M. AZZOUZ

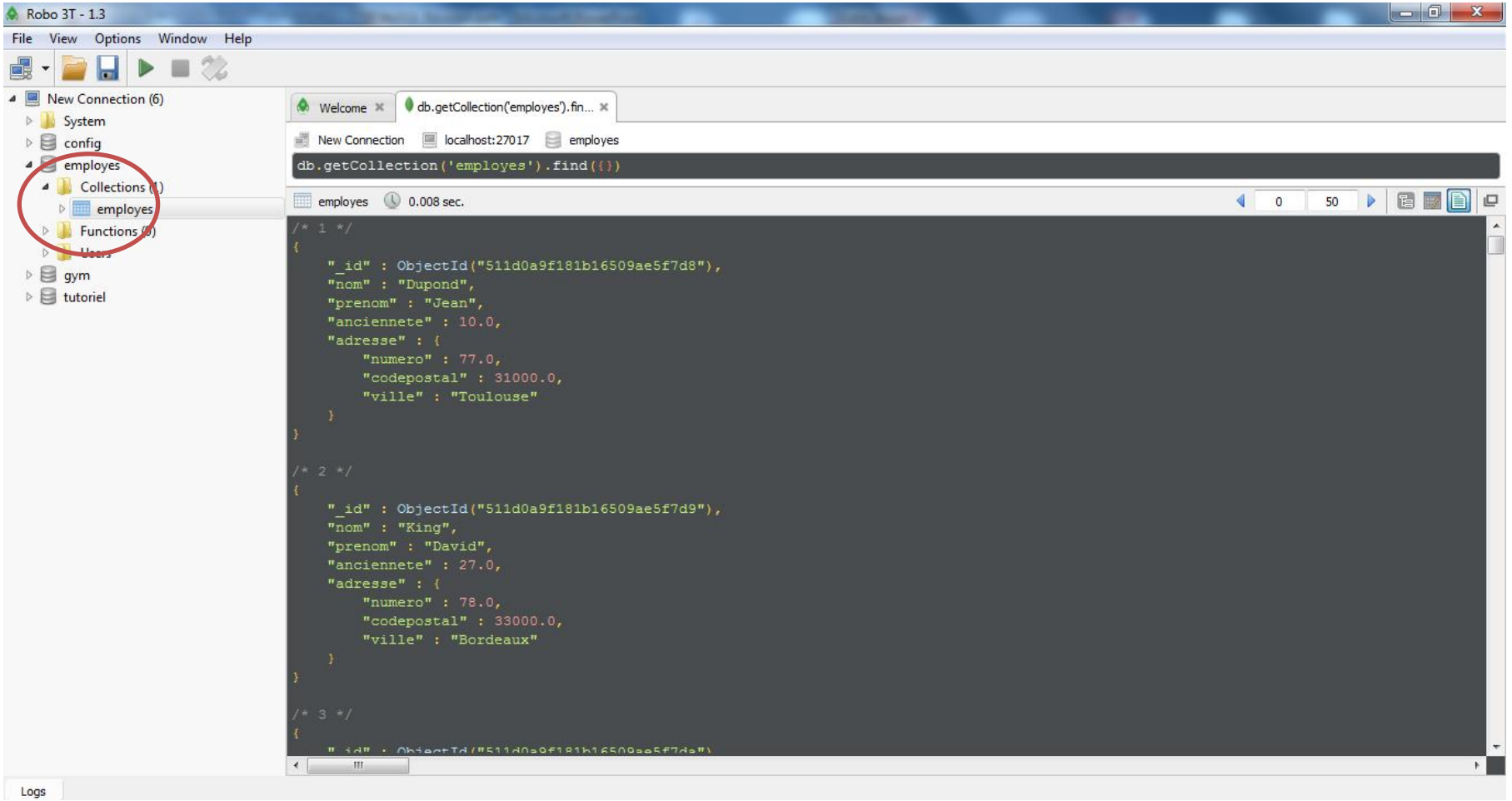
Bases de Données Avancées

TP Révision : MongoDB

– USTHB Master 01 IL–
M. AZZOUZ
Dernière mis à jour :Juillet 2020

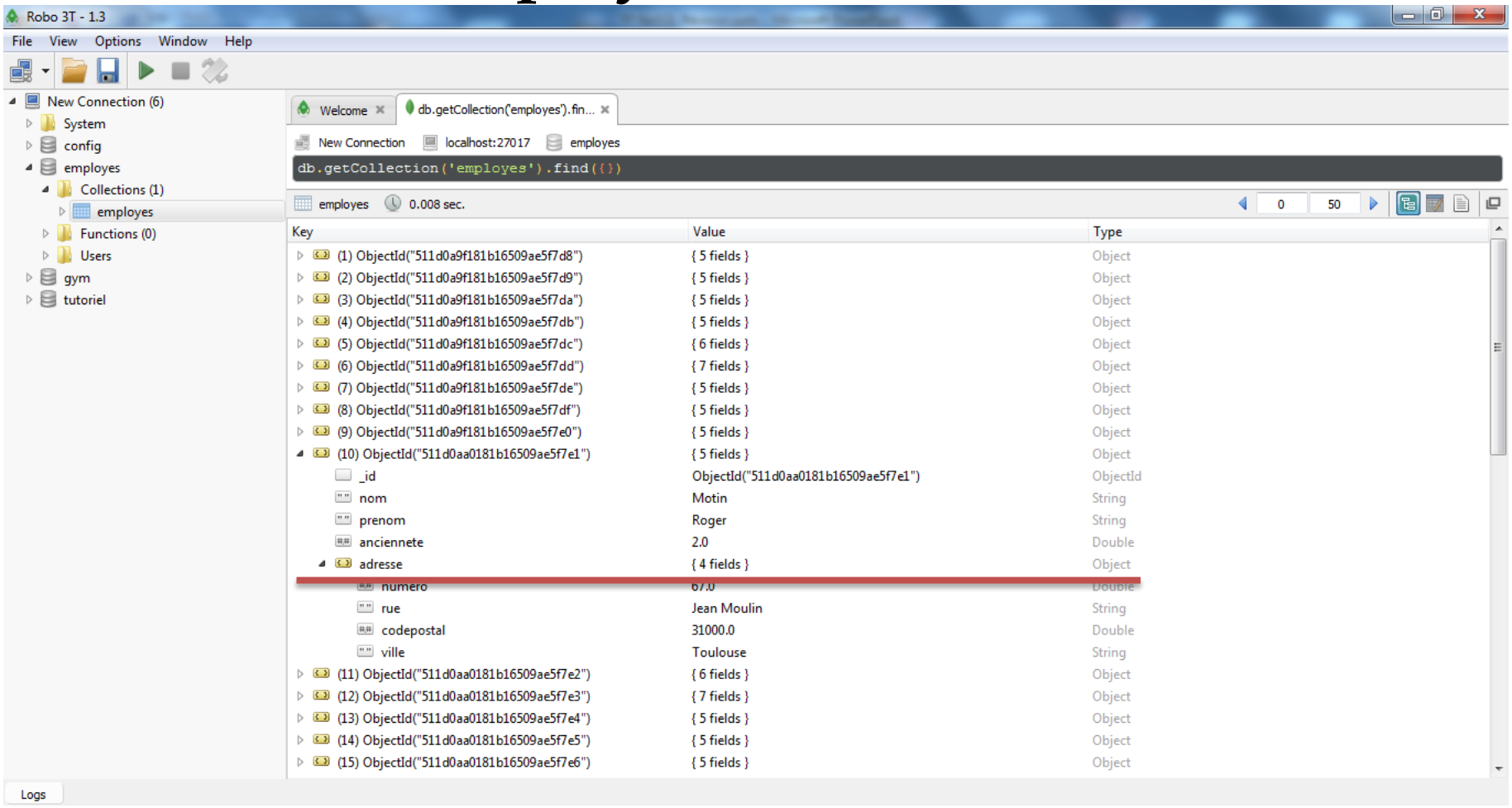
TP Révision

Vue d'ensemble sur la BD employe:



TP Révision

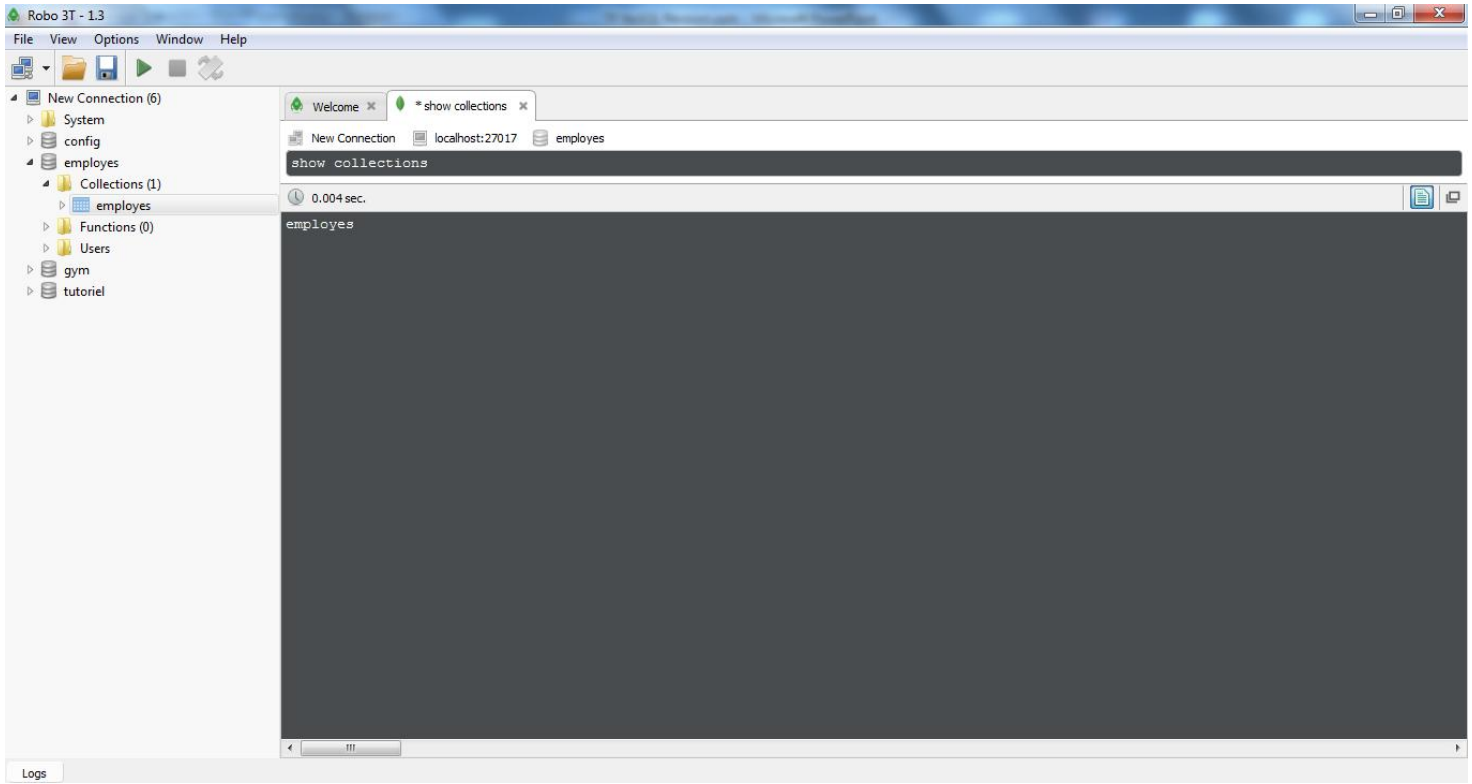
Collection employes:



TP Révision

a. **afficher toutes les collections de la base**

➢ **Réponse :** show collections



TP Révision

b. afficher tous les documents de la base

- **Réponse :**

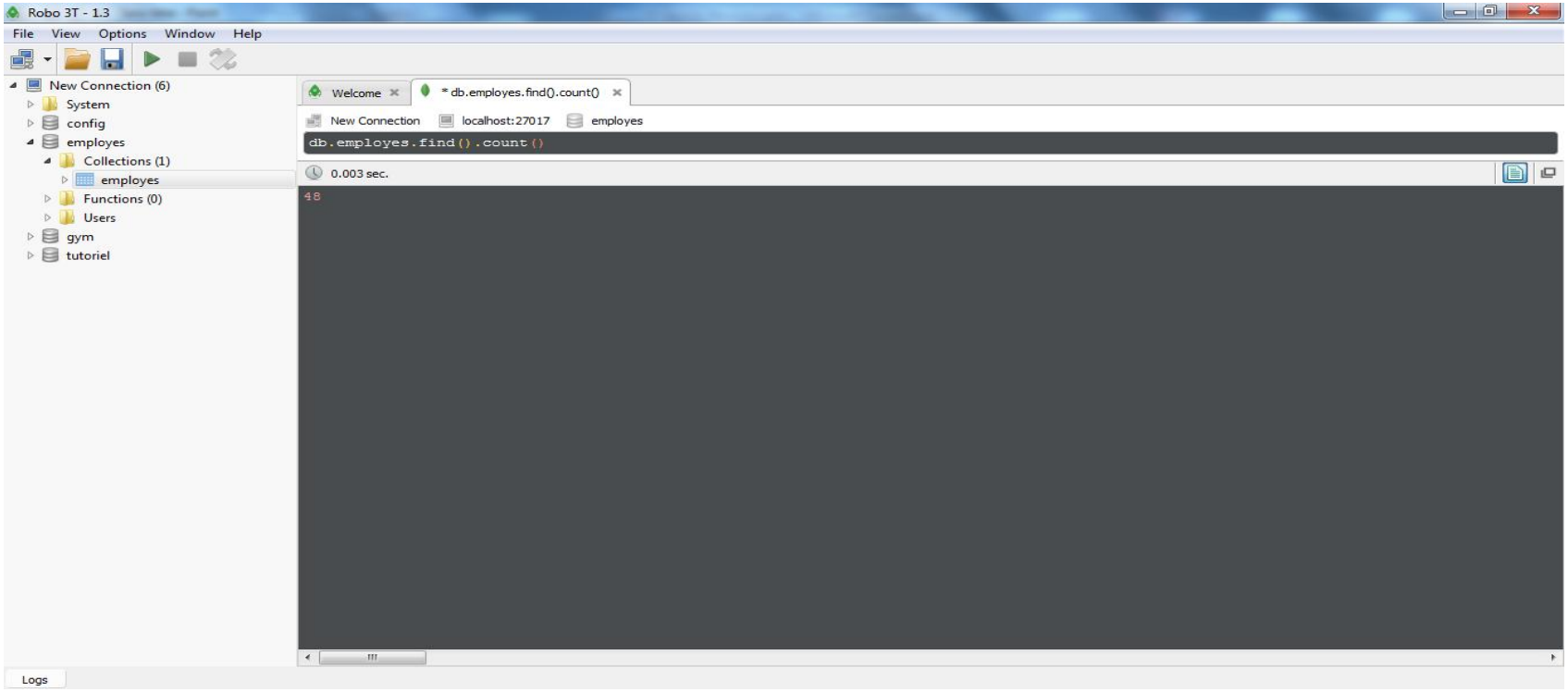
```
var coll = db.getCollectionNames()
for ( var i = 0; i < coll.length ; i ++)
{
db.getCollection(coll[i]).find()
}
```

TP Révision

c. compter le nombre de documents de la collection employes

➢ **Réponse :**

db.employes.find().count()

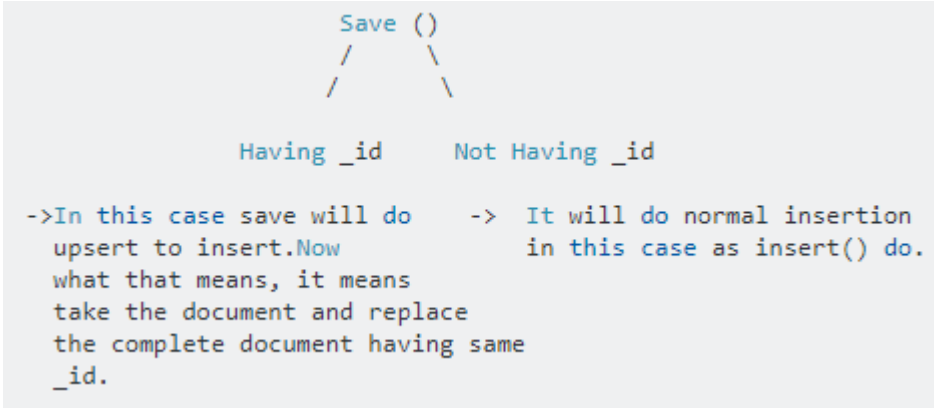


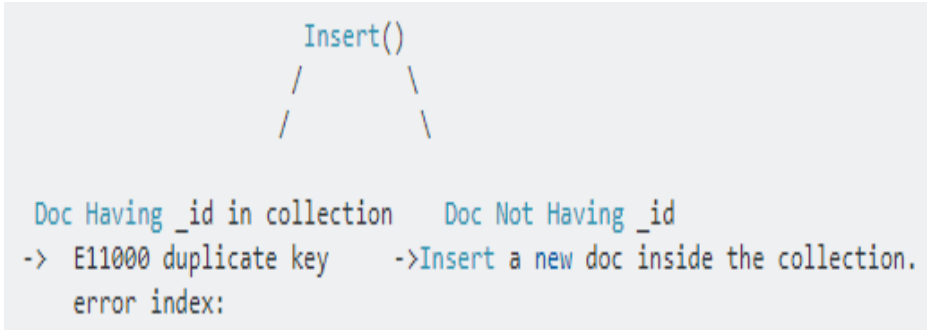
TP Révision

d. inserer de deux manières différentes deux employés avec les champs nom, prénom et soit prime soit ancienneté

➢ **Réponse :**

db.employes.**insert** ({ "nom" :"Alan","prenom" :"Joe", "anciennete" :10}) ;
db.employes.**save**({"nom":"Wick", "prenom" :"John", "prime" :150}) ;





TP Révision

e. **afficher la liste des employés dont le prénom est David**

- **Réponse :**

```
db.employes.find (
{ "prenom" :"David"},
{"_id":0});
```

TP Révision

f. afficher la liste des employés dont le prénom commence ou se termine par D

- **Réponse :**

db.employes.find({**$or**: [{ "prenom" :/^D/}, {"prenom" :/d$/}]})

Ou bien

db.employes.find({"prenom" :**/^D.*|.*d$/**})

TP Révision

g. afficher la liste des personnes dont le prénom commence par D et contient exactement 5 lettres

- **Réponse :**

```
db.employes.find({"prenom":/^D/}). forEach
(
function (p){
if(p.prenom.length == 5){
print (p)
}})
Ou bien
db.employes.find({"prenom":/^D[a-z]{4}$/});
```

TP Révision

h. afficher la liste des personnes dont le prénom commence et se termine par une voyelle

- **Réponse :**

```
db.employes.find(
{
"prenom": /^[AEIOUY].*[aeiouy]$/
 }) ;
```

TP Révision

i. **afficher la liste des personnes dont le prénom commence et se termine par une même lettre**

- **Réponse :**

```
db. employes.find().forEach(function (p){
pre = p.prenom.toLowerCase ();
if( pre[0] == pre[pre.length-1]){
print ( p )
}
})
```

TP Révision

j. afficher les nom et prénom de chaque employé ayant une ancienneté > 10

- **Réponse :**

```
db.employes.find (
{ "anciennete" :{ $gt :10}} ,

{ "_id" :0, "nom" :1, "prenom" :1}
);
```

TP Révision

k. afficher les nom et adresse complète des employés ayant un attribut rue dans l'objet adresse

- **Réponse :**

```
db. employes.find (

{"adresse.rue": { $exists : true } },

{ "nom" :1, "adresse" :1}
);
```

TP Révision

1. **incrémenter de 200 la prime des employés ayant déjà le champ prime**

➢ **Réponse :**

```
db.employes.updateMany (
{ "prime" :{ $exists : true }},
{ $inc :{ "prime" :200}});
```

Ou bien

```
db.employes.update (
{ "prime" :{ $exists : true }},
{ $inc :{ "prime" :200}} ,
{ multi : true })
```

TP Révision

m. afficher les trois premières personnes ayant la plus grande valeur d'ancienneté

- **Réponse :**

```
db.employes.find({"anciennete":
db.employes.aggregate([
{$group: {_id: null,
"anciennetemax": {$max:"$anciennete"}}}
]).next().anciennetemax
},
 {"_id":0}).limit (3) ;
```

TP Révision

n. regrouper les personnes dont la ville de résidence est Toulouse (afficher nom, prénom et ancienneté)

- **Réponse :**

```
db.employes.find (

{"adresse.ville": "Toulouse"},

{ "_id" :0,"nom" :1, "prenom" :1, "anciennete" :1}
);
```

TP Révision

o. **afficher les personnes dont le prénom commence par M et la ville de résidence est soit Foix soit Bordeaux**

- **Réponse :**

```
db. employes . find (
{"prenom" : /^M/,
"adresse.ville": {$in : ["Foix", "Bordeaux"]}
});
```

TP Révision

p. mettre à jour l'adresse de Dominique Mani : nouvelle adresse (numero : 20, ville : 'Marseille', codepostal : '13015'). Attention, il n'y aura plus d'attribut rue dans adresse

- **Réponse :**

```
db.employes.update (
{ "prenom" : "Dominique","nom" : "Mani"},
{$set : {"adresse.numero" : 20,
"adresse.ville" : "Marseille",
"adresse.codepostal" : "13015"},
$unset :{"adresse.rue" : 1}
});
```

TP Révision

q. **attribuer une prime de 1 500 à tous les employés n'ayant pas de prime et dont la ville de résidence est différente de Toulouse, Bordeaux et Paris.**

➢ **Réponse :**

```
db.employes.updateMany (

{"adresse.ville":{$nin:["Paris","Toulouse","Bordeaux"]},
"prime" :{ $exists : false }
},

{ $set :{ "prime" :1500}}
);
```

TP Révision

r. remplacer le champ téléphone, pour les documents ayant un champ tel), par un tableau nommé téléphones contenant la valeur du champ tel(le champ tel est à supprimer)

- **Réponse :**

```
db.employes.find ({ "tel" : { $exists : true }}).forEach (
function (p) {
db.employes.update(
{ "_id" :p._id },
{$push :{ "telephones" : p.tel },
$unset :{ "tel" :1}}
)
})
```

TP Révision

s. créer un champ prime pour les documents qui n'en disposent pas et de l'affecter à 100 * nombre de caractère du nom de la ville

- **Réponse :**

```
db.employes.find ({ "prime" : { $exists :false}}) . forEach
(function ( p ){
db.employes.update (
{ "_id" : p._id },
{$inc :{ "prime" : 100*p.adresse.ville.length }}
)});
```

TP Révision

t. **créer un champ mail dont la valeur est égale soit à nom.prénom@formation.fr pour les employés ne disposant pas d'un champ téléphone, soit à prénom.nom@formation.fr (nom et prénom sont à remplacer par les vraies valeurs de chaque employé)**

- **Réponse :**

```
db.employes.find().forEach (
function (p) {
email = p.nom + '.' + p.prenom + '@formation.fr ';
if (p.telephones)
email = p.prenom + '.' + p.nom + '@formation.fr ';
db.employes.update(
{ "_id" :p._id },
{ $set :{ "mail" : email }})
});
```

TP Révision

u. **calculer et afficher la somme de l'ancienneté pour les employés disposant du même prénom**

- **Réponse :**

```
db.employes.aggregate(
{ $group : {"_id" : "$prenom",
"ancienneteCum" : { $sum : "$anciennete"}}},
{ $sort : { "_id" : 1}}
);
```