# Lab 9 – Due by Saturday, April 10th by 9pm

# (MongoDB – Query) Find documents

## Objective

In this Lab, you learn to query a database in MongoDB by using **find** command.

## Getting Started

Open your Windows command prompt and go the following directory where MongoDB is installed:

* cd C:\Program Files\MongoDB\Server\4.4\**bin**

To run MongoDB, execute ***mongod***

* mongod

When MongoDB starts successfully, open another Windows command prompt and go the same *bin* directory:

* cd C:\Program Files\MongoDB\Server\4.4\**bin**

and execute ***mongo (****this session you will use to find your Documents)*

* mongo

In this lab, you will use **products.json** dataset. Download products.json from Blackboard and then IMPORT it, like it has been explained in “How to use Mongo Import” (you will need to open a Third command prompt session). You will create a new database called **inventory** and then place a new collection **products** there.

> After executing the command, the data is imported to the *inventory* database. To make sure data is imported successfully, go to the MongoDB shell and execute the following command to see the imported documents:

* show dbs

You should see the database *inventory* added to the list of your databases. To see the documents inside the database:

* use inventory
* db.products.find()

## Submission

You submit this file with answers (in the provided space). Also, include the **outputs** with your code. Name the file L09\_LASTNAME.docx”.

## Tasks

1. Write a query to return *name* and *price* of each product in the *inventory* database.

|  |
| --- |
| db.products.find({},{"\_id":0,"name":1,"price":1})  OUTPUT:  { "name" : "Phone Extended Warranty", "price" : 38 }  { "name" : "Phone Service Core Plan" }  { "name" : "AC3 Phone", "price" : 200 }  { "name" : "Cable TV Basic Service Package" }  { "name" : "Phone Service Basic Plan" }  { "name" : "AC3 Case Black", "price" : 12.5 }  { "name" : "AC3 Case Green", "price" : 12 }  { "name" : "AC3 Series Charger", "price" : 19 }  { "name" : "AC3 Case Red", "price" : 12 }  { "name" : "Phone Service Family Plan" }  { "name" : "AC7 Phone", "price" : 320 } |

1. Write a query to return *name* and *price* for products of type *accessory* in the *inventory* database.

|  |
| --- |
| db.products.find({"type":"accessory"},{"\_id":0,"name":1,"price":1,"type":1})  OUTPUT  { "name" : "AC3 Case Black", "type" : [ "accessory", "case" ], "price" : 12.5 }  { "name" : "AC3 Case Green", "type" : [ "accessory", "case" ], "price" : 12 }  { "name" : "AC3 Series Charger", "type" : [ "accessory", "charger" ], "price" : 19 }  { "name" : "AC3 Case Red", "type" : [ "accessory", "case" ], "price" : 12 } |

1. Write a query to return *name* and *price* for products with price between $12 and $20 (Values *12* and *20* are included).

|  |
| --- |
| db.products.find({"price":{"$gte":12,"$lte":20}},{"\_id":0,"name":1,"price":1})  OUTPUT  { "name" : "AC3 Case Black", "price" : 12.5 }  { "name" : "AC3 Case Green", "price" : 12 }  { "name" : "AC3 Series Charger", "price" : 19 }  { "name" : "AC3 Case Red", "price" : 12 } |

1. Write a query to return *id*, *name*, *price*, and *type* for products that are not of type *accessory*.

|  |
| --- |
| db.products.find({"type":{"$ne":"accessory"}},{"\_id":1,"name":1,"price":1,"type":1})  OUTPUT  { "\_id" : ObjectId("507d95d5719dbef170f15bfb"), "name" : "Phone Extended Warranty", "type" : "warranty", "price" : 38 }  { "\_id" : ObjectId("507d95d5719dbef170f15bff"), "name" : "Phone Service Core Plan", "type" : "service" }  { "\_id" : "ac3", "name" : "AC3 Phone", "type" : "phone", "price" : 200 }  { "\_id" : ObjectId("507d95d5719dbef170f15c01"), "name" : "Cable TV Basic Service Package", "type" : "tv" }  { "\_id" : ObjectId("507d95d5719dbef170f15bfe"), "name" : "Phone Service Basic Plan", "type" : "service" }  { "\_id" : ObjectId("507d95d5719dbef170f15c00"), "name" : "Phone Service Family Plan", "type" : "service" }  { "\_id" : "ac7", "name" : "AC7 Phone", "type" : "phone", "price" : 320 } |

1. Write a query to return *id*, *name*, *price*, and type for products with type *accessory* or *service*.

|  |
| --- |
| db.products.find({"type":{"$in":["accessory","service"]}},{"\_id":1,"name":1,"price":1,"type":1})  OUTPUT:  { "\_id" : ObjectId("507d95d5719dbef170f15bff"), "name" : "Phone Service Core Plan", "type" : "service" }  { "\_id" : ObjectId("507d95d5719dbef170f15bfe"), "name" : "Phone Service Basic Plan", "type" : "service" }  { "\_id" : ObjectId("507d95d5719dbef170f15bfc"), "name" : "AC3 Case Black", "type" : [ "accessory", "case" ], "price" : 12.5 }  { "\_id" : ObjectId("507d95d5719dbef170f15bfa"), "name" : "AC3 Case Green", "type" : [ "accessory", "case" ], "price" : 12 }  { "\_id" : ObjectId("507d95d5719dbef170f15bf9"), "name" : "AC3 Series Charger", "type" : [ "accessory", "charger" ], "price" : 19 }  { "\_id" : ObjectId("507d95d5719dbef170f15bfd"), "name" : "AC3 Case Red", "type" : [ "accessory", "case" ], "price" : 12 }  { "\_id" : ObjectId("507d95d5719dbef170f15c00"), "name" : "Phone Service Family Plan", "type" : "service" } |

1. Write a query to return *id*, *name*, *price*, and *type* for products that do have the *type* key.

|  |
| --- |
| db.products.find({"type":{"$exists":false}},{"\_id":1,"name":1,"price":1,"type":1})  OUTPUT:  //nothing |

1. Write a query to return *id*, *name*, *price*, and *type* for products that their type is both *accessory* and *case*.

|  |
| --- |
| db.products.find({"type":{"$all":["accessory","case"]}},{"\_id":1,"name":1,"price":1,"type":1})  OUTPUT:  { "\_id" : ObjectId("507d95d5719dbef170f15bfc"), "name" : "AC3 Case Black", "type" : [ "accessory", "case" ], "price" : 12.5 }  { "\_id" : ObjectId("507d95d5719dbef170f15bfa"), "name" : "AC3 Case Green", "type" : [ "accessory", "case" ], "price" : 12 }  { "\_id" : ObjectId("507d95d5719dbef170f15bfd"), "name" : "AC3 Case Red","type" : [ "accessory", "case" ], "price" : 12 } |