
```

# Collection

- Une **collection** est un **ensemble de documents**,
- On vient de voir que c'est l'équivalent d'une **table** en relationnel,
- Contrairement aux bases de données relationnelles, les collections n'ont pas de schéma spécifique que les documents doivent respecter,
- Les champs des documents d'une collection sont libres et peuvent être différents d'un document à un autre. Le seul champ commun est obligatoire est le "**\_id**".
- Néanmoins pour que la base soit maintenable, il est préférable d'avoir dans une collection des documents de même type

# Création d'une collection dans une base

- Pour créer une nouvelle collection dans une base mongodb, on utilise la commande suivante:

```
db.createCollection ('nomCOollection')
```

- **Remarque:** la création d'une collection de cette manière n'est pas nécessaire, une collection est créée, une fois on crée des documents à l'intérieur,

# Lister les collections

- Une collection pour une base de données MongoDB est l'équivalent d'une table pour une base de données relationnelle, sauf que la collection n'a pas de schéma (voir après)
- Une base de données MongoDB, gère une collection de documents et non des tables,
- Pour lister les collections d'une base de données, on exécute la commande

`show collections`

```
use admin
show collections
use local
show collections
use config
show collections[
```

```
switched to db admin
system.version
switched to db local
startup_log
switched to db config
system.sessions
```

**admin** contient la collection **system.version**  
**local** contient la collection **startup\_log**  
Et **config** contient la collection **system.sessions**



# Supprimer une collection

- Pour supprimer une collection d'une base de données, on exécute la commande:

```
db.nomcollection.drop()
```

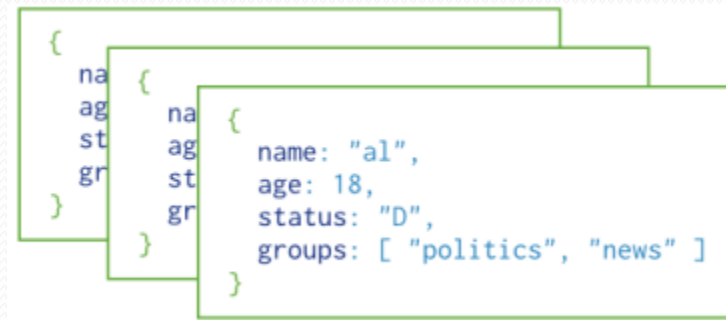
# Document

- Les documents sont les unités de base dans une base MongoDB,
- Ils sont l'équivalent des enregistrements des tables dans une base de données relationnelle,
- **Ils sont représentés sous forme d'objets JSON**
- Tout document appartient à une collection et a un champ appelé **\_id** qui l'identifie dans la base de données,
- Exemple:

```
{  
  "nom" : "Sanaa",  
  "filier" : "Dev Digital",  
  "niveau" : "2A",  
  "option": "Mobile"  
}
```

# Collection

- Collection avec des documents de **même schéma**



```
{
  name: "al",
  age: 18,
  status: "D",
  groups: [ "politics", "news" ]
}
```

- Collection avec des documents de **schémas différents**



```
{
  name: "al",
  age: 18,
  status: "D"
}
```

# création d'une collection dans une base de données

- Soit la base de données **myDB** qu'on a déjà créé,
- Pour créer une collection **myCollection** dans la base **myDB**, il suffit d'ajouter un (ou plusieurs) documents à la collection dans la base de données:
  - Utiliser la base de données  
**use myDB**
  - Ajouter un document (simple objet JSON dans ce cas) à la collection:

```
db.myCollection.insert(  
  { "id" : "Sanaa",  
    "filier" : "Dev Digital",  
    "niveau" : "2A",  
    "option": "Mobile"  
  }  
)
```

# Ajouter un document à une collection

- Si une collection est déjà créée et on veut lui ajouter des documents, on peut utiliser le code suivant:

```
db.getCollection("NomCollection").insert({...})
```

# Ajouter un document à une collection

- On peut utiliser la commande **insertOne()** si on veut insérer un seul document

```
db.myCollection.insertOne(  
  
  { "id" : "Sanaa",  
    "filiere" : "Dev Digital",  
    "niveau" : "2A",  
    "option": "Mobile" }  
)
```

- On peut utiliser la commande **insertMany()** si on veut insérer plusieurs documents à la fois dans un tableau.

```
db.myCollection.insertMany(  
  
  [  
    { "id" : "Sanaa", "filiere" : "Dev Digital", "niveau" : "2A", "option": "Mobile" },  
    { "id" : "Sanaa", "filiere" : "Dev Digital", "niveau" : "2A", "option": "Mobile" }  
  ]  
)
```

# création d'une collection dans une base de données

- Afficher les bases de données du serveur:

**Show dbs**

- On doit avoir ce résultat

```
Bulk/this.insert@src/mongo/shell/bulk_api.js:650:20
DBCollection.prototype.insert@src/mongo/shell/collection.js:313:13
@(shell):1:1
admin    0.000GB
config  0.000GB
local    0.000GB
switched to db myDB
Document inséré dans la collection → WriteResult({ "nInserted" : 1 })
admin    0.000GB
config  0.000GB
local    0.000GB
La base de données est effectivement créée → myDB    0.000GB
```