Барышников Владимир Романович К33391 – ису 334718

Вопросы – 2,5,8,11

**2.** Inclusion Projection даёт возможность выбрать определенные поля для включения в результат запроса. Таким образом можно указать, какие поля хотим видеть в результатах какого-либо запроса. *Пример*: задание 8.2.1, где мы выводили только название города и информацию о мэре.  
  
Exclusion Projection позволяет выбирать все поля, кроме указанных, для исключения из результата запроса. *Пример*: задание 8.1.3, где мы исключали данные о предпочтении и поле единорога  
  
\_id:  
• \_id - это уникальный идентификатор каждого документа в коллекции, его нельзя изменить или удалить  
• Поле \_id по умолчанию включено в результаты всех запросов, даже если остальные поля не указаны   
• Если хотим исключить \_id из результата запроса, то это нужно прописать в Exclusion Projection (*Пример*: задание 8.1.6 в лабораторной работе)  
  
**5.** Синтаксис агрегации:  
db.collection.aggregate([  
  { $match: { case: value } }, — Фильтрация по определенному признаку  
  { $group: { \_id: "$case", total: { $sum: 1 } } }, — Группировка по какому-либо признаку. В результате это выведет уникальные значения по $case и количество документов в каждой группе  
  { $sort: {case: -1 } } — Сортировка (в данном случае по убыванию столбца case)  
])  
  
Метод sort используется при нахождении элементов при помощью find (к примеру, db.unicorns.find({}).sort({name: 1}) выведет нам всех единорогов, отсортированных в алфавитном порядке их имен)  
  
$sort в агрегации, в отличие от обычного sort, позволяет работать с более сложными и сгруппированными данными  
Пример:  
db.unicorns.aggregate([  
  { $group: { \_id: "$gender", count: { $sum: 1 } } },  
  { $sort: { count: -1 } }   
])  
Тут мы сортируем по получившемуся уже во время агрегации столбу count и, таким образом, сначала выводим пол с наибольшим количеством единорогов14:39

**8.** Операторы сравнения:  
  
$gt - Greater Than (больше чем).  
$lt - Less Than (меньше чем).  
$gte - Greater Than or Equal (больше или равно).  
$lte - Less Than or Equal (меньше или равно).  
  
 $regex:  
  
$regex - позволяет использовать регулярные выражения для поиска. Например: db.unicorns.find({name: {$regex: /a.\*a/i }}) — поиск среди единорогов тех , которые в своем имени имеют две буквы A (без учета регистра). Выводится Aurora, Solnara, Ayna

**11.** Создание БД:

use task11db  
  
Создание предметов:  
db.courses.insertMany([  
  {\_id: 1,  
    name: "Математика",  
    instructor: "Дробцева А. И.",  
  attestation: "exam"},  
  {\_id: 2,  
    name: "История",  
    instructor: "Бонапартов Н. И.",  
    attestation: "pass"},  
  {\_id: 3,  
    name: "Физика",  
    instructor: "Гравитцев Н. О.",  
    attestation: "exam"}])  
  
Создание учеников ():  
db.students.insertMany([  
  { \_id: 1,  
    name: "Анна Сидоркина",  
    age: 20,},  
  {\_id: 2,  
    name: "Петр Отличнов",  
    age: 22,},  
  {\_id: 3,  
    name: "Ольга Неизвестная",  
    age: 19,}]);  
  
Затем обновляем значения учеников — добавляем к ним их курсы:  
db.students.update({ \_id: 1 }, { $set: { courses: [1, 3] } });  
db.students.update({ \_id: 2 }, { $set: { courses: [2, 3] } });  
db.students.update({ \_id: 3 }, { $set: { courses: [1] } });

function printStudentsCourses() {  
  var students = db.students.find();  
  
  while (students.hasNext()) {  
    var student = student.next();  
    print("Студент: " + student.name);  
    print("Курсы:");  
  
    for (var i = 0; i < student.courses.length; i++) {  
      var courseId = student.courses [i];  
      var course = db.courses.findOne({ \_id: courseId });  
      if (course) {  
        print(" - " + course.name);}}}}  
  
printStudentsCourses();

