LAB SUBMISSION SYSTEM: GOOGLE CLASSROOM

AT82.02

DATA MODELING AND MANAGEMENT

LAB6: MONGODB- DOCUMENT MODEL

Co-funded by the Erasmus+ Programme of the European Union

DS&AI

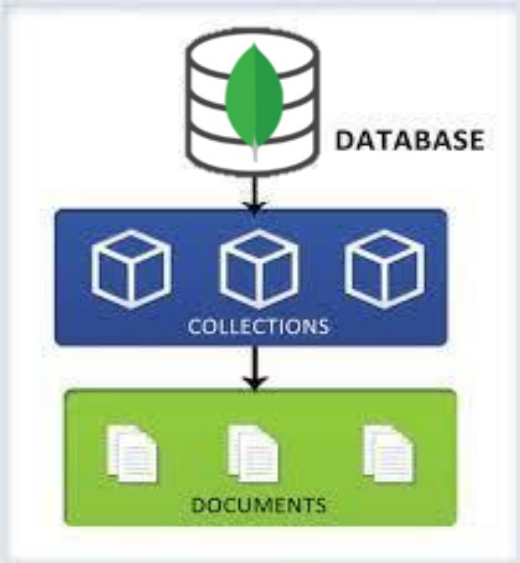
Outline

1. Connect to a MongoDB Cloud database
2. CRUD Operations of MongoDB
   - Create (Insert), Update, Delete Documents
   - Read data
     - Simple query
     - Query nested field
     - Aggregate function
     - Join/Link documents

MongoDB Lab Architecture

- 2 Databases
  - Shop101
  - RDBProject
- Shop101
  - Customer
- Username
  - st_dmm/st_dmm
  - onlne/online
  - offline/offline

DDM2020 MongoDB Cloud



MongoDB Shell

```
> db.collection.insert({company:"10gen", product:"MongoDB"})
>
> db.collection.findOne()
{
        "_id": ObjectId("5106c1c2fc629bfe52792e86"),
        "company": "10gen"
        "product": "MongoDB"
}
```

mongo "mongodb+srv://dmmcluster.fluoi.gcp.mongodb.net/<dbname>"
--username <username>
--password <password>

i.e., mongo "mongodb+srv://dmmcluster.fluoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

1.Connect to shop101 database

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
D:\_Phd_Candidate\AITTA-DMM\mongodb-win32-x86_64-windows-4.4.1\bin>mongo "mongodb+s
rv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm
MongoDB shell version v4.4.1
connecting to: mongodb://dmmcluster-shard-00-01.f1uoi.gcp.mongodb.net:27017,dmmclus
ter-shard-00-02.f1uoi.gcp.mongodb.net:27017,dmmcluster-shard-00-00.f1uoi.gcp.mongod
b.net:27017/shop101?authSource=admin&compressors=disabled&gssapiServiceName=mongodb
&replicaSet=atlas-omm7a6-shard-0&ssl=true
Implicit session: session { "id" : UUID("1a9513c0-f07a-4484-a914-c078154589fc") }
MongoDB server version: 4.2.9
WARNING: shell and server versions do not match
Error while trying to show server startup warnings: user is not allowed to do actio
n [getLog] on [admin.]
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> l
uncaught exception: ReferenceError: l is not defined :
@(shell):1:1
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

Common command in mongoDB shell

more command : https://docs.mongodb.com/manual/reference/mongo-shell/

Show databases

Listing all the databases in mongoDB

Show collections

Listing all the tables in the current database

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> show databases
RDBProject  0.000GB
admin       0.000GB
local       1.224GB
shop101     0.000GB
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> show collections
customer
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

Common command in mongoDB shell

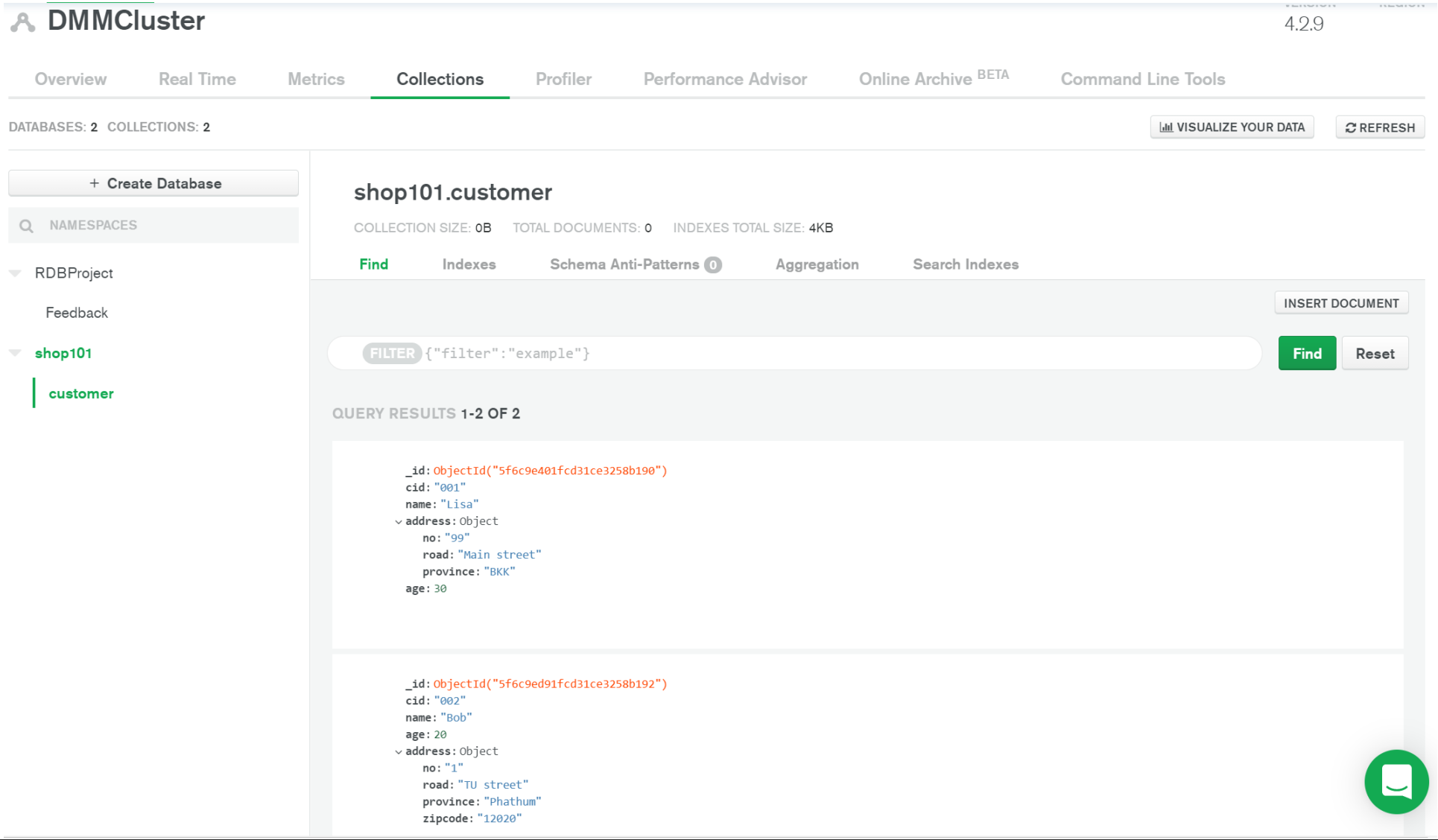
more command : https://docs.mongodb.com/manual/reference/mongo-shell/

db.collection.find() : *Find all documents in the collection.*

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.customer.find()
{ "_id" : ObjectId("5f6c9e401fcd31ce3258b190"), "cid" : "001", "name"
: "Lisa", "address" : { "no" : "99", "road" : "Main street", "province
" : "BKK" }, "age" : 30 }
{ "_id" : ObjectId("5f6c9ed91fcd31ce3258b192"), "cid" : "002", "name"
: "Bob", "age" : 20, "address" : { "no" : "1", "road" : "TU street", "
province" : "Phathum", "zipcode" : "12020" } }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

Shop101 Database in MongoDB Cloud Atlas



2. INSERT Operations of MongoDB

Insert Documents

db.collection.insertMany()

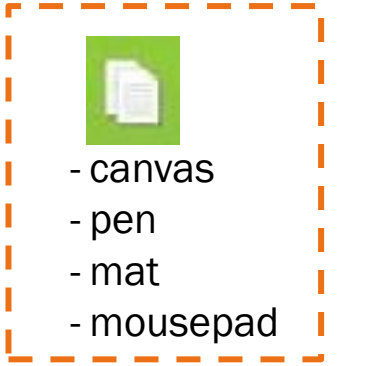
Syntax:

```
db.collection.insertMany(
    [ <document 1> , <document 2>, ... ]
)
```

Document Model Architecture

DDM2020 MongoDB Server

Insert information about inventory



Insert inventory documents

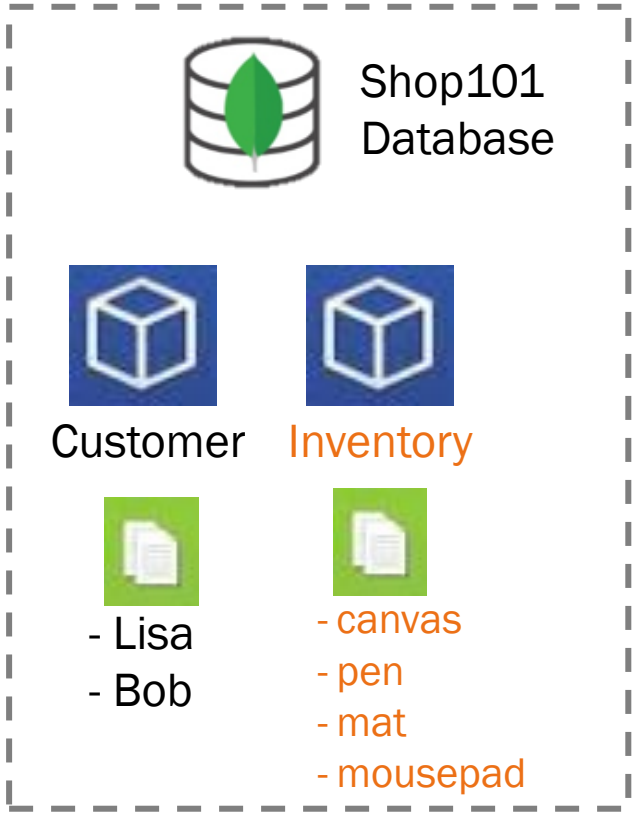
```
db.inventory.insertOne(
  { item: "canvas", qty: 100, price: 500, tags: ["cotton"], size: { h: 28, w: 35.5, uom: "cm" } })
db.inventory.insertMany([
  { item: "pen", qty: 25, price: 200, tags: ["red"], size: { h: 14, w: 1, uom: "cm" } },
  { item: "mat", qty: 85, price: 99,  tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },
  { item: "mousepad", qty: 25, price: 29, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "in"
} }
])
```

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.inventory.insertOne(
...     { item: "canvas", qty: 100, price: 500, tags: ["cotton"], size: { h: 28, w: 35.5, uom: "cm" } })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5f6cb1e662783426b926a1b7")
}
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.inventory.insertMany([
...     { item: "pen", qty: 25, price: 200, tags: ["red"], size: { h: 14, w: 1, uom: "cm" } },
...     { item: "mat", qty: 85, price: 99,  tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },
...     { item: "mousepad", qty: 25, price: 29, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "in" } }
... ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("5f6cb1e662783426b926a1b8"),
                ObjectId("5f6cb1e662783426b926a1b9"),
                ObjectId("5f6cb1e662783426b926a1ba")
        ]
}
```

Verify the inventory insertion

DDM2020 MongoDB Cloud

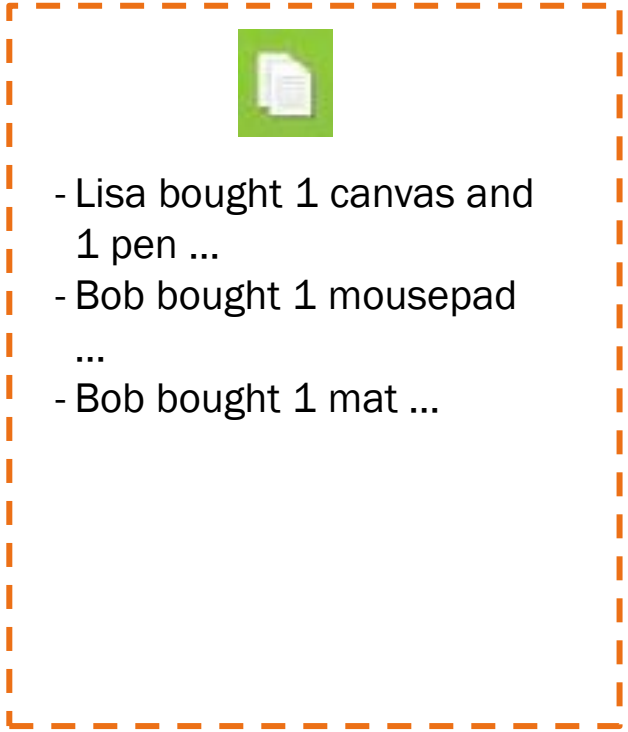


Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> show collections
customer
inventory
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.inventory.find()
{ "_id" : ObjectId("5f6cb1e662783426b926a1b7"), "item" : "canvas", "qty"
: 100, "price" : 500, "tags" : [ "cotton" ], "size" : { "h" : 28, "w" : 3
5.5, "uom" : "cm" } }
{ "_id" : ObjectId("5f6cb1e662783426b926a1b8"), "item" : "pen", "qty" : 2
5, "price" : 200, "tags" : [ "red" ], "size" : { "h" : 14, "w" : 1, "uom"
 : "cm" } }
{ "_id" : ObjectId("5f6cb1e662783426b926a1b9"), "item" : "mat", "qty" : 8
5, "price" : 99, "tags" : [ "gray" ], "size" : { "h" : 27.9, "w" : 35.5,
"uom" : "cm" } }
{ "_id" : ObjectId("5f6cb1e662783426b926a1ba"), "item" : "mousepad", "qty
" : 25, "price" : 29, "tags" : [ "gel", "blue" ], "size" : { "h" : 19, "w
" : 22.85, "uom" : "in" } }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

Insert Sales information

Insert transaction about sales

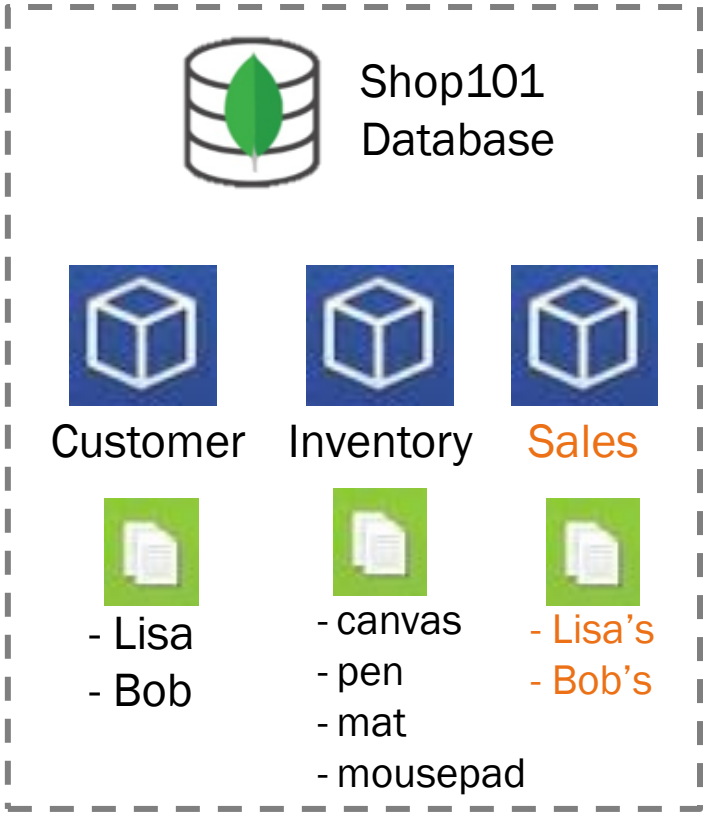


Insert sales

```
db.sales.insertOne(
  { item: [{pid: "canvas", qty: 1, price: 500},{pid: "pen", qty: 1, price: 200}], customer:"Lisa" ,
total:700 , couponused:true  })
db.sales.insertOne(
  { item: [{pid: "mousepad", qty: 1, price: 29}], customer:"Bob" , total:29, couponused:true  })
db.sales.insertOne(
  { item: [{pid: "mat", qty: 1, price: 99}], customer:"Bob" , total:99, couponused:false  })
```

Verify the sales insertion

DDM2020 MongoDB Cloud




```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.sales.find().pretty()
{
        "_id" : ObjectId("5d8f62648fb57f941738209b"),
        "item" : [
                {
                        "pid" : "canvas",
                        "qty" : 1,
                        "price" : 500
                },
                {
                        "pid" : "pen",
                        "qty" : 1,
                        "price" : 200
                }
        ],
        "customer" : "Lisa",
        "total" : 700,
        "couponused" : true
}
{
        "_id" : ObjectId("5d8f63588fb57f941738209c"),
```

3. UPDATE Operations of MongoDB

Update customer add status field

Relational Database SQL	MongoDB Command
Alter table customer add status char(1) + Update customer set status ="A"	db.customer.updateMany({ }, {$set: {status: "A"}})
Update customer set status ="I" where name ="Lisa"	db.customer.updateMany({name:"Lisa" }, {$set: {status: "I"}})

- Lisa
- Bob

- Lisa status="I"
- Bob status="A"

4. Basic QUERY in MongoDB

FIND()

Query Documents

db.collection.find()

`db.collection.find(query, projection)`

- **query**: document
  - Optional. Specifies selection filter using query operators. To return all documents in a collection, omit this parameter or pass an empty document ({}).
- **projection**: document
  - Optional. Specifies the fields to return in the documents that match the query filter. To return all fields in the matching documents, omit this parameter.

https://docs.mongodb.com/manual/reference/method/db.collection.find/#db.collection.find

Note: pretty() - displays the results in a formatted way.

Basic Select

SQL SELECT Statements	MongoDB find() Statements
`SELECT *` `FROM Customer`	`db.customer.find().pretty()`

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find().pretty()
{
        "_id" : ObjectId("5d8e32391c9d440000ea487d"),
        "cid" : "001",
        "name" : "Lisa",
        "address" : {
                "no" : "99",
                "road" : "main street",
                "province" : "BKK"
        },
        "age" : 30,
        "status" : "I"
}
{
        "_id" : ObjectId("5d8e34411c9d440000ea487f"),
        "cid" : "002",
        "name" : "Bob",
        "address" : {
                "no" : "1",
                "road" : "TU street",
                "province" : "Phathum",
                "zipcode" : "12020"
        },
        "age" : 20,
        "status" : "A"
}
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

Projection (return some fields)

```
SELECT cid,name
FROM Customer
```

```
db.Customer. find(
    { },
    {cid:1, name:1,_id:0}
)
```

Show

Not Show

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find({},{cid:1,name:1,_id:0})
{ "cid" : "001", "name" : "Lisa" }
{ "cid" : "002", "name" : "Bob" }
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

Order by

Select * from customer order by name	db.customer.find().sort({name:1})	1 = ASC
Select * from customer order by name desc	db.customer.find().sort({name:-1})	-1 = DESC

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.customer.find({},{cid:1,name:1,_id:0})
.sort({name:1})
{ "cid" : "002", "name" : "Bob" }
{ "cid" : "001", "name" : "Lisa" }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.customer.find({},{cid:1,name:1,_id:0})
.sort({name:-1})
{ "cid" : "001", "name" : "Lisa" }
{ "cid" : "002", "name" : "Bob" }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

WHERE condition

```
SELECT *
FROM Customer
WHERE Name ="Lisa"
```

```
db.Customer.find(
        {name ="Lisa"}
)
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find({name:"Lisa"}).pretty()
{
        "_id" : ObjectId("5d8e32391c9d440000ea487d"),
        "cid" : "001",
        "name" : "Lisa",
        "address" : {
                "no" : "99",
                "road" : "main street",
                "province" : "BKK"
        },
        "age" : 30
}
```

```
SELECT *
FROM Customer
WHERE name != "Lisa"
```

```
db.Customer.find(
    { name : { $ne:"Lisa" } }
)
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find({name:{$ne:"Lisa"}}).pretty()
{
        "_id" : ObjectId("5d8e34411c9d440000ea487f"),
        "cid" : "002",
        "name" : "Bob",
        "address" : {
                "no" : "1",
                "road" : "TU street",
                "province" : "Phathum",
                "zipcode" : "12020"
        },
        "age" : 20
}
```

```sql
SELECT *
FROM Customer
WHERE age > 25
```

```mongo
db.Customer.find(
    { age: { $gt: 25 } }
)
```

```sql
SELECT *
FROM Customer
WHERE age < 25
```

```mongo
db.Customer.find(
    { age: { $lt: 25 } }
)
```

Query Operators

- $eq: Matches values that are equal to a specified value.
- $gt: Matches values that are greater than a specified value.
- $gte: Matches values that are greater than or equal to a specified value.
- $in: Matches any of the values specified in an array.
- $lt: Matches values that are less than a specified value.
- $lte: Matches values that are less than or equal to a specified value.
- $ne: Matches all values that are not equal to a specified value.
- $nin: Matches none of the values specified in an array.

AND

```
SELECT *
FROM Customer
WHERE age > 25
AND   age <= 50
```

```
db.Customer.find(
    { age: { $gt: 25, $lte: 50 } }
)
```

Example : db.customer.find({$and : [{status: "A"}, {age:20}]})

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.customer.find({$and : [{status: "A"}, {age:20}]})
{ "_id" : ObjectId("5f6c9ed91fcd31ce3258b192"), "cid" : "002", "name" : "Bob", "age" : 20, "address" :
 { "no" : "1", "road" : "TU street", "province" : "Phathum", "zipcode" : "12020" }, "status" : "A" }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

OR

```sql
SELECT *
FROM Customer
WHERE status = "A"
OR age = 20
```

```mongo
db.Customer.find(
    { $or: [ { status: "A" } , { age: 20 } ] }
)
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find({$or:[{status:"A"},{age: 20}]}).pretty()
{
        "_id" : ObjectId("5d8e34411c9d440000ea487f"),
        "cid" : "002",
        "name" : "Bob",
        "address" : {
                "no" : "1",
                "road" : "TU street",
                "province" : "Phathum",
                "zipcode" : "12020"
        },
        "age" : 20,
        "status" : "A"
}
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

LIKE

```
SELECT *
FROM   Customer
WHERE  name  like "%is%"
```

```
db.Customer.find( { name : /is/ } )
```

-or-

```
db.Customer.find( { name : { $regex: /is/ } } )
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find({name :/is/}).pretty()
{
        "_id" : ObjectId("5d8e32391c9d440000ea487d"),
        "cid" : "001",
        "name" : "Lisa",
        "address" : {
                "no" : "99",
                "road" : "main street",
                "province" : "BKK"
        },
        "age" : 30,
        "status" : "I"
}
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

IN

```
db.Customer .find( { status: { $in: [ "A", "D" ] } } )
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.customer.find({status:{$in: ["A","D"]}}).pretty()
{
        "_id" : ObjectId("5d8e34411c9d440000ea487f"),
        "cid" : "002",
        "name" : "Bob",
        "address" : {
                "no" : "1",
                "road" : "TU street",
                "province" : "Phathum",
                "zipcode" : "12020"
        },
        "age" : 20,
        "status" : "A"
}
```

5. QUERY on Complex fields

NESTED FIELDS OR ARRAY FIELDS

Complex Fields

Nested Field

Ex. Size

Array Field

Ex. Tags

Command Prompt - mongo "mongodb+srv://dmmcluster.f1uoi.gcp.mongodb.net/shop101" --username st_dmm --password st_dmm

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.inventory.find().pretty()
{
        "_id" : ObjectId("5f6cb1e662783426b926a1b7"),
        "item" : "canvas",
        "qty" : 100,
        "price" : 500,
        "tags" : [
                "cotton"
        ],
        "size" : {
                "h" : 28,
                "w" : 35.5,
                "uom" : "cm"
        }
}
{
        "_id" : ObjectId("5f6cb1e662783426b926a1b8"),
        "item" : "pen",
        "qty" : 25,
        "price" : 200,
        "tags" : [
                "red"
        ],
        "size" : {
                "h" : 14,
                "w" : 1,
                "uom" : "cm"
        }
}
```

Query Nested Field

Uses **dot notation** to access fields in an embedded document:

- Select all documents where uom of size is in “in” unit

```
db.inventory.find( { "size.uom": "in" } )
```

- selects items where their height less than 15 CM.

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.inventory.find( { "size.h": { $lt: 15 }, "size.uom": "cm" } ).pretty()
{
        "_id" : ObjectId("5d8e3f03ba0c8c019a09414f"),
        "item" : "pen",
        "qty" : 25,
        "price" : 200,
        "tags" : [
                "red"
        ],
        "size" : {
                "h" : 14,
                "w" : 1,
                "uom" : "cm"
        }
}
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

Query Array Field

- selects items where their tags has “blue”.

```
db.inventory.find({tags : "blue"}).pretty()
```

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.inventory.find({tags : "blue"}).pretty()
{
        "_id" : ObjectId("5f6cb1e662783426b926a1ba"),
        "item" : "mousepad",
        "qty" : 25,
        "price" : 29,
        "tags" : [
                "gel",
                "blue"
        ],
        "size" : {
                "h" : 19,
                "w" : 22.85,
                "uom" : "in"
        }
}
```

6. Aggregate Operations

AGGREGATION OPERATIONS GROUP VALUES FROM MULTIPLE DOCUMENTS TOGETHER AND CAN PERFORM A VARIETY OF OPERATIONS ON THE GROUPED DATA TO RETURN A SINGLE RESULT.

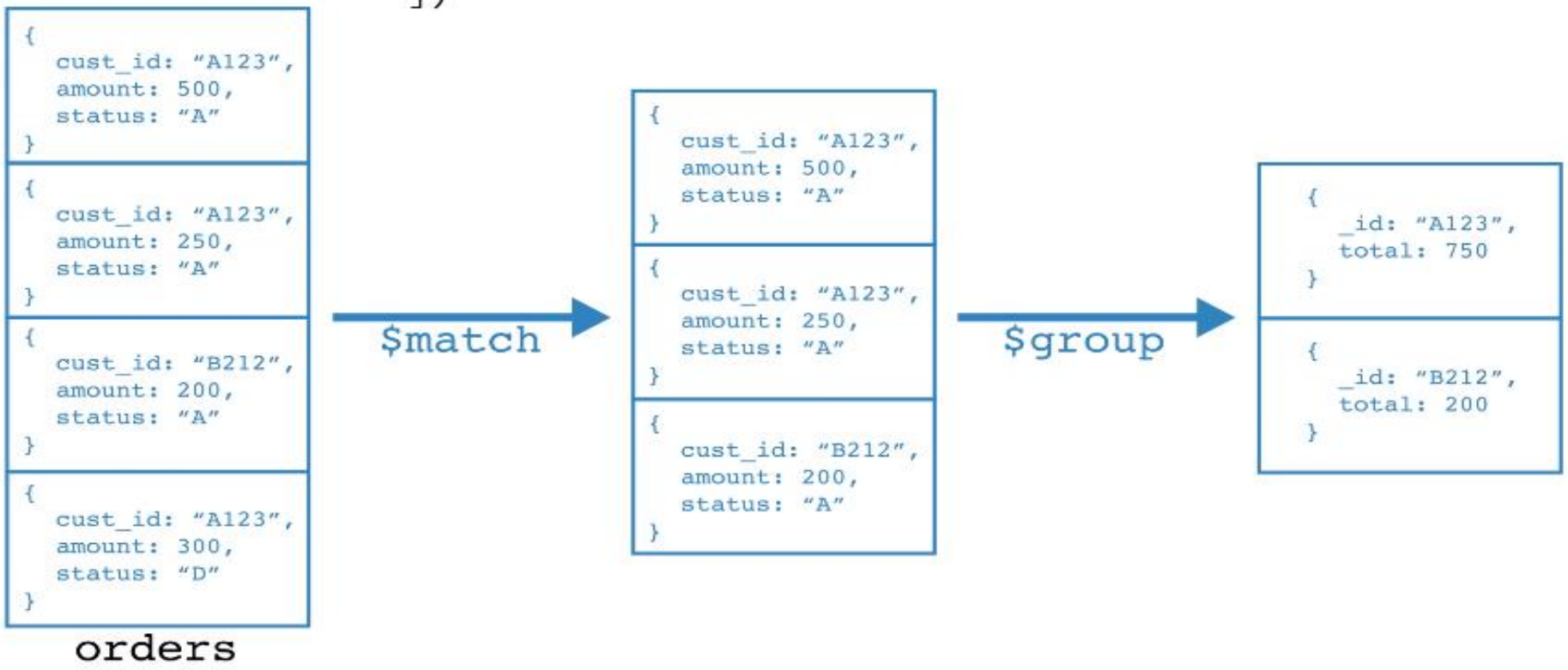
SQL Terms, Functions, and Concepts	MongoDB Aggregation Operators
WHERE	`$match`
GROUP BY	`$group`
HAVING	`$match`
SELECT	`$project`
ORDER BY	`$sort`
LIMIT	`$limit`
SUM()	`$sum`
COUNT()	`$sum` `$sortByCount`
join	`$lookup`

EX. SQL -> Select sum(amount) from orders where status = “A” group by cust_id

Mongo -> ???

Collection

```
db.orders.aggregate( [
  $match stage ➡ { $match: { status: “A” } },
  $group stage ➡ { $group: { _id: “$cust_id”,total:  {$sum: “$amount” } } }
                 ])
```



Sales documents

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.sales.find().pretty()
{
        "_id" : ObjectId("5d8f62648fb57f941738209b"),
        "item" : [
                {
                        "pid" : "canvas",
                        "qty" : 1,
                        "price" : 500
                },
                {
                        "pid" : "pen",
                        "qty" : 1,
                        "price" : 200
                }
        ],
        "customer" : "Lisa",
        "total" : 700,
        "couponused" : true
}
{
```

```
}
{
        "_id" : ObjectId("5d8f63588fb57f941738209c"),
        "item" : [
                {
                        "pid" : "mousepad",
                        "qty" : 1,
                        "price" : 29
                }
        ],
        "customer" : "Bob",
        "total" : 29,
        "couponused" : true
}
{
        "_id" : ObjectId("5d9027dd8fb57f941738209d"),
        "item" : [
                {
                        "pid" : "mat",
                        "qty" : 1,
                        "price" : 99
                }
        ],
        "customer" : "Bob",
        "total" : 99,
        "couponused" : false
}
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

SUM

```
db.sales.aggregate([
                    { $group: { _id: "$customer", total: { $sum: "$total" } } }
                   ])
```

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([
            { $group: { _id: "$customer", total: { $sum: "$total" } } }
...                                   ])
{ "_id" : "Bob", "total" : 128 }
{ "_id" : "Lisa", "total" : 700 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

```
db.sales.aggregate([ { $match: { couponused:true } },
                     { $group: { _id: "$customer", total: { $sum: "$total" } } }
                   ])
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.sales.aggregate([    { $match: { couponused:true } },
  { $group: { _id: "$customer", total: { $sum: "$total" } } } ])
{ "_id" : "Bob", "total" : 29 }
{ "_id" : "Lisa", "total" : 700 }
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

COUNT

db.sales.aggregate([{ $group: { _id: "$customer", total: { $sum: 1} } }
])

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([
{ $group: { _id: "$customer", total: { $sum: 1} } } ])
{ "_id" : "Lisa", "total" : 1 }
{ "_id" : "Bob", "total" : 2 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

Note:

FIND() also has COUNT() function

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.sales.find({customer:"Lisa"}).count()
1
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.sales.find({customer:"Bob"}).count()
2
```

Average, Min, Max

```
db.sales.aggregate([
                    { $group: { _id: "$customer", average: { $avg: "$total" } } }
                   ])

db.sales.aggregate([
                    { $group: { _id: "$customer", minimum: { $min: "$total" } } }
                   ])

db.sales.aggregate([
                    { $group: { _id: "$customer", maximum: { $max: "$total" } } }
                   ])
```

Result

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([
{ $group: { _id: "$customer", minimum: { $min: "$total" } } } ])
{ "_id" : "Bob", "minimum" : 29 }
{ "_id" : "Lisa", "minimum" : 700 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([
{ $group: { _id: "$customer", maximum: { $max: "$total" } } } ])
{ "_id" : "Bob", "maximum" : 99 }
{ "_id" : "Lisa", "maximum" : 700 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([
{ $group: { _id: "$customer", average: { $avg: "$total" } } } ])
{ "_id" : "Bob", "average" : 64 }
{ "_id" : "Lisa", "average" : 700 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

HAVING

```
db.sales.aggregate([{ $group: { _id: "$customer", total: { $sum: "$total" } } } ,
                    {$match: {total: {$gt: 150 }}}
                   ])
```

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.sales.aggregate
([{ $group: { _id: "$customer", total: { $sum: "$total" } } } ,
 {$match: {total: {$gt: 150 }}}])
{ "_id" : "Lisa", "total" : 700 }
MongoDB Enterprise Cluster0-shard-0:PRIMARY>
```

Sort and Limit in Aggregate

```
db.sales.aggregate([
{$group:{_id:"$customer",total: {$sum:"$total"}}},
{ $sort: { total: -1 } }
])
```

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([{$group:
{_id:"$customer",total: {$sum:"$total"}}},{ $sort: { total: -1 } }])
{ "_id" : "Lisa", "total" : 700 }
{ "_id" : "Bob", "total" : 128 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([{$group:
{_id:"$customer",total: {$sum:"$total"}}},{ $sort: { total:1 } }])
{ "_id" : "Bob", "total" : 128 }
{ "_id" : "Lisa", "total" : 700 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

```
db.sales.aggregate([
{$group:{_id:"$customer",total:
{$sum:"$total"}}},
{ $sort: { total: -1 }},
{$limit:1}
])
```

```
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY> db.sales.aggregate([{$group:
{_id:"$customer",total: {$sum:"$total"}}}, { $sort: { total: -1 }},{$limit:1}
])
{ "_id" : "Lisa", "total" : 700 }
MongoDB Enterprise atlas-omm7a6-shard-0:PRIMARY>
```

7. JOIN Operations

JOIN

- Syntax

```
{
   $lookup:
     {
       from: <collection to join>,
       localField: <field from the input documents>,
       foreignField: <field from the documents of the "from" collection>,
       as: <output array field>
     }
}
```

Join Customer vs. Sale

```
db.sales.aggregate([
  {
    $lookup:
      {
        from: "customer",
        localField: "customer",
        foreignField: "name",
        as: "cust_info"
      }
  }
]).pretty()
```

```
{
        "_id" : ObjectId("5d8f62648fb57f941738209b"),
        "item" : [
                {
                        "pid" : "canvas",
                        "qty" : 1,
                        "price" : 500
                },
                {
                        "pid" : "pen",
                        "qty" : 1,
                        "price" : 200
                }
        ],
        "customer" : "Lisa",
        "total" : 700,
        "couponused" : true,
        "cust_info" : [
                {
                        "_id" : ObjectId("5d8e32391c9d440000ea487d"),
                        "cid" : "001",
                        "name" : "Lisa",
                        "address" : {
                                "no" : "99",
                                "road" : "main street",
                                "province" : "BKK"
                        },
                        "age" : 30,
                        "status" : "I"
                }
        ]
}
```

Lab 6 Assignment

SUBMISSION SYSTEM: GOOGLE CLASSROOM

Mongo Shell download

- **Windows**

https://downloads.mongodb.org/windows/mongodb-shell-windows-x86_64-4.4.1.zip

- **MACOS**

brew install mongodb/brew/mongodb-community-shell

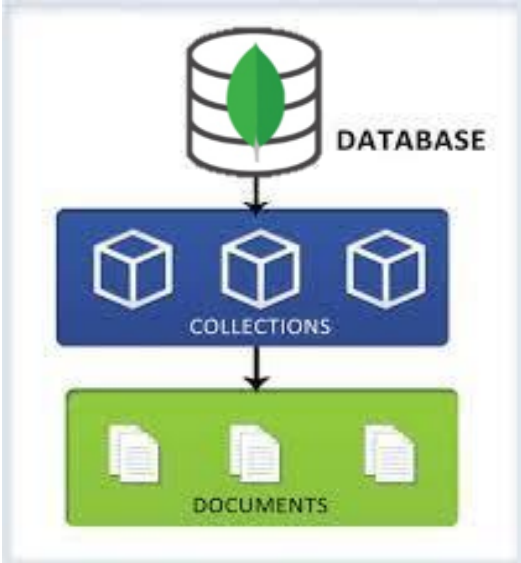
- **Ubuntu 16.04,18.04,20.04**

https://downloads.mongodb.org/linux/mongodb-shell-linux-x86_64-ubuntu1604-4.4.1.tgz
https://downloads.mongodb.org/linux/mongodb-shell-linux-x86_64-ubuntu1804-4.4.1.tgz
https://downloads.mongodb.org/linux/mongodb-shell-linux-x86_64-ubuntu2004-4.4.1.tgz

1. Connect to RDBProject

- 2 Databases
  - Shop101
  - RDBProject
- RDBProject
  - Feedback
- Username
  - st_dmm/st_dmm
  - onlne/online
  - offline/offline

DDM2020 MongoDB Cloud



MongoDB Shell

```
> db.collection.insert({company:"10gen", product:"MongoDB"})
>
> db.collection.findOne()
{
        "_id": ObjectId("5106c1c2fc629bfe52792e86"),
        "company": "10gen"
        "product": "MongoDB"
}
```

mongo "mongodb+srv://dmmcluster.fluoi.gcp.mongodb.net/<dbname>"
 --username <username>
--password <password>

i.e., mongo "mongodb+srv://dmmcluster.fluoi.gcp.mongodb.net/RDBProject" --username st_dmm --password st_dmm

2. Peer Review -> JSON Documents

	A	B	C	D	E	F	G	H
1				Peer Feedback Form				
2		Reviewer Name	xx	xx	xx			
3		Meeting Room#	xx					
4								
5								
6	Team#	Project Title	Member#	Presenter Name	Time Taken (min)	Summary of Presentation	Strengths	Suggestions
7	19	Bike sharing	19-1	Mr. XYZ	15	good presentation	precise and easy to understand	practice presentation skills
8	20							
9								
10								
11								
12								
13								


```
{
"member_id": "19-1",
"presenter_name": "Mr. XYZ",
"project_id" : "19",
"project_name": "Bike sharing",
"summary_of_presentation": "the presentation was well prepared and_ ",
"strengths": [ "precise", "easy to understand" ],
"suggestions": ["practice presentation skills" ]
...
}
```

3. Insert into feedback table in RDBProject MongoDB

4. Write MongoDB command to retrieve Task1-3 information

5. Submit the commands and the capture images in DMM GOOGLE CLASSROOM

Deadline TUE SEP 29 11:59 AM

LAB 5: REDIS	SUBMIT on DSAI
LAB 6: MONGODB	SUBMIT on GOOGLE CLASSROOM
LAB 7-10	SUBMIT on GOOGLE CLASSROOM

References

- https://docs.mongodb.com/manual/
- https://docs.mongodb.com/manual/reference/sql-comparison/#examples
- https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/index.html#lookup-single-equality

Good Luck for the Midterm Exams

SEE YOU NEXT WEDNESDAY

Thank you.