

**Projet NoSQL**

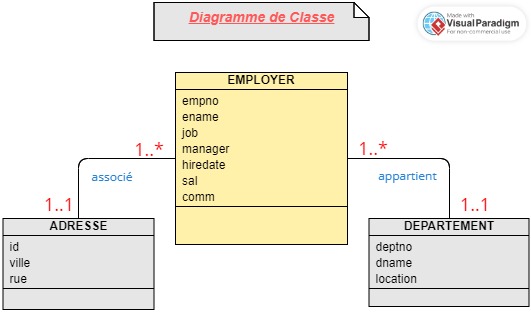


* Travail réalisé par les étudiants : Hamdi Helmi

Jaouabi Oussama

Ben Sghaier Omar

* Section : IF5
* AU : 2023 – 2024
* Proposé par L’enseignante : Houda Alaya



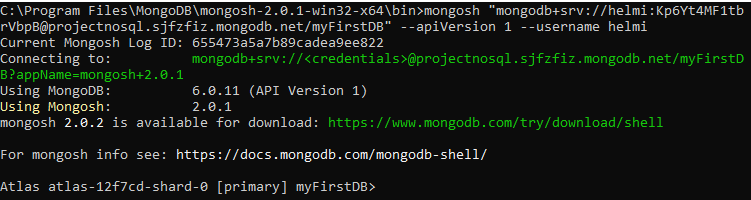
Le diagramme de classe suivante représente les collections utilisés dans notre base de donnée de gestion des employés avec :

* La collection Employer qui contient les informations sur tous

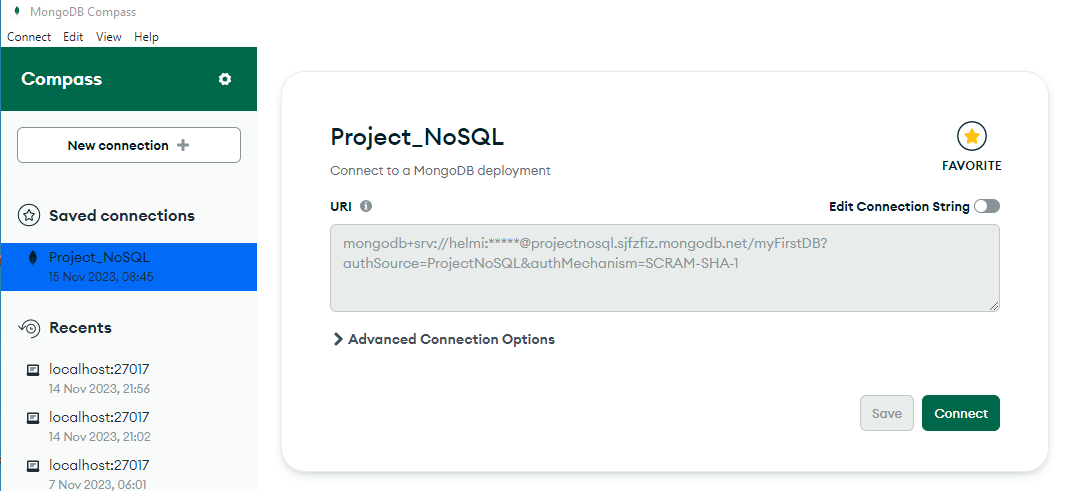
les employés.

* La collection Departement qui contient les informations sur tous les départements.
* La collection Adresse , contenant des informations sur les addresses de tous les employés.

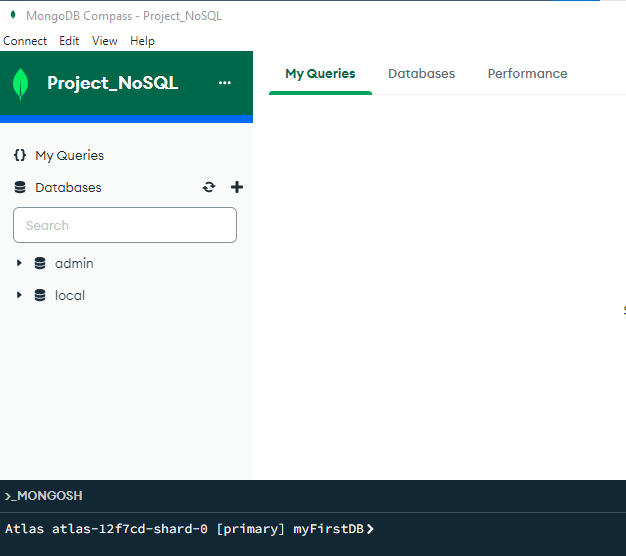
1. **Connection au cluster en ligne:**



**Ou bien à l’aide du Compass**



Puis on clique sur connect on obtient :



1. **Création de la base de donnée:**



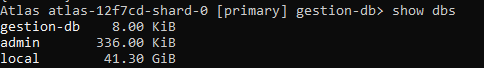
1. **Création de la collection employer:**

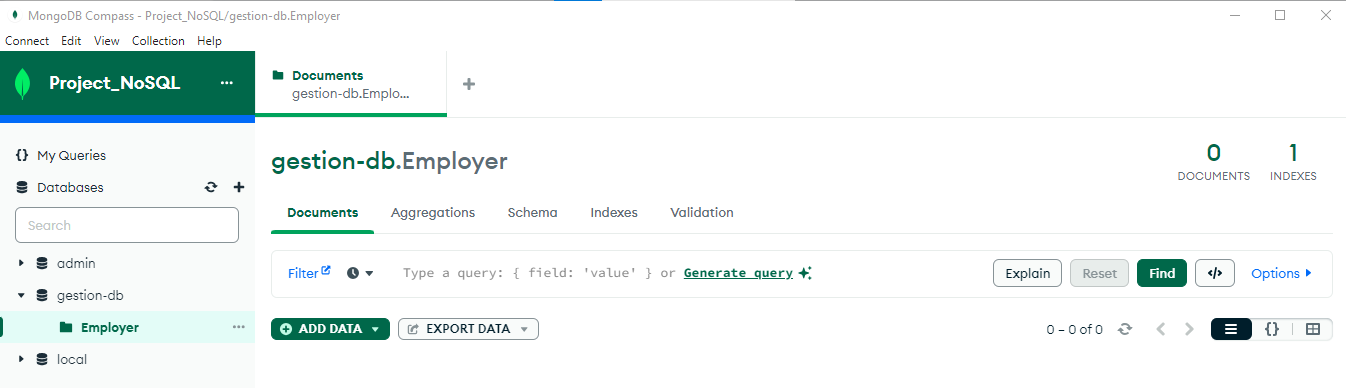
Deux solutions:

* Directement : db.createCollection(‘Employer’)
* En insérant un document: db.Employer.insert({empno : 1,ename:”helmi”,…})



1. **Lister les bases de données existantes:**





1. **Lister les collections existantes:**

On peut utiliser trois méthodes pour afficher les collections

* show collections , OU
* show tables , OU
* db.getCollectionNames()





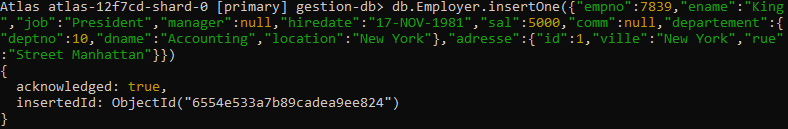


***L’insertion:***

Deux syntaxes :

* db.nomCollection.insert({clé1:’valeur1’,clé2:’valeur2’,…})
* ou bien obj=({clé1:’valeur1’,clé2:’valeur2’,…})

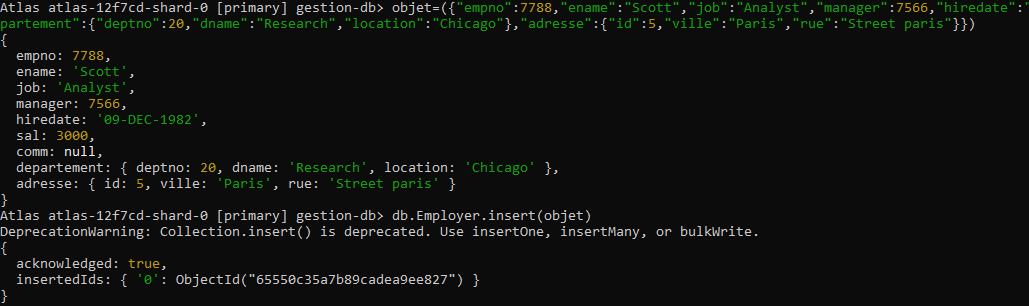
1. insertOne() :



1. insertMany() :

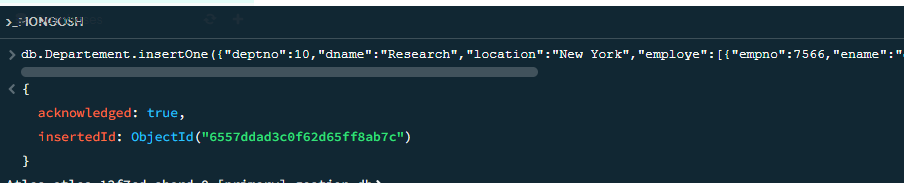


1. insert(objet)



On peut utiliser la collection department et insérer des documents:

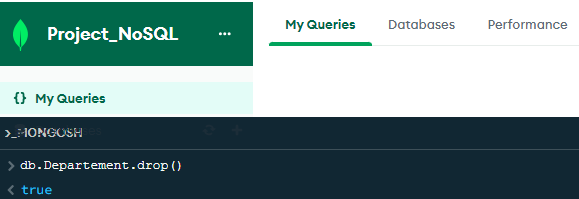
Exemple d’insertions d’un documents dans la collection department:





Pour supprimer une collection on tape :

db.nomCollection.drop():



Dans la suite en continuer le travail sur la collection employer:

***La sélection:***

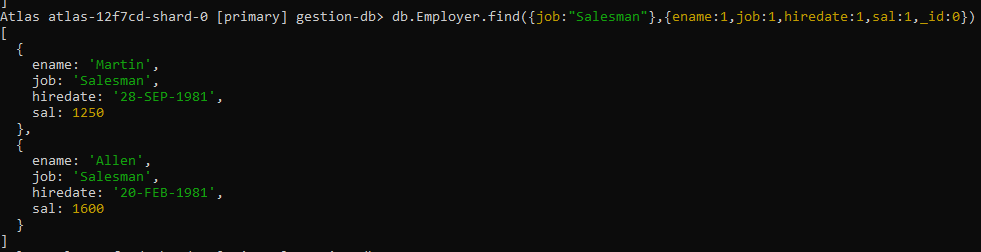
1. **Récupérer tous les documents d’une collection :**



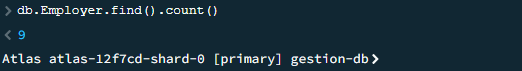




1. **Afficher seulment ename , job , hiredate et le salaire des employés dont leurs job est Salesman :**

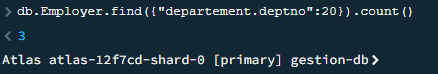


1. **Compter le nombre de documents:**

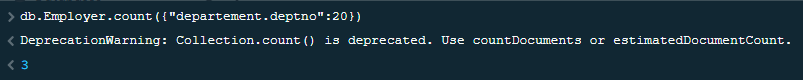


1. **Nombre des émployers qui travail dans le département numéro 20:**

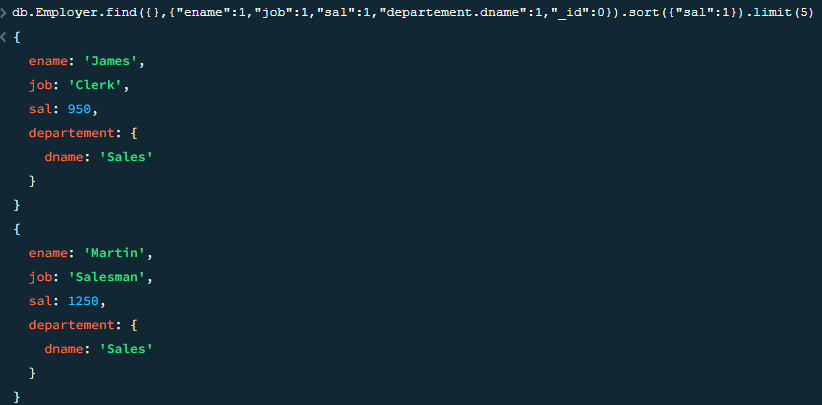
* Méthode1:



* Méthode2:



**10.Afficher seulement nom , poste , salaire et le nom du département selon l’ordre croissant du salaire en limittant l’affichage à 5 documents:**



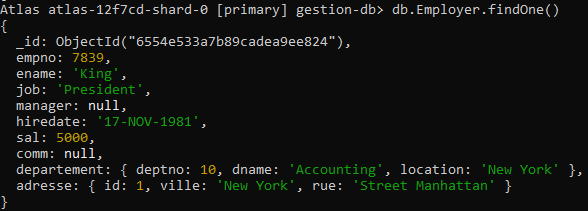




**11.Sauter quelque documents:**

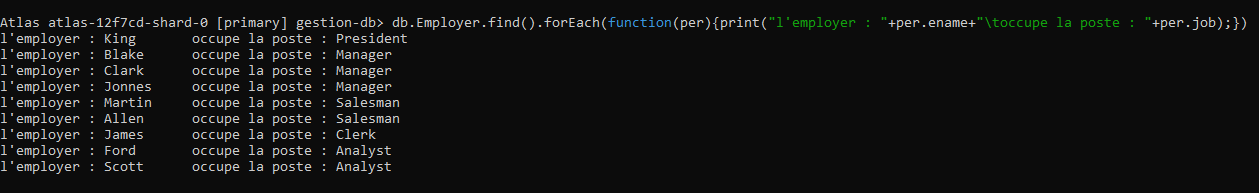


**12.Afficher seulement le premier documents:**

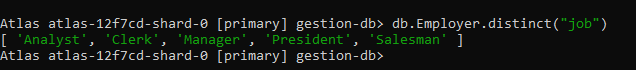


On peut aussi travailler avec find()[0] : db.Employer.find()[0]

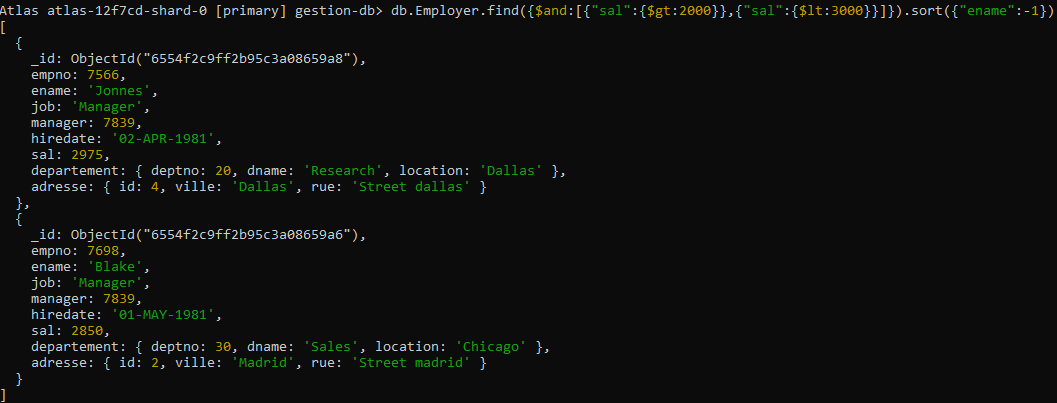
**13.Appeler une function pour chaque document de la selection:**



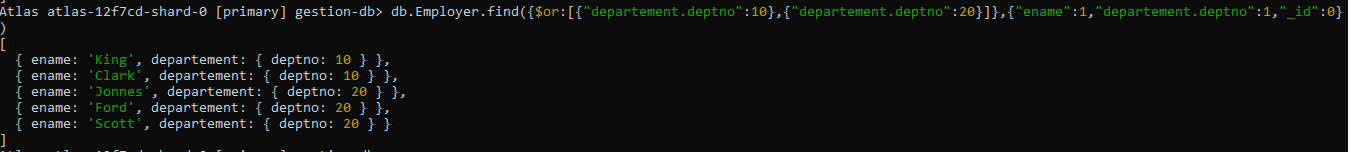
**14.Afficher les différents postes:**



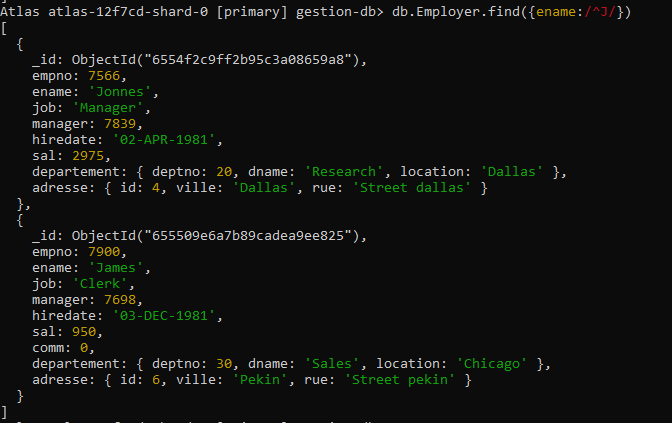
**15.Afficher les employés qui ayant une salaire entre 3000 et 2000 dans un ordre décroissant du nom:**



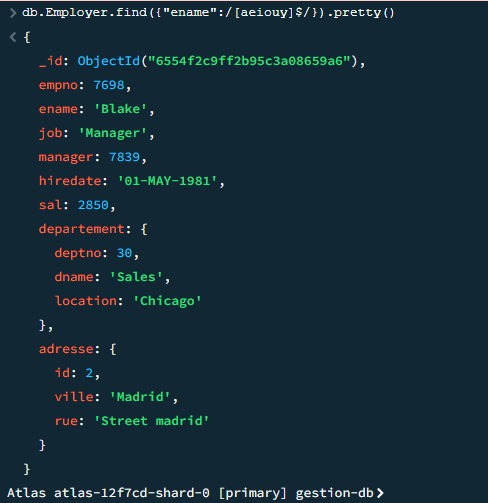
**16.Lister les noms et le numéro du département des employés qui travaillent dans les départements 10 ou 20 :**



**17.Chercher les employés dont le nom commence par “J”:**



**18.Chercher les employés dont le nom se termine par un voyelle avec un affichage sous format JSON:**



**19.Chercher les employés dont le nom commence par une lettre comprise entre [C-M] et leurs villes d’origine peuvent etre “Chicago” ou “Dallas”:**



***La modification :***

* update() : pour modifier un ou plusieurs documents selon une ou plusieurs conditions.
* updateOne() : pour modifier uniquement le premier enregistrement de la sélection.
* updateMany(): pour modifier plusieurs documents
* replaceOne(): pour remplacer le premier élément de la selection.

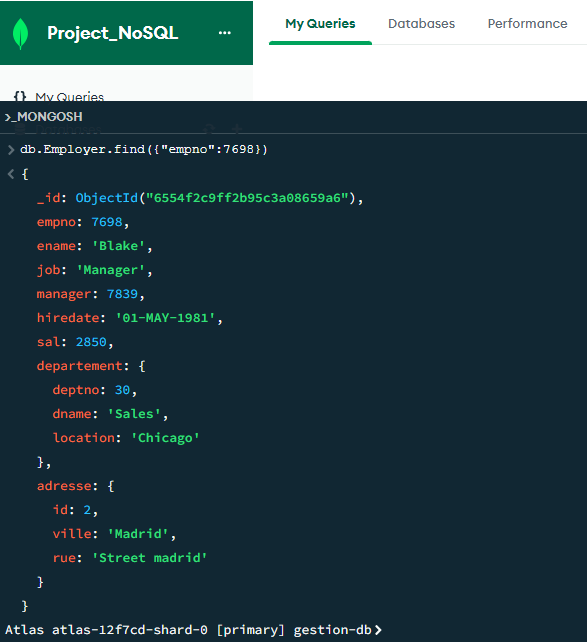
La méthode update():

db.nomCollection.update() prend au moins deux paramètres:

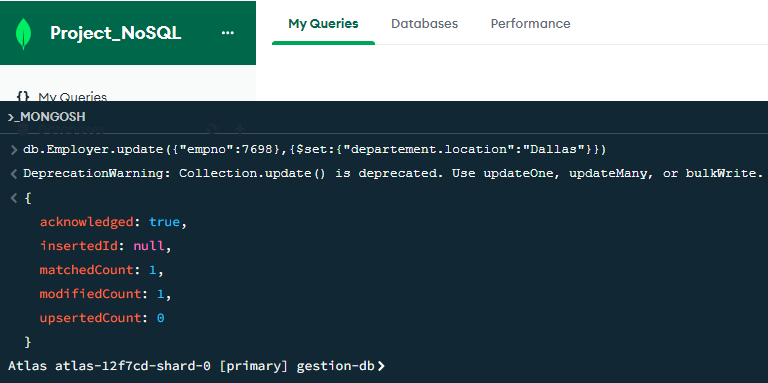
* le(s) élément(s) concerné(s) par la modification
* les modifications
* quelques options

**20.Modifier la localisation de l’employé numéro 7698 par “Dallas” :**

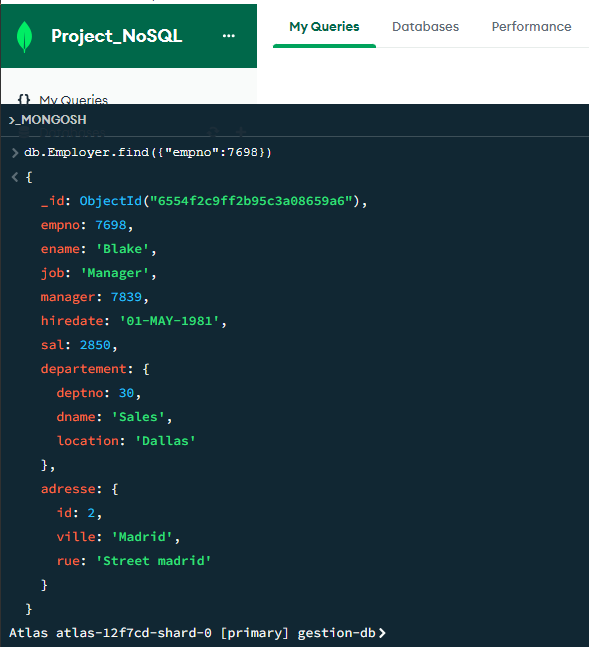
On affiche tous les informations sur ce employé



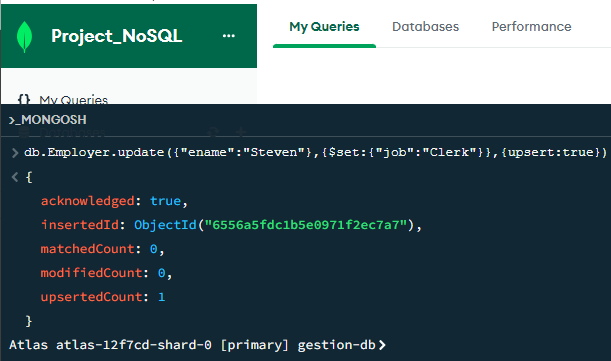
On faire la modification:

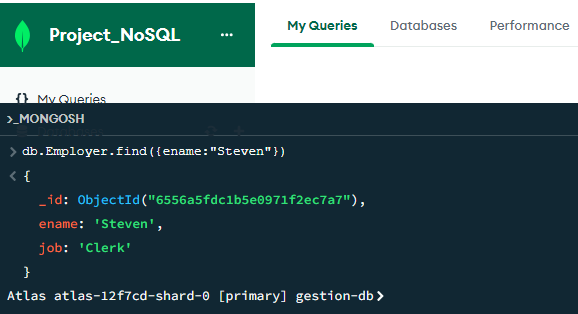


On l’affiche de nouveau :



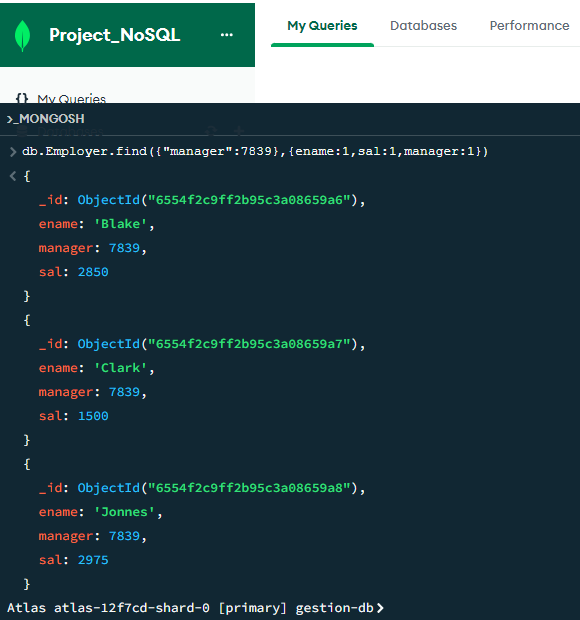
**21.On peut effectuer une insertion si aucun document ne remplit la condition en ajoutant upsert : true**



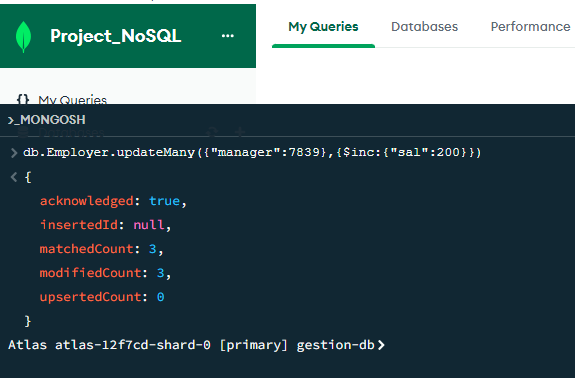


**22.Augmenter le salaire des employés dont leurs manager est 7839 par 200 :**

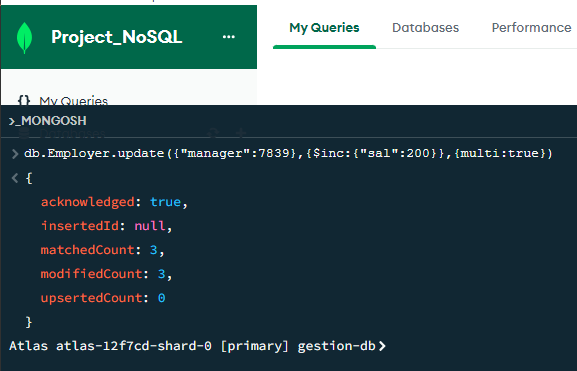
avant modification :



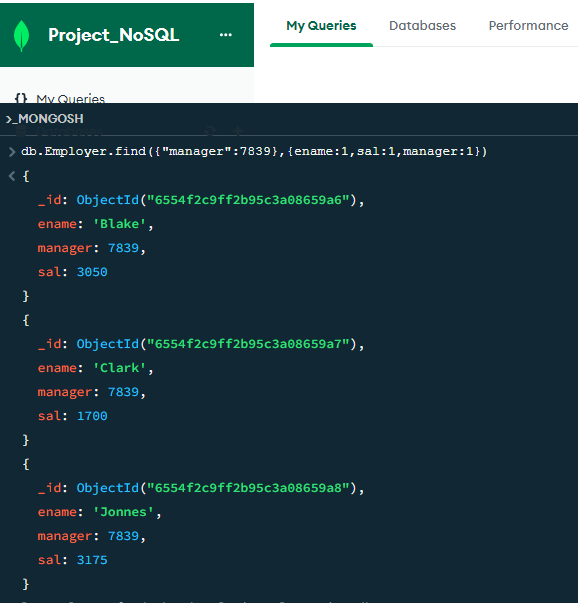
Modification méthode1:



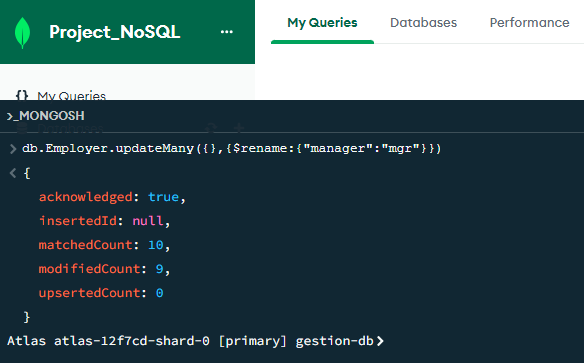
Modification méthode2:



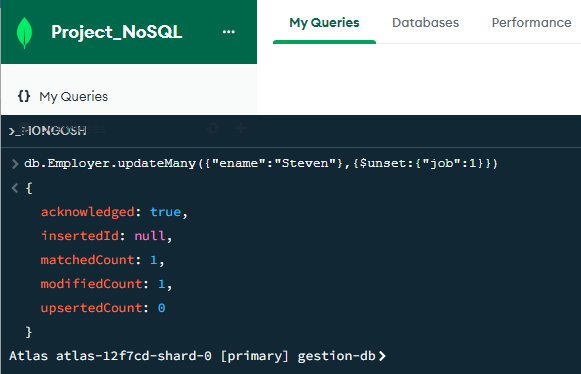
Après modification:

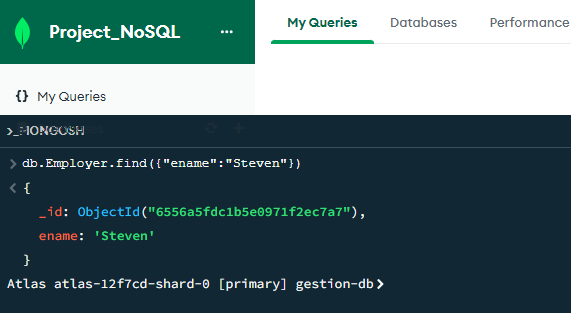


**23.Renommer le champ manager par mgr:**



**24.Supprimer la poste des employés qui ayant le nom “Steven”:**



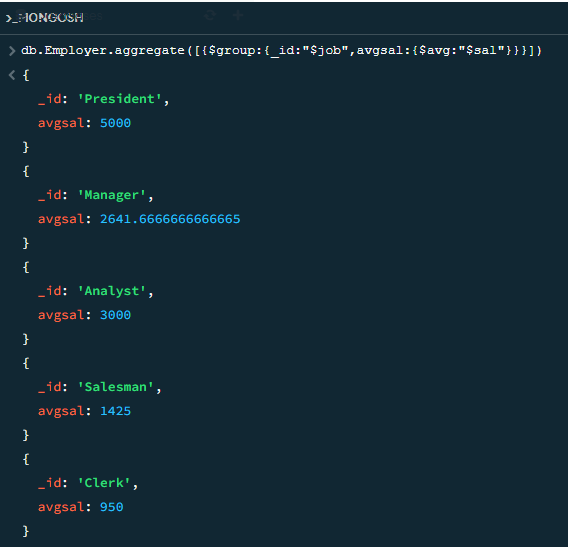


***Les agrégations:***

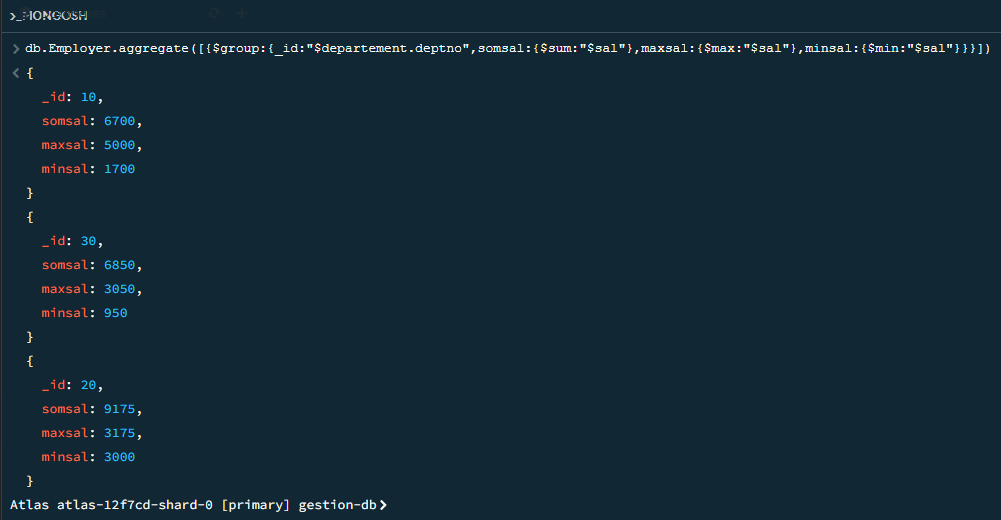
**25.Choisir Aléatoirement trois documents de la collections:**



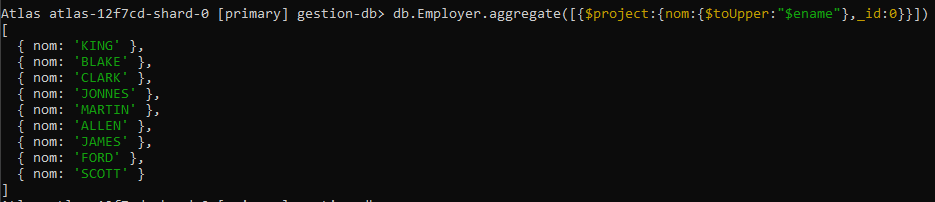
**26.Calculer le salaire moyenne pour chaque poste:**



**27.Donner la somme ,le minimum et le maximum des salaires pour chaque dépatement:**

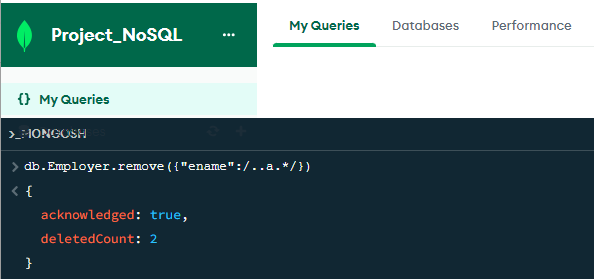


**28.Afficher les noms des employés du département en majuscule :**



***Suppression:***

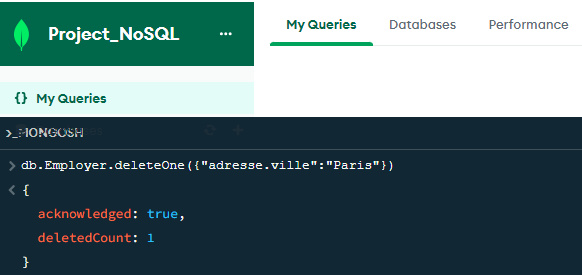
**29.Supprimer les employés dont le troisiéme caractére du nom est un “a”:**



Ou bien on utilise deleteMany():

db.Employer.deleteMany({“ename”:/..a.\*/})

**30.Supprimer l’employé qui habite dans la ville “Paris”:**



***Les Graphiques:***

