Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут”

Кафедра АСОІУ

**ЗВІТ**

про виконання комп’ютерного практикуму № 5

з дисципліни

“ Програмні додатки з використанням баз даних”

по темі

«MongoDB»

Варіант 1

|  |  |  |
| --- | --- | --- |
| Прийняв: |  | Виконав: |
| Cт.вик. Клименко О. М. |  | студент 3-го курсу  гр. ІП-51 ФІОТ  Булатов Дмитро Эгорович |

Київ – 2017

**ЗМІСТ:**

[1 Завдання 3](#_Toc495230724)

[2 Виконання Завдання 4](#_Toc495230725)

# Завдання

Завдання 1:

1. Встановити MongoDB
2. Створити базу MongoDB на основі реляційної бази предметної області

Завдання 2:

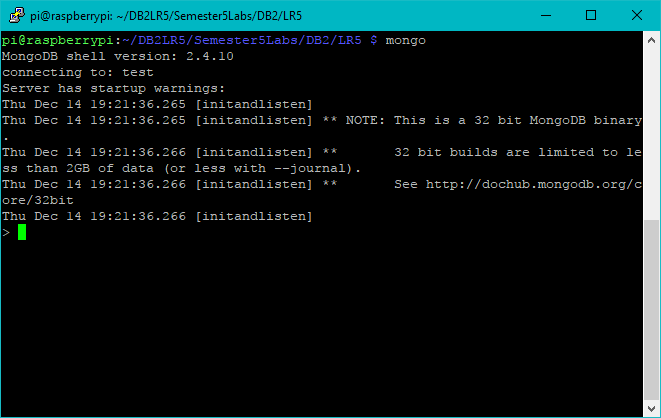
1. Перепроектувати схеми БД та доповнити даними
2. Написати до своєї предметної області 2 запити з групуванням та 2 запити з використанням агрегатних функцій.
3. Виконати запит в MongoDB, який за кодом, що зберігається в одній колекції, виводить назву, що зберігається в іншій колекції, тобто аналог JOIN в SQL.

**Преметна область**: Відеопрокат

# Виконання Завдання

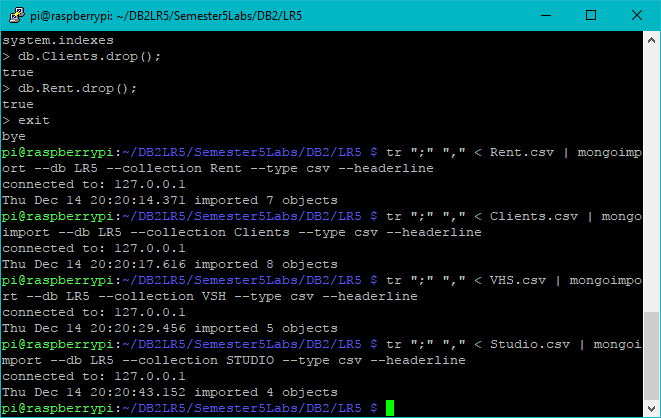
1. Завдання по базовим операціям з MongoDB:
   1. Встановлення MongoDB

*Результат виконання*:



* 1. Створити базу MongoDB на основі реляційної бази предметної області:

*Результат виконання*:



1. Завдання по написанню запитів в MongoDB:
   1. Перепроектувати схеми БД та доповнити даними

Опис дій: В колекції Clients було додане поле Rents. Колекція Rent була видалена, оскільки містила дублюючу інформацію. В наслідок всіх цих операцій ми отримали структуру яка достатньо зрозуміла для людини, оскільки оперує об’єктами та зручний доступ до всіх необхідних полів даних. Але тепер база даних не нормалізована (відповідає лише 1 НФ), що ускладнить її подальшу модифікацію і супроводження.

* 1. Написати до своєї предметної області 2 запити з групуванням та 2 запити з використанням агрегатних функцій.
     1. *Призначення*: Підрахувати кількість користувачів зробивших хоча б одне замевлення

*Текст запиту*:

db.Clients.aggregate([

{

$match: {

'Rents.0': {$exists: true}

}

},

{

$group: {

\_id: null,

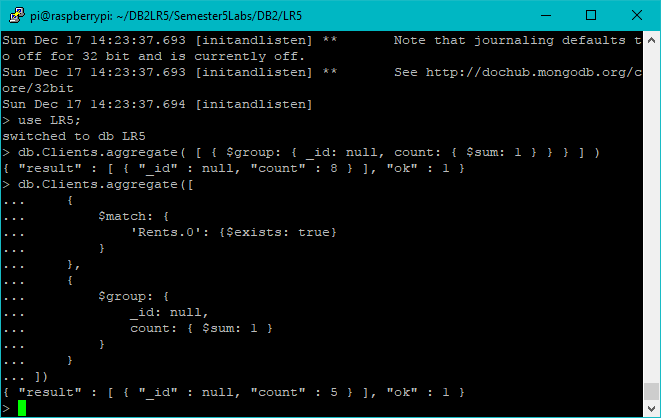
count: { $sum: 1 }

}

}

])

*Результат виконання*:



* + 1. *Призначення*: Підраховує кількість студій з кожної країни

*Текст запиту*:

db.Studios.aggregate([

{

$group: {

\_id: "$Country",

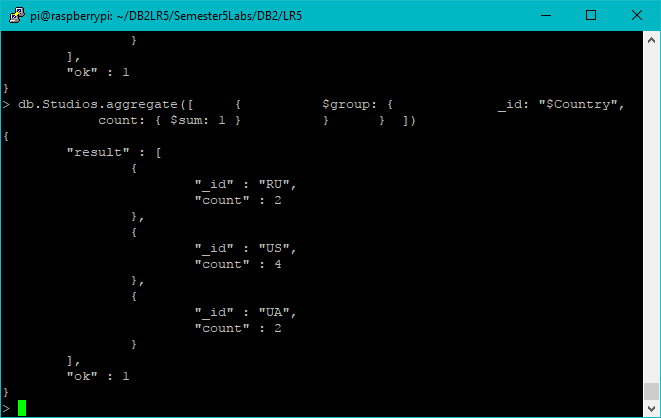
count: { $sum: 1 }

}

}

])

*Результат виконання*:



* + 1. *Призначення*: Вивести останню дату оренди касети

*Текст запиту*:

db.Clients.aggregate([

{

$unwind:"$Rents"

},

{

$group:{

\_id: null,

lastDate: {

$max: "$Rents.StartDate"

}

}

}

])

*Результат виконання*:



* + 1. *Призначення*: Вивести продюсера зробившого найбільшу кількість касет.

*Текст запиту*:

db.VHS.aggregate([

{

$group:{

\_id: "$Director",

directorVHSs: {

$sum: 1

}

}

},

{

$sort:{

directorVHSs: -1

}

},

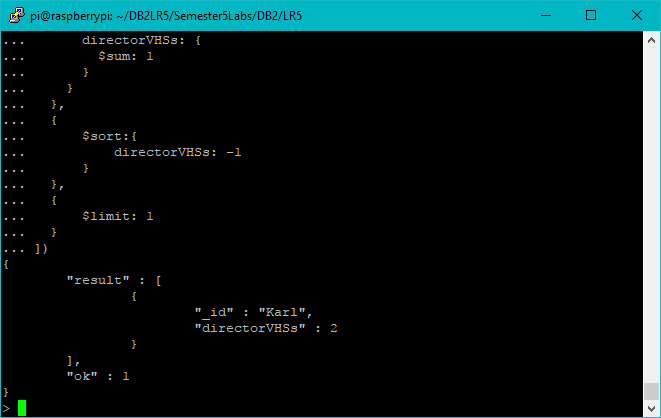
{

$limit: 1

}

])

*Результат виконання*:



* 1. Виконати запит в MongoDB, який за кодом, що зберігається в одній колекції, виводить назву, що зберігається в іншій колекції, тобто аналог JOIN в SQL.

*Призначення*: Вивести всі данні про студію кожної касети.

*Текст запиту*:

var mapVHS = function() {emit(this.StudioID, {StudioID: this.StudioID, VHS\_ID: this.ID, Director: this.Director})};

var mapStudio = function() {emit(this.ID, {StudioID: this.ID, StudioCountry: this.Country, StudioName: this.Name})};

var reduceF = function(key, values) {

var res = {};

values.forEach(function(v){

if(!res.StudioID)

res.StudioID = v.StudioID;

if(!res.VHS\_ID)

res.VHS\_ID = v.VHS\_ID;

if(!res.Director)

res.Director = v.Director;

if(!res.StudioCountry)

res.StudioCountry = v.StudioCountry;

if(!res.StudioName)

res.StudioName = v.StudioName;

});

return res;

};

db.VHS.mapReduce(mapVHS, reduceF, {out: {reduce: 'joined'}});

db.Studios.mapReduce(mapStudio, reduceF, {out: {reduce: 'joined'}});

*Результат виконання*:

