```
////////////////////////////////////////////////////////////////////////////
// Exercice 1 :                                                           //
////////////////////////////////////////////////////////////////////////////

//Collection niveau
{
  id_niveau: "DSI3",
  annee: 3,
  specialite: "DSI"
}
//Collection matiere
{
  code_matiere: "GDM",
  nom_matiere: "Gestion des Données Massives",
  type: "Cours intégré"
}
//Collection plan_etude
{
  id_niveau: "DSI3",
  code_matiere: "DBA",
  coefficient: 1.5,
  volume_semaine: 3
}

//Solution dénormalisée : Imbrication des matières dans niveau (les proprietes de
l'association aussi)
{
    id_niveau: "DSI3",
    annee: 3,
    specialite: "DSI",
    matieres: [
        {
          code_matiere: "DBA",
          nom_matiere: "Base de Données Avancées",
          type: "Atelier",
          coefficient: 1.5,
          volume_semaine: 3
        },
        {
          code_matiere: "GDM",
          nom_matiere: "Gestion des Données Massives",
          type: "Cours intégré",
          coefficient: 1,
          volume_semaine: 1.5
        },
        {
          code_matiere: "POO",
          nom_matiere: "Développement Mobile",
          type: "Atelier",
          coefficient: 1.5,
          volume_semaine: 3
        }
    ]
}
```

```
//Q2.a
// Il manque les information le coefficient, le nom et le code de la matière
db.niveau.insertOne({
    id_niveau: "DSI3",
    annee: 3,
    specialite: "DSI"
});
db.matiere.insertOne({
    code_matiere: "GDM",
    nom_matiere: "Gestion des Données Massives",
    type: "Cours intégré"
});
db.plan_etude.insertOne({
    id_niveau: "DSI3",
    code_matiere: "GDM",
    coefficient: 1.5,
    volume_semaine: 1.5
});

//Q2.b
db.niveau.aggregate([
    //jointure avec plan_eude pour récupérer les code_matieres dans un tableau plan
    {$lookup:{from:"plan_etude", localField:"id_niveau", foreignField:"id_niveau",
as:"plan"}},
    //taille différente de zéro ==> il y a au moins une matiere
    {$match:{plan:{$not:{$size:0}}}},
    //projection pour garder année et spécialité
    {$project:{_id:0, annee:1, specialite:1}}
])

//Q2.c
db.niveau.aggregate([
    //Filtrer les niveau pour garder ceux de 2eme année
    {$match:{annee:2}},
    //jointure avec plan_eude pour récupérer les code_matiere dans un tableau plan
    {$lookup:{from:"plan_etude", localField:"id_niveau", foreignField:"id_niveau",
as:"plan"}},
    //jointure avec matiere pour récupérer nom_matiere dans un tableau matiere
    {$lookup:{from:"matiere", localField:"plan.code_matiere", foreignField:"code_matiere",
as:"matiere"}},
    //vérifier la présence de matiere avec un nom commençant par A
    {$match:{"matiere.nom_matiere":/^A/}},
    //projection pour la spécialité
    {$project:{_id:0, specialite:1}}
])
//Q2.d
db.niveau.aggregate([
    //jointure avec plan_eude pour récupérer les volume_semaine dans un tableau plan
    {$lookup:{from:"plan_etude", localField:"id_niveau", foreignField:"id_niveau",
as:"plan"}},
    //aplatir le tableau pour prépare l'agrégation
    {$unwind:"$plan"},
    //calcul de la somme des volume_semaine par id_niveau
    {$group:{_id:"$id_niveau", Volume:{$sum:"$plan.volume_semaine"}}}
])
```

```
////////////////////////////////////////////////////////////////////////////
// Exercice 2 :                                                           //
////////////////////////////////////////////////////////////////////////////

//Q1 Un modèle relationnel équivalent (n'est pas le seul), les colonnes entre [] sont les
clés primaires
Etudiants([numero], nom, prenom, groupe, #code_niveau)
Niveaux([code_niveau], annee, specialite)
Matieres([code_matiere], nom_matiere)
Notes([#numero, #code_matiere], note_ds, note_examen)

//Q2.a
db.etudiant.updateOne(
    {_id:ObjectId("52ed478d266ef548d")},
    {$push:{matieres:{libelle:"Anglais",ds:14.25,examen:13}}}
);
// ou
db.etudiant.updateOne(
    {_id:ObjectId("52ed478d266ef548d")},
    {$push:{matieres:{$each:[{libelle:"Anglais",ds:14.25,examen:13}], $position:0}}}
);

//Q2.b
db.etudiant.updateMany({},{$set:{absence:0}});

//Q2.c
db.etudiant.updateMany({niveau:{annee:1, specialite:"TI"}},{$inc:{absence:1}});

//Q2.d
db.etudiant.deleteMany({absence:{gt:4}});

//Q2.e
db.etudiant.updateMany({nom:/^A/},{$set:{groupe:1}});

//Q2.f
db.etudiant.find(
    {
        groupe:1,
        niveau:{annee:3,specialite:"DSI"},
        nom:/^A/
    },
    {_id:0, nom:1, prenom:1, matieres:1}
);
//Q2.g
db.etudiant.find({"matieres.examen":{$lt:10}},{_id:0, nom:1, prenom:1, matieres:1});

//Q2.h
db.etudiant.find(
    {
        "matieres.examen":{$not:{$lt:8}},
        "matieres.ds":{$not:{$lt:8}}
    },
    {_id:0, nom:1, prenom:1, matieres:1}
);
```

```
//Q2.i
db.etudiant.find(
    {
        "matieres.examen":{$not:{$gte:10,$lte:14}},
        "matieres.ds":{$not:{$gte:10,$lte:14}}
    },
    {_id:0, nom:1, prenom:1, matieres:1}
);

//Q2.j
// même groupe => même niveau(annee et specialite) et groupe
//donc grouper par 2 champs
db.etudiant.aggregate([
    {$group:{_id:{niv:"$niveau", grp:"$groupe"}, nbetudiants:{$sum:1}}},
    {$sort:{nbetudiants:-1}}
])

//Q2.k
db.etudiant.aggregate([
    {$match:{niveau:{annee:2, specialite:"SEM"},groupe:1}},
    {$unwind:"$matieres"},
    {$match:{"matieres.libelle":"POO"}},
    {$group:{_id:null, poomoyenne:{$avg:"$matieres.examen"}}},
])
```

```
////////////////////////////////////////////////////////////////////////////
// Exercice 3 :                                                           //
////////////////////////////////////////////////////////////////////////////

// Voici une proposition parmi plusieurs qui sont possibles

//Q1 :
// il faut comprendre que puisque un client achete plusieurs article donc il faut faire
// recours aux tableaux de documents
// ou utilser 2 collections avec référence (clé étrangère dans le modèle relationnel)
{
    numero_client:1,
    nom:"",
    prenom:"",
    adresse:"",
    article:[{description:"",prix_unitaire:1}]
}


//Q2 :
//même principe pour la relation entre representant et client puis client et commandes mais
puisque
//nous avons besoin seulement de la date des facture le tableau dates_commandes contient
uniquement des dates
//sans ajouter les autres propriétés des commandes
{
    numero_representant:1,
    nom:"",
    prenom:"",
    adresse:"",
    client:[
        {
            numero_client:"",
            nom_client:"",
            dates_commandes:["2022-12-15","2022-12-14"]
        },
        {
            numero_client:"",
            nom_client:"",
            dates_commandes:["2022-12-15","2022-12-14"]
        }
    ]
}
```