TP n°2 : Requêtes – Framework d'agrégation

Soit « salles » la collection construite de la manière suivante :

```
db.salles.insertMany([
{"_id": 1,
 "nom": "AJMI Jazz Club",
 "adresse": {"numero": 4, "voie": "Rue des Escaliers Sainte-Anne",
             "codePostal": "84000","ville": "Avignon",
             "localisation": {"type": "Point",
                              "coordinates": [43.951616, 4.808657]}
            },
"styles": ["jazz", "soul", "funk", "blues"],
"avis": [{"date": new Date('2024-11-01'),
          "note": NumberInt(8)},
         {"date": new Date('2024-11-30'),
          "note": NumberInt(9)}],
"capacite": NumberInt(300),
"smac":true
},
{"_id": 2,
 "nom": "Paloma",
 "adresse": {"numero": 250, "voie": "Chemin de l'aérodrome",
             "codePostal": "30000","ville": "Nîmes",
             "localisation": {"type": "Point",
                              "coordinates": [43.856430,4.405415]}
            },
"avis": [{"date": new Date('2024-07-06'),
          "note": NumberInt(10)}],
"capacite": NumberInt(4000),
"smac":true
},
{"_id": 3,
 "nom": "Sonograf",
 "adresse": {"voie": "D901",
             "codePostal": "84250","ville": "Le Thor",
             "localisation": {"type": "Point",
                              "coordinates": [43.923005,5.020077]}
            },
"capacite": NumberInt(200),
"styles": ["blues", "rock"],
}
])
```

- fichier « salles.json » disponible sur celene

Exercice 1

Écrivez le *pipeline* qui affichera dans un champ nommé `ville` le nom de celles abritant une salle de plus de 50 personnes ainsi qu'un booléen nommé grande qui sera positionné à la valeur « vrai » lorsque la salle dépasse une capacité de 1 000 personnes. Voici le squelette du code à utiliser dans le *shell* :

```
pipeline = [ ...
]
db.salles.aggregate(pipeline)
```

- *Exercice 2*

Écrivez le *pipeline* qui affichera dans un champ nommé `apres_extension` la capacité d'une salle augmentée de 100 places, dans un champ nommé `avant_extension` sa capacité originelle, ainsi que son nom.

- *Exercice 3*

Écrivez le *pipeline* qui affichera, par numéro de département, la capacité totale des salles y résidant. Pour obtenir ce numéro, il vous faudra utiliser l'opérateur `$substrBytes` dont la syntaxe est la suivante :

```
{$substrBytes: [ < chaîne de caractères >, < indice de départ >, < longueur > ]}
```

- *Exercice 4*

Écrivez le *pipeline* qui affichera, pour chaque style musical, le nombre de salles le programmant. Ces styles seront classés par ordre alphabétique.

- *Exercice 5*

À l'aide des *buckets*, comptez les salles en fonction de leur capacité :

- celles de 100 à 500 places
- celles de 500 à 5000 places

- *Exercice 6*

Écrivez le *pipeline* qui affichera le nom des salles ainsi qu'un tableau nommé avis_excellents qui contiendra uniquement les avis dont la note est de 10.

- *Exercice 7*

Écrivez le pipeline qui affichera le nombre total de salles pour chaque ville. Les résultats devront être classés par ordre décroissant du nombre de salles.

- *Exercice 8*

Créez un pipeline pour afficher, par style musical, le nombre de salles le programmant. Utilisez *$sortByCount* pour afficher les styles du plus populaire au moins populaire.

- *Exercice 9*

Écrivez le pipeline qui renvoie les salles dont la note moyenne des *avis* dépasse *8*. Affichez le nom de chaque salle et sa note moyenne dans un champ moyenne.

- *Exercice 10*

Écrivez le pipeline qui affiche uniquement les salles dont la capacité est inférieure à *200* places. Affichez uniquement le nom et la capacité de ces salles.

- *Exercice 11*

Écrivez un pipeline qui ajoute un champ nommé taille avec la valeur *"petite"* pour les salles de moins de 100 places, *"moyenne"* pour les salles entre 100 et 500 places, et *"grande"* pour les salles de plus de 500 places.

- *Exercice 12*

Créez un pipeline pour afficher la capacité moyenne des salles de chaque ville. Utilisez *$group* pour agréger par ville et calculez la capacité moyenne avec *$avg.*

- *Exercice 13*

Écrivez le pipeline qui affiche, pour chaque salle, le nom de la salle, et un tableau *mauvais_avis* qui contient uniquement les avis dont la note est inférieure ou égale à 3.

- *Exercice 14*

Utilisez *$bucket* pour regrouper les salles en trois catégories : celles de moins de 100 places, celles entre 100 et 500 places, et celles de plus de 500 places. Affichez le nombre de salles pour chaque catégorie.

- *Exercice 15*

Écrivez le pipeline qui affiche le nom des salles et un champ *avis_moyen_dernier_mois* qui calcule la note moyenne des avis publiés uniquement au cours des 30 derniers jours.

- *Exercice 16*

Écrivez un pipeline qui affiche pour chaque ville le style de musique le plus populaire (c'est-à-dire celui programmé dans le plus de salles) et le nombre total de salles programmant ce style.

- *Exercice 17 (difficile)*

Écrivez un pipeline qui affiche pour chaque ville le style de musique le plus populaire (celui programmé dans le plus de salles) et le style avec la meilleure note moyenne des avis. Si une salle n'a pas de notes, elle est ignorée pour le calcul de la meilleure note moyenne.

- *Exercice 18*

Écrivez un pipeline qui affiche la note moyenne des avis des six derniers mois, la note la plus élevée de cette période, et déterminer si la salle a une "très bonne note" (true si la note moyenne est supérieure ou égale à 8).