//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4  GUN     \*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

////---------------------------------------------------------------------------------

//// 9) inglizce dersini alanlarin subelere gore sayilari hesaplayan sorguyu yaziniz.

////---------------------------------------------------------------------------------

db.ogrenciler.aggregate([

{$match:{dersler:"inglizce"}},

{$group:{\_id:"$sube", IngAlanlarinSayisi:{$sum:1}}}

]);

////---------------------------------------------------------------------------------

//// 10) Fizik ve Ma dersi alanların subelere gore sayilari hesaplayan sorguyu yaziniz.

////---------------------------------------------------------------------------------

db.ogrenciler.aggregate([

  {$match:{dersler:["fizik","mat"]}},

  {$group:{\_id:["$sube","$isim"],fizikveMat:{$sum:1}}}

]);

////---------------------------------------------------------------------------------

//// 11) A şubesindeki ogrencileri yasa göre azalan sırada, nota göre yükeslen sırada siralayiniz.

////---------------------------------------------------------------------------------

db.ogrenciler.aggregate([

{$match:{sube:"A"}},

{$sort:{yas:-1, not:1}}

]);

use("kisiler")

db.hesaplar.insertMany(

[{"isim" : "ali", "harcamalar":[-80, -40, -50, -120], "kazanc":[100, 150]},

{"isim" : "Can", "harcamalar" : [-60, -30, -20], "kazanc" : [200, 50, 130] },

{"isim" : "Canan", "harcamalar" : [-80, -40, -50], "kazanc" : [300, 450] },

{"isim" : "Betul", "harcamalar" : [-80, -120], "kazanc" : [500, 50, 70, 10] },

{"isim" : "Ahsen", "harcamalar" : [-140, -50, -120], "kazanc" : [400] },

{"isim" : "Su", "harcamalar" : [-120], "kazanc" : [22, 375, 65] },

{"isim" : "Pelin", "harcamalar" : [-180, -40, -70, -12], "kazanc" : [500, 650, 400]}]);

////---------------------------------------------------------------------------------

//// 13) var olan dokumanlarda her kisi icin toplam harcama, toplam kazanc ve

////     bakiye degerlerini hesaplayarak dokumana ekleyen sorguyu yaziniz.

////     bakiye'ye gore siralayiniz.

////     NOT: Sorguda olmayan bir fiedl icin $addField metodu kullanilabilir.

////---------------------------------------------------------------------------------

pipeline1 = [

   {$addFields :{toplamHarcama: {$sum: "$harcamalar"}, toplamKazanc: {$sum: "$kazanc"}}} ,

   {$addFields:{bakiye:{$add:["$toplamHarcama" ,"$toplamKazanc"]}}},

   {$sort:{bakiye:-1}},

   {$project:{"harcamalar":0, kazanc:0,\_id:0}}

 ]

 db.hesaplar.aggregate(pipeline1);

 ////---------------------------------------------------------------------------------

//// 14) hesaplar collection’nin muhasabe adinda bir kopyasini olusturalim.

////---------------------------------------------------------------------------------

// use("kisiler")

db.hesaplar.aggregate( {$project: {\_id:0}},{$merge:{into:"muhasabe"}});

////---------------------------------------------------------------------------------

//// 15) hesaplar collection’nindan harcamalar kismini cikararark kazanclar adinda

////     yeni bir kopyasini olusturalim.

////---------------------------------------------------------------------------------

db.hesaplar.aggregate({$project: {harcamalar:0, \_id:0}}, {$merge: {into:“kazanclar”}});

////\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

////                                  BULKWRITE

////  Birden fazla islemi ayni koutla gerceklestirmek icin BulkWrite kullanilabilir.

////\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

use("kisiler");

db.hikayeler.insertMany(

[{"yazar" : "Mehmet Bak", "fiyat" : 60, "yayinYili" : 1998 },

{"yazar" : "Ali Gel", "fiyat" : 75, "yayinYili" : 2000 },

{"yazar" : "Su Ak", "fiyat" : 90, "yayinYili" : 2005 },

{"yazar" : "İsmet Kaç", "fiyat" : 35, "yayinYili" : 2010 },

{"yazar" : "Pelin Su", "fiyat" : 80, "yayinYili" : 2020 },

{"yazar" : "Suat Ok", "fiyat" : 54, "yayinYili" : 2001}]);

db.hikayeler.bulkWrite([

  {insertOne:{"document": {yazar: "Selim Genc", fiyat: 77, yayinYili : 2021 }}},

  {updateOne:{"filter":{yazar: "Ali Gel"}, "update": {$set: {fiyat:99}}}},

  {replaceOne:{"filter":{yazar:"Pelin Su"}, "replacement": {yazar: "Ceylan Kaya", fiyat:55}}},

  {deleteOne:{"filter" : {yazar: "Suat Ok"}}}

]);

////==================================================================================

////                          UNIONWITH (FULL JOIN)

////==================================================================================

//  - SQL'deki UNION ALL komutuna benzer.

//  - 2 collection'nin birlesimini gerceklestirmek icin kullamilir.

//  - Ornegin, $unionWith 2 farklı collection'daki pipeline sonuclarini tek bir

//    sonuc setinde birlesitirir.

//  - Bu sonuc seti pipeline'da bir sonraki asamaya iletibilir.

//  - SYNTAX:

//  - { $unionWith: { coll: "<collection>", pipeline: [ <stage1>, ... ] } }

////==================================================================================

use("kisiler");

db.romanlar.insertMany(

[{"yazar" : "Mehmet Bak", "fiyat" : 60, "yayinEvi" : "Yildiz", "adet": 1000 },

{"yazar" : "Ali Gel", "fiyat" : 75, "yayinEvi" : "MaviAy", "adet": 1200 },

{"yazar" : "Su Ak", "fiyat" : 90, "yayinEvi" : "Caliskan", "adet": 2200},

{"yazar" : "Meryem Can", "fiyat" : 35, "yayinEvi" : "MorEv","adet": 560},

{"yazar" : "Pelin Su", "fiyat" : 80, "yayinEvi" : "Hedef", "adet": 890 },

{"yazar" : "Suat Ok", "fiyat" : 54, "yayinEvi" : "Sinir","adet": 245}]);

db.denemeler.insertMany(

[{"muellif" : "Mehmet Bak", "fiyat" : 34, "yayinEvi" : "Yildiz", "adet": 400 },

{"muellif" : "Deniz Kos", "fiyat" : 44, "yayinEvi" : "Yildiz", "adet": 350 },

{"muellif" : "Su Ak", "fiyat" : 50, "yayinEvi" : "MorEv", "adet": 200},

{"muellif" : "İsmet Kaç", "fiyat" : 25, "yayinEvi" : "Hedef","adet": 800},

{"muellif" : "Ali Gel", "fiyat" : 40, "yayinEvi" : "Hedef", "adet": 1200 },

{"muellif" : "Meryem Can", "fiyat" : 22, "yayinEvi" : "MaviAy","adet": 300}]);

////---------------------------------------------------------------------------------

//// 1)  Her 2 collection'dan tum dokumanlari "adet" sutununa (field) gore siralayarak

////     listeleyiniz. (FULL JOIN)

// ////---------------------------------------------------------------------------------

// db.romanlar.aggregate([

//  {$unionWith: { coll: "denemeler"}},

//  {$sort:{adet: -1}}

// ]);

db.denemeler.aggregate([

 {$unionWith: { coll: "romanlar"}},

 {$sort:{adet: -1}},

 {$project: {\_id:0}}

]);

////---------------------------------------------------------------------------------

//// 2) Her bir yayinevinin toplam kitap sayilarini (2 collection icin) hesaplayan

////    ve siralayan sorguyu yaziniz.

////---------------------------------------------------------------------------------

db.romanlar.aggregate([

  {$unionWith:{coll:"denemeler"}},

  {$group:{\_id:"$yayinEvi", toplamAdet:{$sum:"$adet"}}},

  {$sort: {toplamAdet:1}}

]);

////---------------------------------------------------------------------------------

//// 3) Her bir yayinEvinin kitap (makale ve roman) satislarindan  elde edecegi toplam

////    geliri hesaplayan sorguyu yaziniz. (carpma icin $multiply)

////---------------------------------------------------------------------------------

db.romanlar.aggregate([

 {$unionWith: {coll:"denemeler"}},

 {$group: {\_id:"$yayinEvi", toplamGelir:{$sum:{$multiply:["$fiyat","$adet"]}}}}

]);

////---------------------------------------------------------------------------------

//// 4) Her bir yazarin toplam kitap sayilarini (2 collection icin) hesaplayan

////    ve siralayan sorguyu yaziniz.

////    PROBLEM:Collection'lardaki sutun isilerim farklı (yazar, muellif)

////---------------------------------------------------------------------------------

// Grup isimleri ayni olmadigi icin (yazar veya muellif) 2 degerin toplamini almayacaktir.

db.romanlar.aggregate([

 {$unionWith: {coll: "denemeler"}},

 {$group: {\_id:"$yazar", toplamKitap: {$sum: "$adet"}}}

])

////==================================================================================

////                       $LOOKUP (LEFT, RIGHT, INNER JOIN)

////    {

////      $lookup:

////      {

////        from: <Join uygulanacak collection>,

////        localField: <giris dokumanindaki ilgili sutunu>,

////        foreignField: <Join yapilacak collectiondaki ilgili sutun>,

////        as: <cikis sutunu>

////      }

////   }

////==================================================================================

////---------------------------------------------------------------------------------

//// 5) romanlar collectionda yer alan her bir yazarin toplam kitap sayilarini (

////    2 collection icin) hesaplayan sorguyu yaziniz. (LEFT JOIN)

////---------------------------------------------------------------------------------

db.romanlar.aggregate(

  {

    $lookup:{

     from:"denemeler",

     localField:"yazar",

     foreignField:"muellif",

     as:"yazarlar"

    }

  },

  {

   $unwind: {path:"$yazarlar", preserveNullAndEmptyArrays:true}

  },

  {

   $project:{toplamKitap: {$add: ["$adet", {$ifNull:["$yazarlar.adet", 0]}]},

   \_id:0, yazar:1

   }

  }

);

////---------------------------------------------------------------------------------

//// 6) Her iki collectionda ortak olan yazarlarin toplam kitap sayilarini ve kitap

////    fiyatlarinin toplamini hesaplayan sorguyu yaziniz. (INNER JOIN)

////---------------------------------------------------------------------------------

db.romanlar.aggregate([

  {

    $lookup: {

      from:"denemeler",

      localField:"yazar",     //romanlardaki (PK)

      foreignField:"muellif", // denemelerdeki(FK)

      as:"ortak"              // yeni dokuman icin

    }

  },

  {

     $unwind:{

        path:"$ortak"

     }

  },w

  {

     $project:{

       toplamFiyat:{$add:["$fiyat","$ortak.fiyat"]},

       toplamAdet:{$add:["$adet","$ortak.adet"]},

       \_id:0, yazar:1

     },

  }

]);