Exercice MongoDB :

Exercice 1 :

1. Créez une nouvelle base de données nommée `DBProduits`
2. Créez une nouvelle collection nommée `produits` et y insérer le document suivant:

- nom: Macbook Pro
- fabriquant: Apple
- prix: 1299.99
- options:
  - Intel Core i5
  - Retina Display
  - Long life battery

Rajoutez un autre document :

- nom: Macbook Air
- fabriquant: Apple
- prix: 1099.99
- ultrabook: true
- options:
  - Intel Core i7
  - SSD
  - Long life battery

Et enfin un dernier document:

- nom: Thinkpad X230
- fabriquant: Lenovo
- prix: 999.99
- ultrabook: true
- options:
  - Intel Core i5
  - SSD
  - Long life battery

3. Effectuez les requêtes de lecture suivantes:

1. Récupérez tous les produits.
2. Récupérez le premier produit
3. Trouvez l'`id` du Thinkpad et faites la requête pour récupérer ce produit avec son `id`.
4. Récupérez les produits dont le `prix` est supérieur à 1200$
5. Récupérez le premier produit ayant le champ `ultrabook` à `true`
6. Récupérez le premier produit dont le `nom` contient Macbook
7. Récupérez les produits dont le `nom` commence par Macbook
8. Supprimez les deux produits dont le `fabricant` est Apple.
9. Supprimez le Lenovo X230 en utilisant uniquement son `id`.

Exercice 2 :

Dans cet exercice, nous allons modéliser un système de facturation très simple.

Créer la collections Factures avec deux document:

Facture numéro `10012A`:

- Date de facture: 2013-07-04
- Client:
    - Nom: Alexandre Terrasa
    - Courriel: foobar@example.com
- Liste des produits (2 produits):
    - Code: MACBOOKAIR
    - Nom: Macbook Air
    - Prix: 999.99
    - Quantite: 1
    - Code: APPLESUPPORT
    - Nom: AppleCare 1 an
    - Prix: 149.99
    - Quantite: 1
- Total: 1149.98

Facture numéro `10013A`:

- Date de facture: 2013-07-05
- Client:
    - Nom: Jacques Berger
    - Courriel: berger.jacques@uqam.ca
- Liste des produits (1 produit) :
    - Code: LENOVOX230
    - Nom: Lenovo Thinkpad X230
    - Prix: 899.99
    - Quantite: 1
- Total : 899.99

Effectuez les actions suivantes:

1. Récupérer la facture avec le numéro est `10013A`
2. Modifier la facture `10012A` en changeant la date pour le 2013-07-03 et le courriel du contact pour alex@example.com
3. Récupérer la facture avec le produit vendu ayant un code `LENOVOX230`
4. Supprimer la facture `10012A`

Exercice 3 :

Créez une nouvelle base de données nommée `DBRestaurants`

Créez la collection `Restaurants`

1. Écrivez une requête MongoDB pour afficher tous les documents des restaurants de la collection.

2. Écrivez une requête MongoDB pour afficher les champs restaurant_id, name, borough et cuisine pour tous les documents de la collection restaurant.

3. Écrivez une requête MongoDB pour afficher les champs restaurant_id, name, borough et cuisine, mais excluez le champ _id pour tous les documents de la collection restaurant.

4. Écrivez une requête MongoDB pour afficher les champs restaurant_id, name, borough et zip code, mais excluez le champ _id pour tous les documents de la collection restaurant.

5. Écrivez une requête MongoDB pour afficher tous les restaurants qui se trouvent dans borough du Bronx.
6. Écrivez une requête MongoDB pour afficher les 5 premiers restaurants qui se trouvent dans borough Bronx.

7. Écrivez une requête MongoDB pour trouver les restaurants qui ont atteint un score supérieur à 80 mais inférieur à 100.

8. Écrivez une requête MongoDB pour trouver les restaurants dont la latitude est inférieure à -95,754168.

9. Écrivez une requête MongoDB pour trouver les restaurants qui ne préparent aucune cuisine "américaine" et dont la score est supérieure à 70 et la latitude inférieure à -65,754168.

10. Écrivez une requête MongoDB pour trouver les restaurants qui ne préparent aucune cuisine 'Américaine' et ont obtenu un score supérieur à 70 et situés dans la longitude inférieure à -65.754168.

11. Écrivez une requête MongoDB pour trouver les restaurants qui ne préparent aucune cuisine 'américaine' et qui ont obtenu grade 'A' n'appartenant pas à borough de Brooklyn. Le document doit être affiché la cuisine dans l'ordre décroissant.

12. Écrivez une requête MongoDB pour trouver l'identifiant, le nom, borough et la cuisine du restaurant pour les restaurants qui contiennent "Wil" comme trois premières lettres de son nom.
13. Écrivez une requête MongoDB pour trouver l'ID, le nom, borough et la cuisine du restaurant pour les restaurants qui contiennent "ces" comme trois dernières lettres de son nom.

14. Écrivez une requête MongoDB pour trouver l'identifiant, le nom, borough et la cuisine du restaurant pour les restaurants qui contiennent "Reg" en trois lettres quelque part dans son nom.

15. Écrivez une requête MongoDB pour trouver les restaurants qui appartiennent à borough du Bronx et préparent des plats américains ou chinois.

16. Écrivez une requête MongoDB pour trouver l'identifiant, le nom, l'arrondissement et la cuisine du restaurant pour les restaurants appartenant à l'arrondissement Staten Island ou Queens ou Bronxor Brooklyn.

17. Écrivez une requête MongoDB pour trouver l'ID, le nom, l'arrondissement et la cuisine du restaurant pour les restaurants qui n'appartiennent pas à l'arrondissement Staten Island ou Queens ou Bronxor Brooklyn.

18. Écrivez une requête MongoDB pour trouver l'identifiant, le nom, l'arrondissement et la cuisine du restaurant pour les restaurants qui ont obtenu un score inférieur ou égal à 10.

19. Écrivez une requête MongoDB pour trouver l'identifiant, le nom, l'arrondissement et la cuisine du restaurant pour les restaurants qui ont préparé cuisine sauf « Américain » et « Chinees » ou le nom du restaurant commence par la lettre « Wil ».

20. Écrivez une requête MongoDB pour trouver l'identifiant, le nom et les notes du restaurant pour les restaurants qui ont obtenu une grade de « A » et un score de 11 sur un ISODate « 2014-08-11T00:00:00Z » parmi de nombreuses dates d'enquête.

21. Écrivez une requête MongoDB pour trouver l'identifiant, le nom et les grades du restaurant pour les restaurants où le 2ème élément du tableau des grades contient une grade de "A" et un score de 9 sur un ISODate "2014-08-11T00:00:00Z".

22. Écrivez une requête MongoDB pour trouver l'identifiant, le nom, l'adresse du restaurant pour les restaurants où le 2ème élément du tableau coord contient une valeur supérieure à 40 et jusqu'à 52.

23. Écrivez une requête MongoDB pour trier le nom des restaurants par ordre croissant avec toutes les colonnes.

24. Écrivez une requête MongoDB pour organiser le nom des restaurants en ordre décroissant avec toutes les colonnes.

25. Écrivez une requête MongoDB pour organiser le nom de la cuisine dans l'ordre croissant et pour ce même arrondissement ,cuisine devrait être dans l'ordre décroissant.

26. Écrivez une requête MongoDB pour savoir si toutes les adresses contiennent la rue ou non.

27. Écrivez une requête MongoDB qui sélectionnera tous les documents de la collection de restaurants où la valeur du champ coord est Double.

28. Écrivez une requête MongoDB qui sélectionnera l'identifiant, le nom et les notes du restaurant pour les restaurants qui renvoient 0 comme reste après avoir divisé le score par 7.

29. Écrivez une requête MongoDB pour trouver le nom du restaurant, l'arrondissement, la longitude et l'attitude et la cuisine pour les restaurants qui contiennent « mon » en trois lettres quelque part dans son nom.

30. Écrivez une requête MongoDB pour trouver le nom du restaurant, l'arrondissement, la longitude et la latitude et la cuisine pour les restaurants qui contiennent "Mad" comme trois premières lettres de son nom.

EX-4

EX4

Exprimez des requêtes simples pour les recherches suivantes :

1. Liste de tous les livres (type « Book ») ;
2. Liste des publications depuis 2011 ;
3. Liste des livres depuis 2014 ;
4. Liste des publications de l’auteur « Toru Ishida » ;
5. Liste de tous les éditeurs (type « publisher »), distincts ;
6. Liste de tous les auteurs distincts ;
7. Trier les publications de « Toru Ishida » par titre de livre et par page de début ;
8. Projeter le résultat sur le titre de la publication, et les pages ;

9. Compter le nombre de ses publications ;
10. Compter le nombre de publications depuis 2011 et par type ;
11. Compter le nombre de publications par auteur et trier le résultat par ordre croissant

---

EX1;

1,db.produits.find();

2,db.produits.findOne();

3,ID.THk = db.produits.find({nom : "ThinkpadX230"})

db.produits.find({_id:ID.THk});

4,prx = {"prix":{"$gt":1200}}

db.produits.find(prx,{});

5,db.produits.findOne({"ultrabook":true},{});

6,db.produits.findOne({"nom":/Macbook/},{});

7,db.produits.findOne({"nom":/^Macbook/},{});

8,db.produits.deleteMany({"fabriquant":"Apple",2});

9,db.produits.deleteOne({_id:ObjectId("63762f1b6848814fd8cdb41a")});

EX2;

1,db.factures.find({"_id":"10013A"});

2,db.factures.updateOne({"_id":"10012A"},{$set:{"Date de facture":"2013-07-03","Client.Courriel":"alex@exampl.com"}});

3,db.factures.find({"Liste des produits.Code":/LENOVOX230/});

4,db.factures.deleteOne({_id:"10012A"});

EX3;

1,db.restaurants.find();

2,db.restaurants.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine":1});

3,db.restaurants.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine":1,"_id":0});

4,db.restaurants.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine":1,"address.zipcode":1,"_id":0});

5,db.restaurants.find({"borough":"Bronx"},{});

6,db.restaurants.find({"borough":"Bronx"},{}).limit(5);

7,db.restaurants.find({grades:{$elemMatch:{score:{$gt:80,$lt:100}}}});

8,lat = {"address.coord":{"$lt":-95.754168}}

db.restaurants.find(lat,{});

9,db.restaurants.find({"grades.score":{$gt:70},"address.coord":{"$lt":-65.754168},"cuisine":{"$ne":"American "}});

10,=9,

11,db.restaurants.find({"grades.grade":{$eq:"A"},"borough":{"$ne":"Brooklyn"},"cuisine":{"$ne":"American "}},{"cuisine":1}).sort({"cuisine":-1});

12,db.restaurants.find({"name":/^Wil/},{});

13,db.restaurants.find({"name":/ces$/},{});

14,db.restaurants.find({"name":/Reg/},{});

15,db.restaurants.find({"borough":"Bronx","cuisine": {"$in" : ["American ","Chinese"]}},{});

16,db.restaurants.find({"borough": {"$in" : ["Staten Island","Queens","Bronx","Brooklyn"]}},{});

17,db.restaurants.find({"borough": {"$nin" : ["Staten Island","Queens","Bronx","Brooklyn"]}},{});

18,db.restaurants.find({grades:{$elemMatch:{score:{$lte:10}}}});

19,db.restaurants.find({"$or" : [{"cuisine": {"$nin" : ["American ","Chinese"]}},{"name":/^Wil/}]},{});

20,db.restaurants.find({"$and" :
[{grades:{$elemMatch:{score:{$eq:11}}}},{grades:{$elemMatch:{grade:{$eq:"A"}}}},{grades:{$elemMatch:{date:{$eq:ISO
Date("2014-03-03T00:00:00.000+0000")}}}}]},{});

21,db.restaurants.find({"$and" :
[{"grades.1.score":{$eq:9}},{"grades.1.grade":{$eq:"A"}},{"grades.1.date":{$eq:ISODate("2014-03-
03T00:00:00.000+0000")}}]},{});

22,db.restaurants.find({"address.coord.1":{$gt:40,$lte:52}},{"name":1,"address":1});

23,db.restaurants.find().sort({"name":1});

24,db.restaurants.find().sort({"name":-1});

25,db.restaurants.find().sort({"cuisine":1,"borough":-1,});

26,db.restaurants.find({"address.street":{$exists:true}});

27,db.restaurants.find({"address.coord":{$type:"double"}});

28,db.restaurants.find({"grades.score":{$mod:[7,0]}},{"restaurant_id":1,"name":1,"grades":1});

29,db.restaurants.find({"name":/mon/},{"name":1,"borough":1,"address.coord":1,"cuisine":1});

30,db.restaurants.find({"name":/^Mad/},{"name":1,"borough":1,"address.coord":1,"cuisine":1});

EX4;

1,db.publis.find({"type":"Book"},{});

2,db.publis.find({"year":{$gte:2011}},{});

3,db.publis.find({"year":{$gte:2014},"type":"Book"},{});

4,db.publis.find({"authors" : "Toru Ishida"},{});

5,db.publis.distinct("editor",{"publisher" : {$exists : "true"}});

6,db.publis.distinct("authors");

7,db.publis.find({"authors" : "Toru Ishida"},{}).sort({"title":1,"pages":1});

pymongo : db.publis.find({"authors" : "Toru Ishida"},{}).sort("title",1).sort("pages",1);

8,db.publis.find({},{"title":1,"pages":1});

9,db.publis.count();

10,db.publis.aggregate([{$match:{"year":{"$gte":2011}}},{$group: {_id: "$type", total : {$sum:1}}}]);

11,db.publis.aggregate([{$group: {_id: "$authors", total : {$sum:1}}}]).sort({"authors":1})