CryptoMinds MongoDB

By Darshan Vasani

✏ MongoDB + Mongoose Notes

- Complete Guide To MongoDB And Mongoose By CryptoMinds: Everything You Need To Now !

MongoDB Shell	MongoDb GUI → [Compass]	Mongoose

✏MongoDb Shell → CLI

✏Introduction to MongoDB Shell:

- The MongoDB Shell is a command-line interface that allows you to interact with the MongoDB database.
- It provides a JavaScript-based environment where you can execute commands and perform operations.

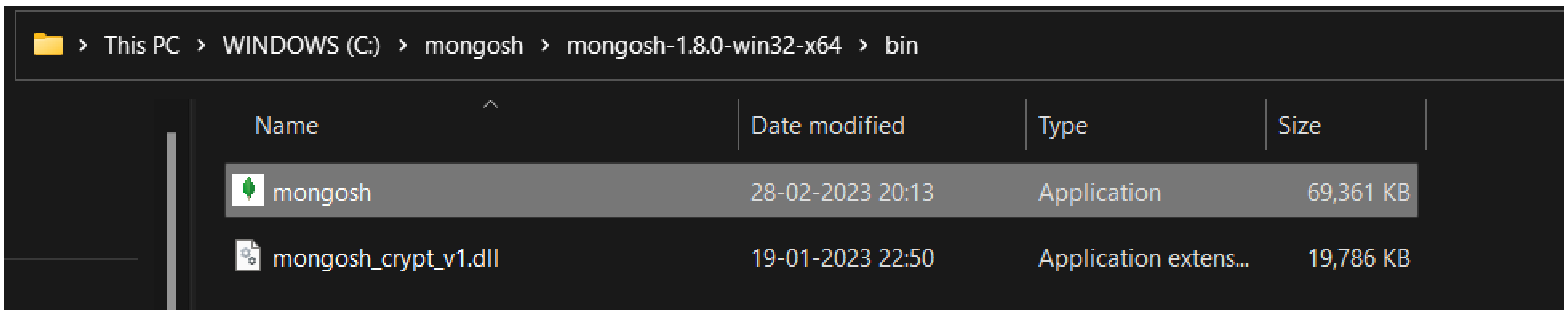
✏ Starting the MongoDB Shell:

- Open your terminal or command prompt.
- Navigate to the MongoDB installation directory.
- Run the command mongo to start the MongoDB Shell.

→ Go to C Drive Where Mongosh Is Unzipped

→ mongosh -> mongosh-1.8.0-win32-x64 -> bin

→ Run -> Mongosh Application -> Hit Enter


This PC › WINDOWS (C:) › mongosh › mongosh-1.8.0-win32-x64 › bin

Name	Date modified	Type	Size
mongosh	28-02-2023 20:13	Application	69,361 KB
mongosh_crypt_v1.dll	19-01-2023 22:50	Application extens...	19,786 KB

Shell Commands

Shell Commands

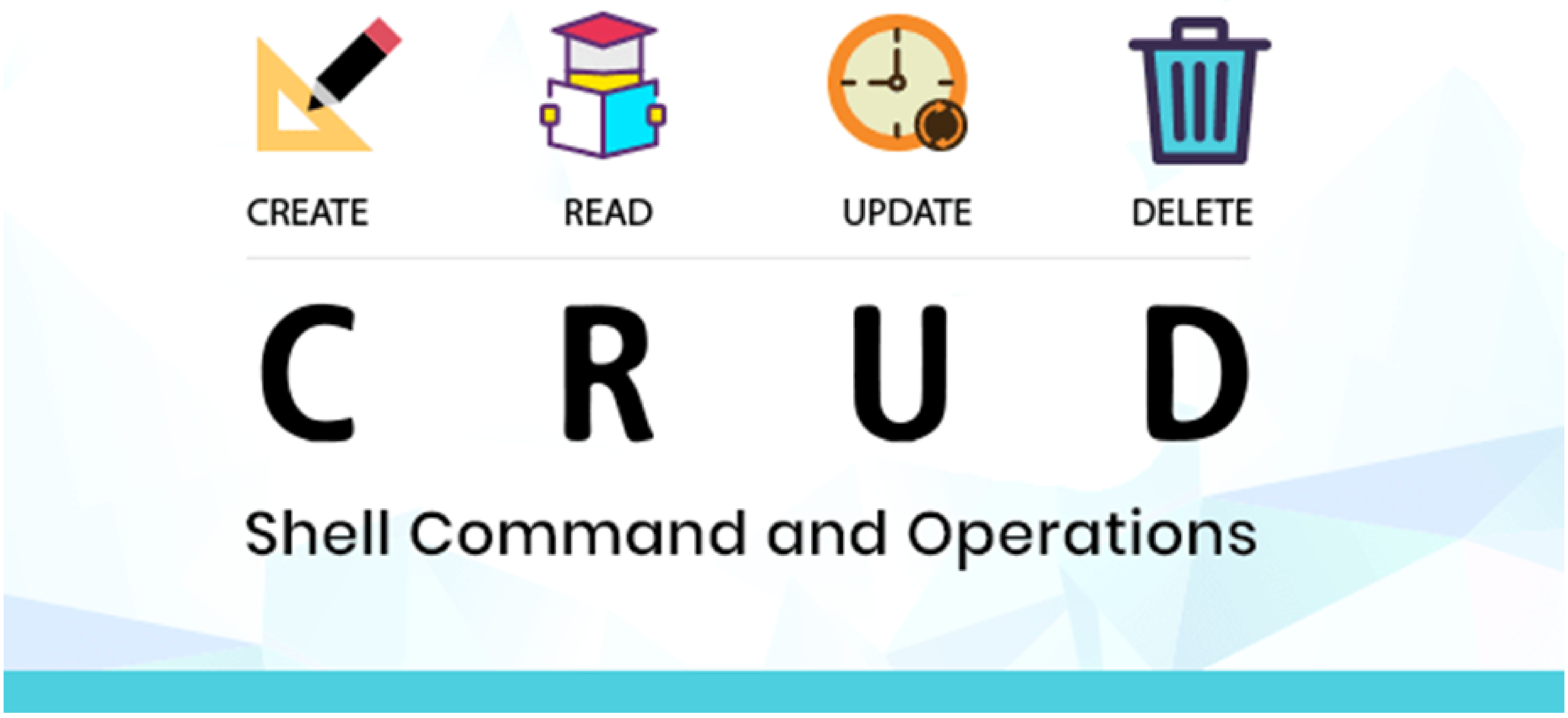
✏Basic Shell Command

- show dbs: Lists all the available databases.

    → To Visible in this command, DB must have at least one collection

- use <database>: Switches to the specified database.
- db: Returns the current database being used.
- show collections: Lists all the collections in the current database.
- db.<collection>.find(): Retrieves all documents from the specified collection.

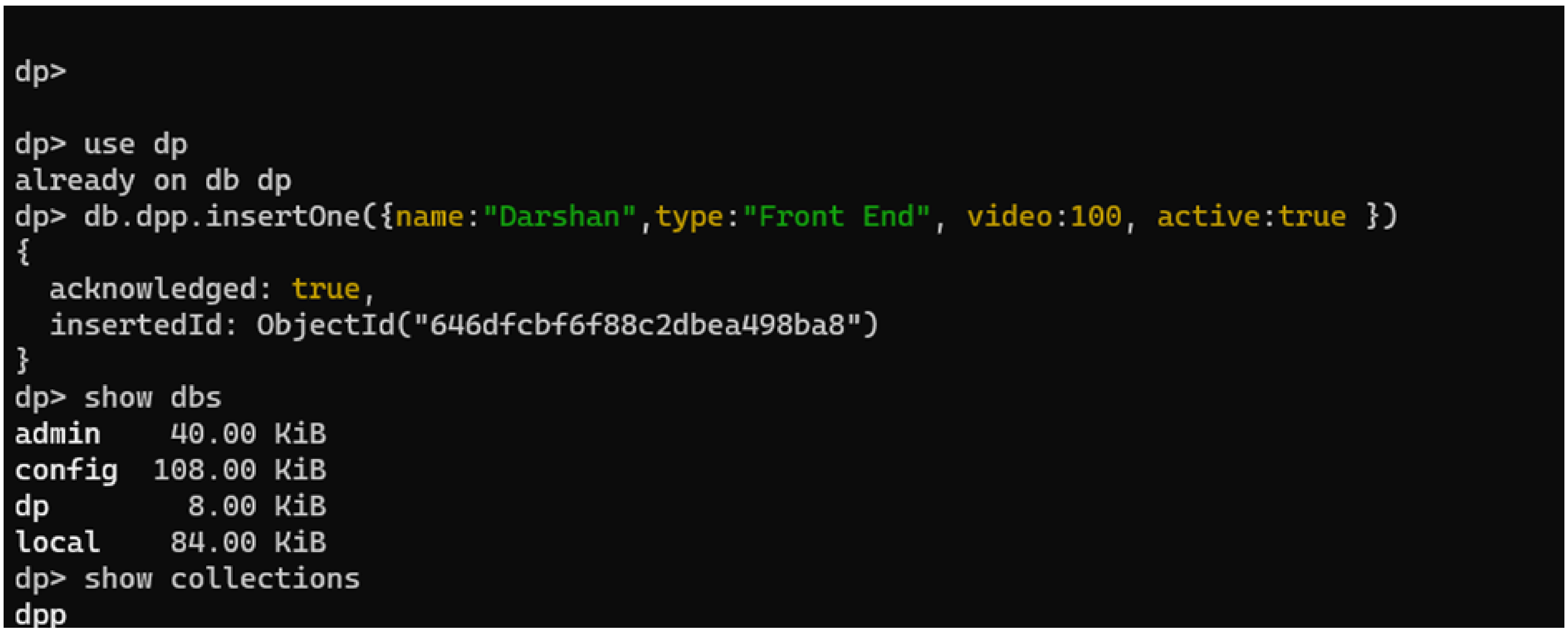
CRUD Operations



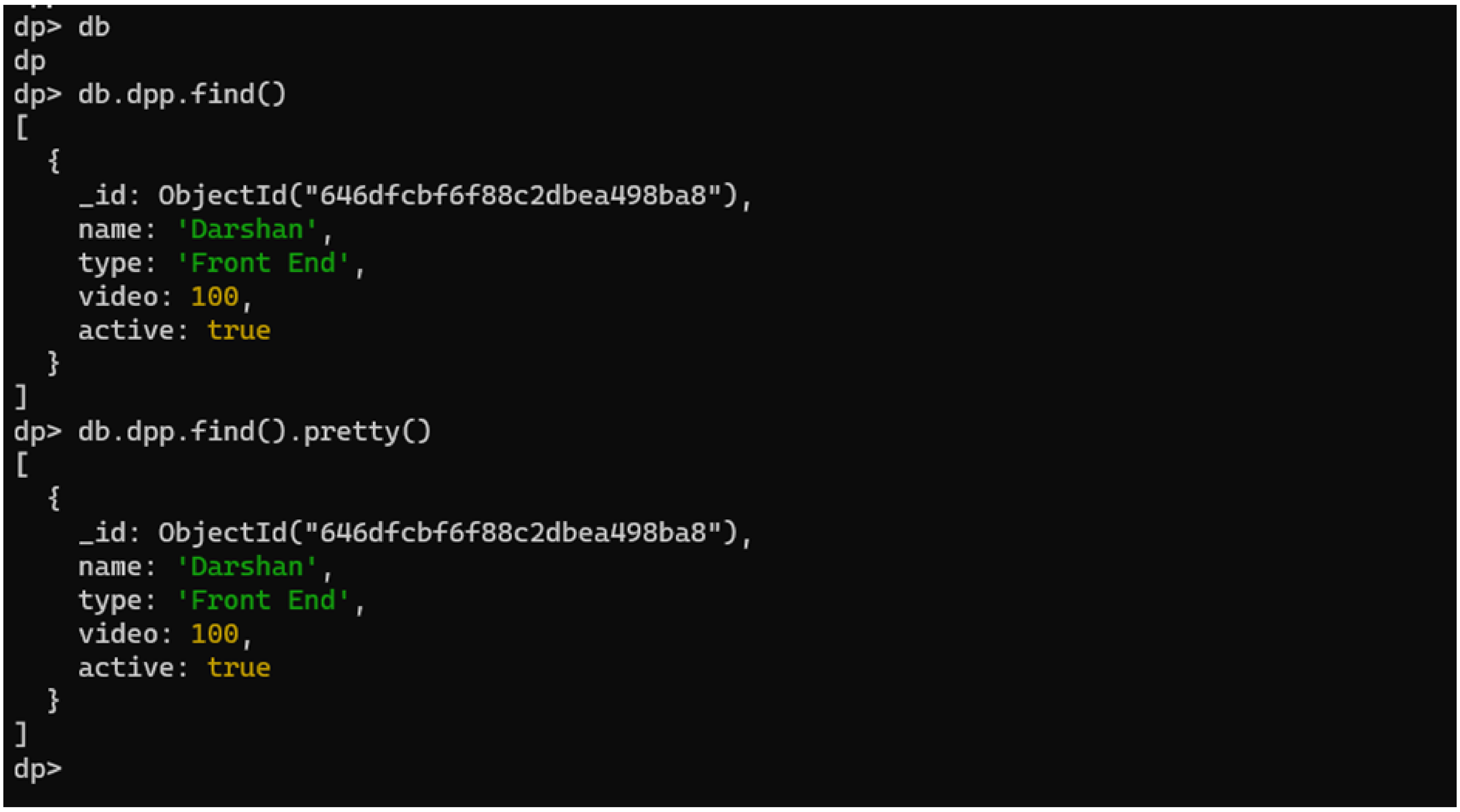
✏Create (Insert) Operation: 😎

- Syntax: db.collectionName.insert(document)
- Example: db.users.insert({ name: "John", age: 30, email: "john@example.com" })
- This will create a new document in the "users" collection with the specified fields and values.

- **insertOne(😎)**
- different between find and find pretty is when you use cmd you can see result.

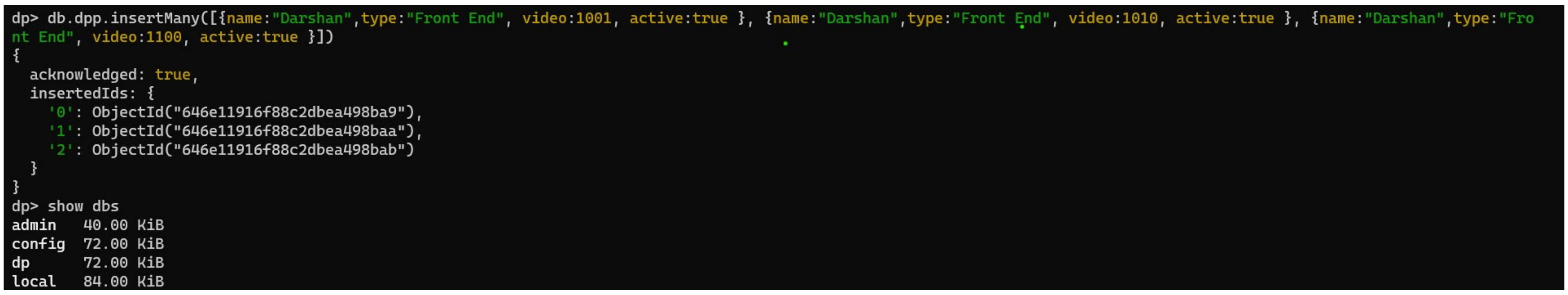


```
dp>

dp> use dp
already on db dp
dp> db.dpp.insertOne({name:"Darshan",type:"Front End", video:100, active:true })
{
  acknowledged: true,
  insertedId: ObjectId("646dfcbf6f88c2dbea498ba8")
}
dp> show dbs
admin     40.00 KiB
config   108.00 KiB
dp         8.00 KiB
local     84.00 KiB
dp> show collections
dpp
```



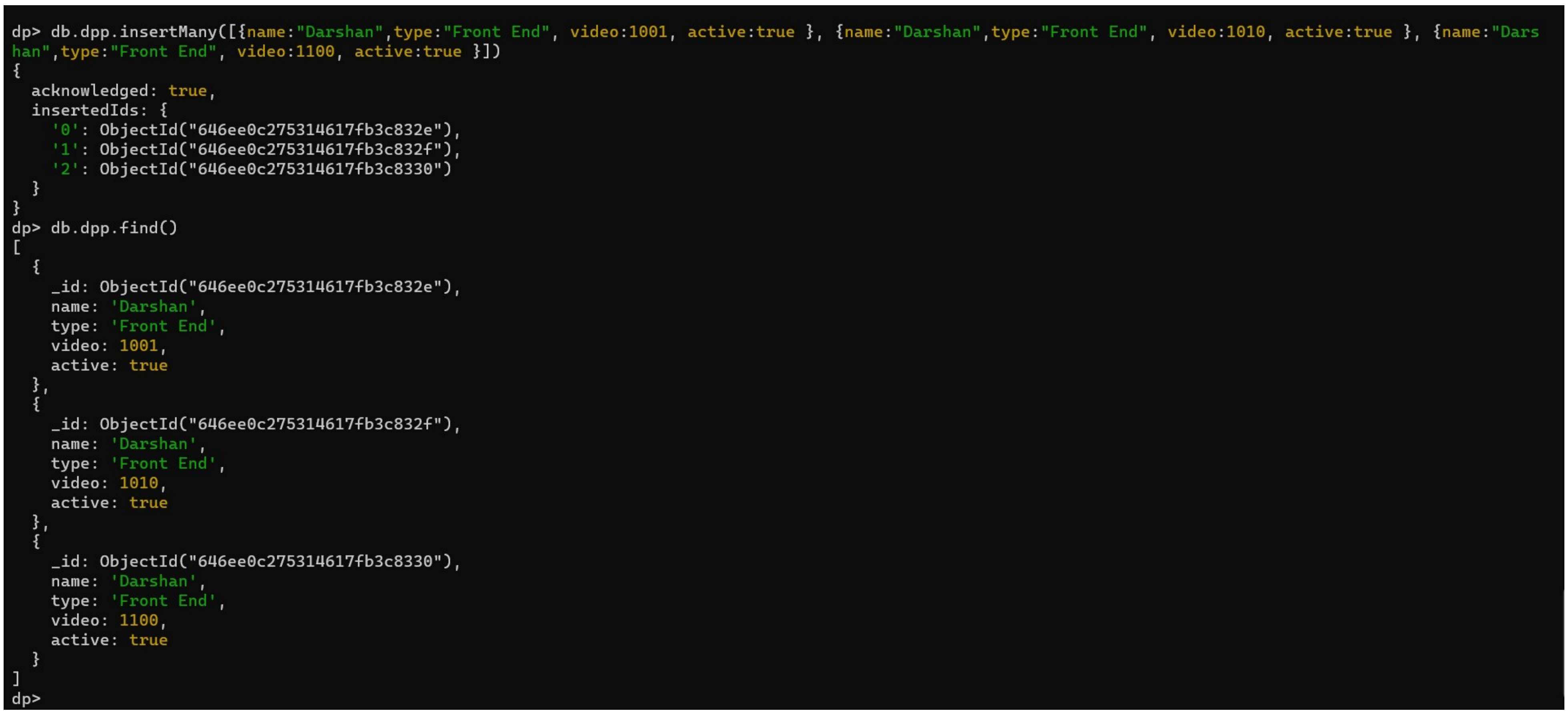
```
dp> db
dp
dp> db.dpp.find()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Front End',
    video: 100,
    active: true
  }
]
dp> db.dpp.find().pretty()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Front End',
    video: 100,
    active: true
  }
]
dp>
```

- **InsertMany(😎)**



```
dp> db.dpp.insertMany([{name:"Darshan",type:"Front End", video:1001, active:true }, {name:"Darshan",type:"Front End", video:1010, active:true }, {name:"Darshan",type:"Fro
nt End", video:1100, active:true }])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("646e11916f88c2dbea498ba9"),
    '1': ObjectId("646e11916f88c2dbea498baa"),
    '2': ObjectId("646e11916f88c2dbea498bab")
  }
}
dp> show dbs
admin   40.00 KiB
config  72.00 KiB
dp      72.00 KiB
local   84.00 KiB
```

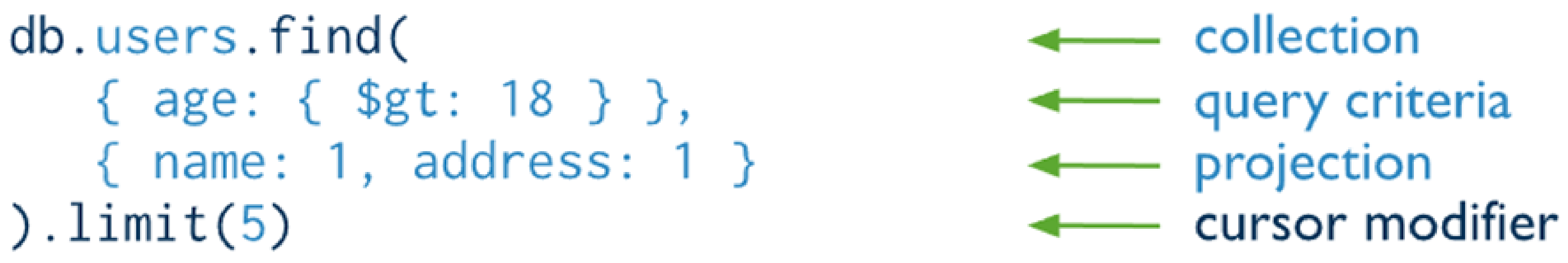
- **Output :**



```
dp> db.dpp.insertMany([{name:"Darshan",type:"Front End", video:1001, active:true }, {name:"Darshan",type:"Front End", video:1010, active:true }, {name:"Dars
han",type:"Front End", video:1100, active:true }])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("646ee0c275314617fb3c832e"),
    '1': ObjectId("646ee0c275314617fb3c832f"),
    '2': ObjectId("646ee0c275314617fb3c8330")
  }
}
dp> db.dpp.find()
[
  {
    _id: ObjectId("646ee0c275314617fb3c832e"),
    name: 'Darshan',
    type: 'Front End',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646ee0c275314617fb3c832f"),
    name: 'Darshan',
    type: 'Front End',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646ee0c275314617fb3c8330"),
    name: 'Darshan',
    type: 'Front End',
    video: 1100,
    active: true
  }
]
dp>
```

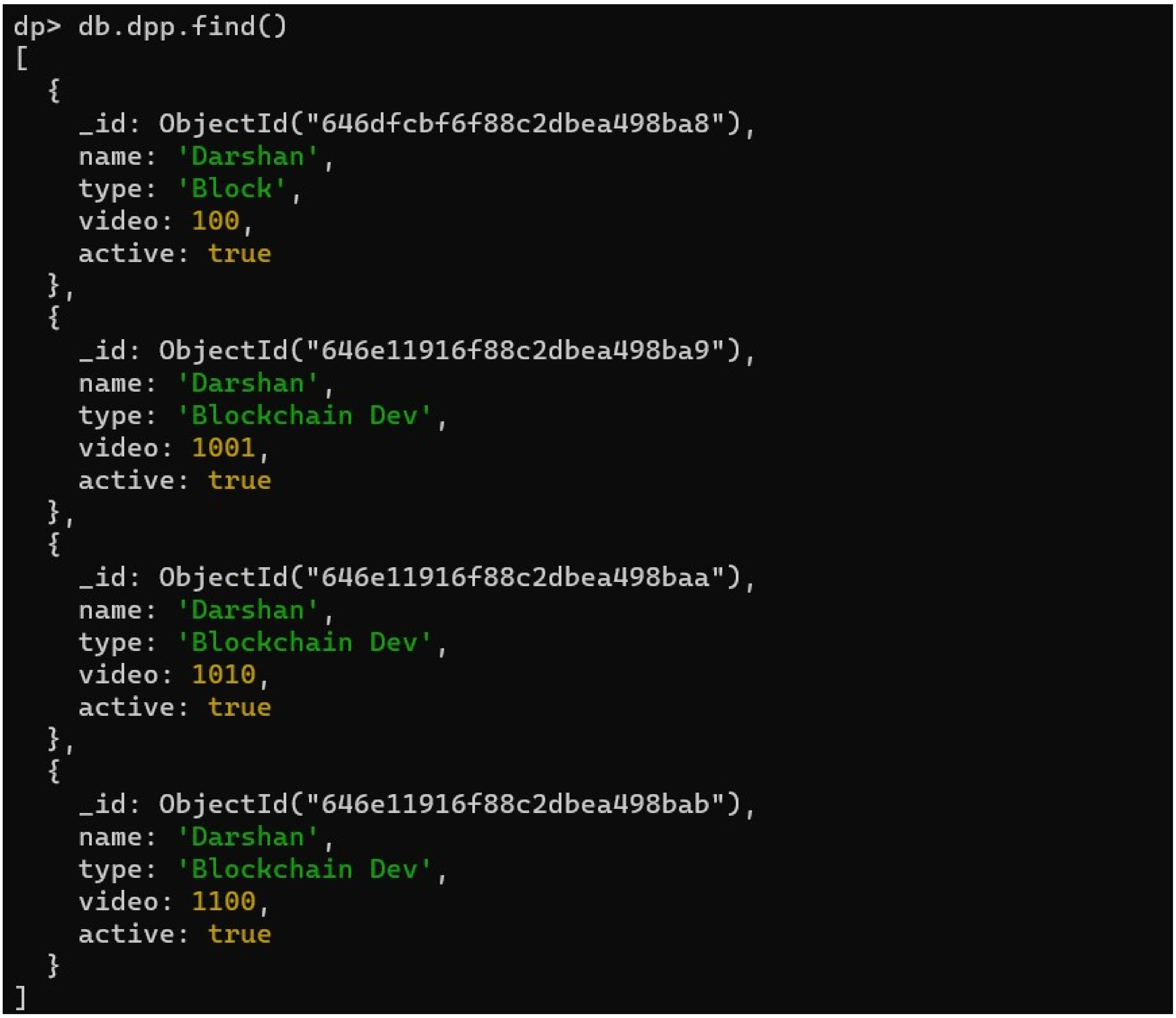
✏Read (Query) Operation:

- Syntax: db.collectionName.find(query, projection)
- Example: db.users.find({ age: { $gte: 25 } }, { name: 1, age: 1 })
- This will find all documents in the "users" collection where the age is greater than or equal to 25 and return only the "name" and "age" fields.



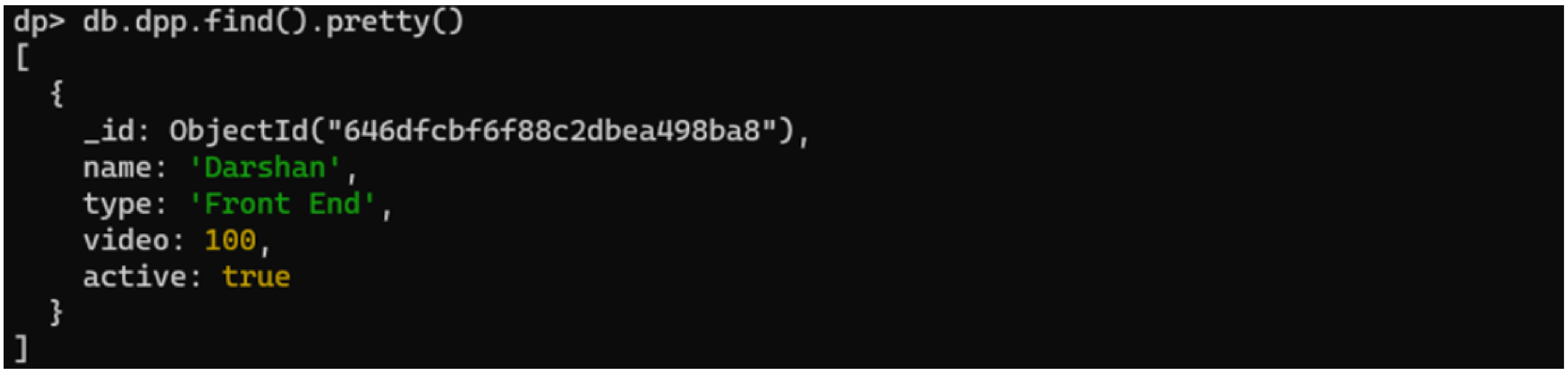
```
db.users.find(
   { age: { $gt: 18 } },
   { name: 1, address: 1 }
).limit(5)
```

1. Find all result of given collection.



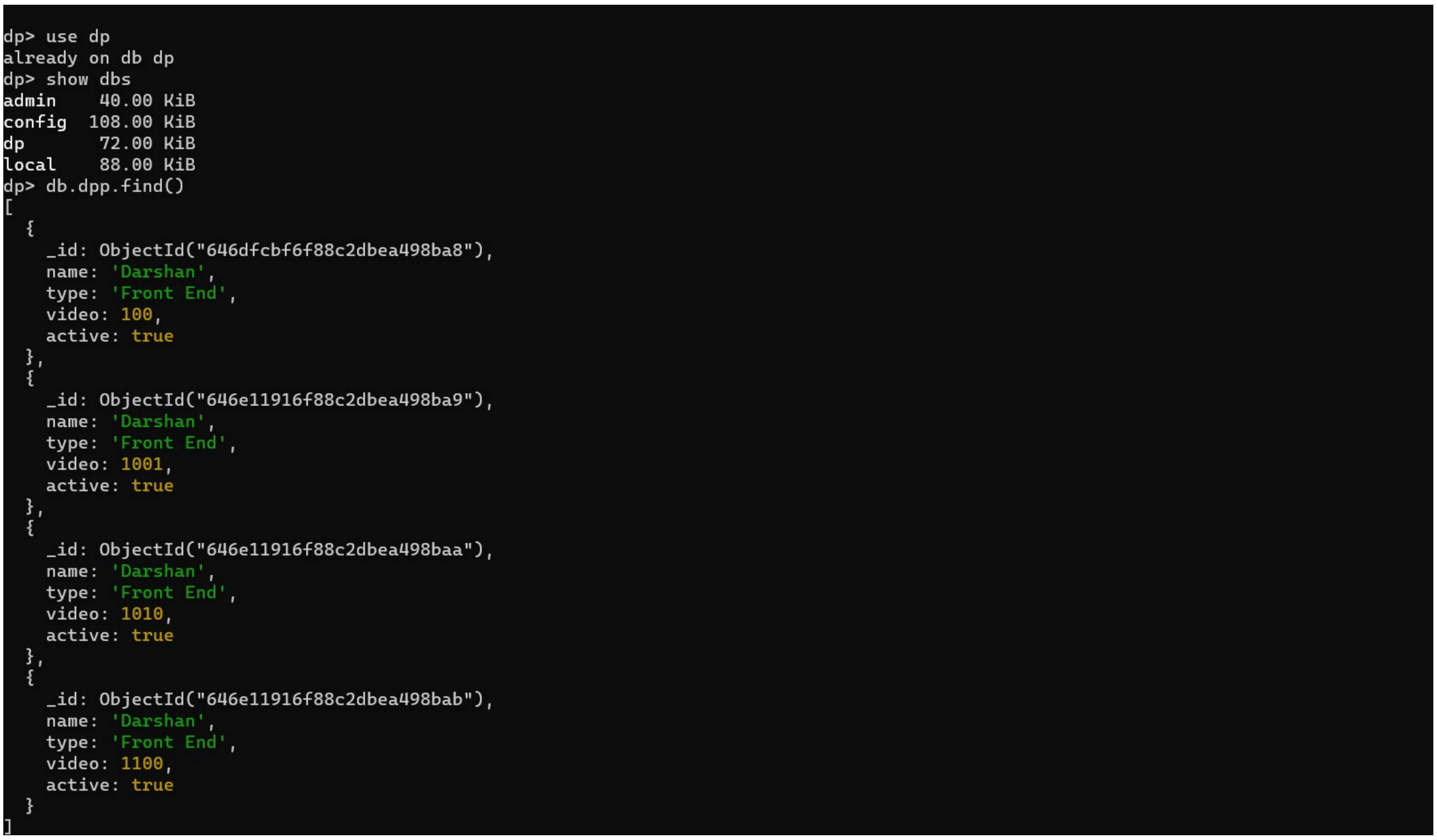
```
dp> db.dpp.find()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Block',
    video: 100,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498bab"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1100,
    active: true
  }
]
```

2 . Show result in pretty format.



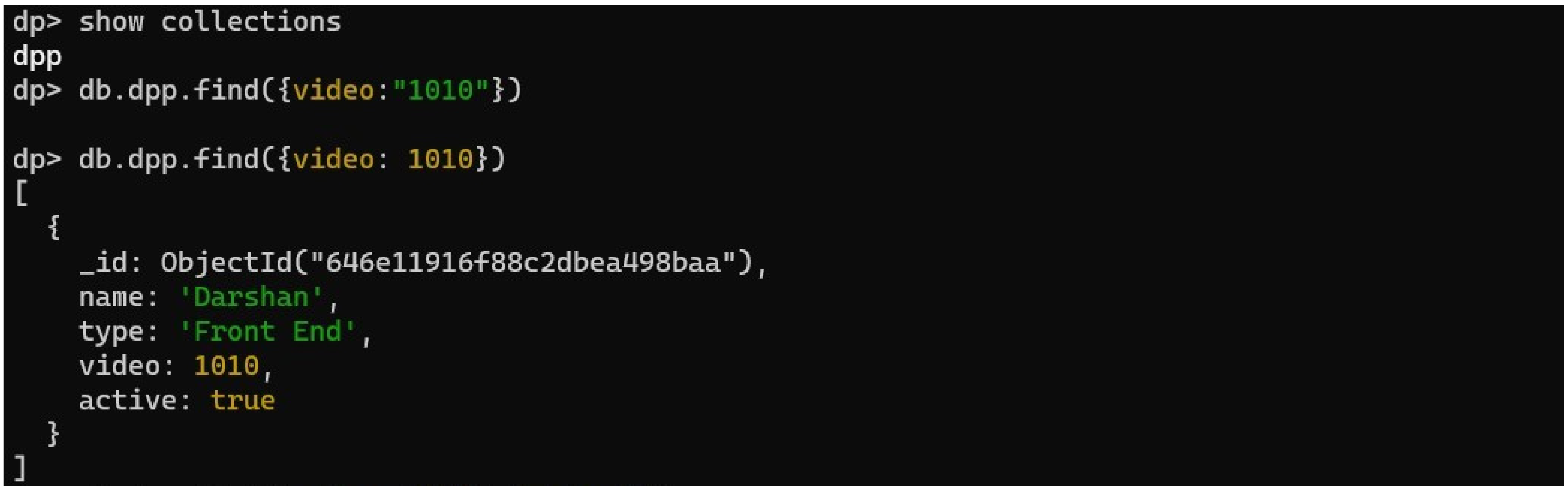
```
dp> db.dpp.find().pretty()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Front End',
    video: 100,
    active: true
  }
]
```

- **Example:2 → Find()**



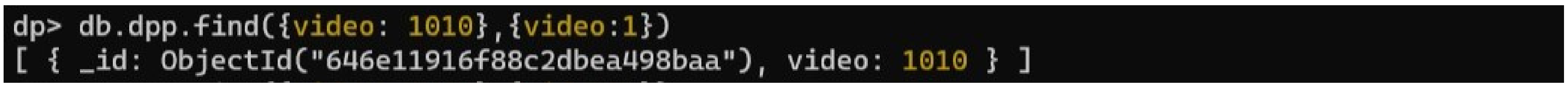
```
dp> use dp
already on db dp
dp> show dbs
admin    40.00 KiB
config  108.00 KiB
dp       72.00 KiB
local    88.00 KiB
dp> db.dpp.find()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Front End',
    video: 100,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Front End',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Front End',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498bab"),
    name: 'Darshan',
    type: 'Front End',
    video: 1100,
    active: true
  }
]
```

3. Get only video:1010 as output.



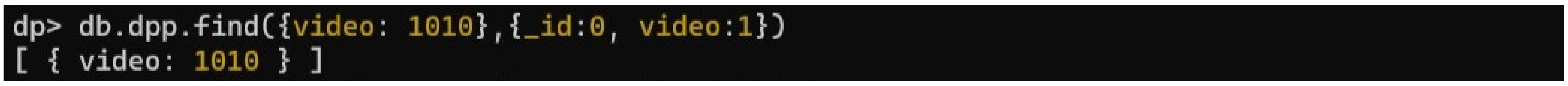
```
dp> show collections
dpp
dp> db.dpp.find({video:"1010"})

dp> db.dpp.find({video: 1010})
[
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Front End',
    video: 1010,
    active: true
  }
]
```

4. Get only video:1010 as output withonly name field.



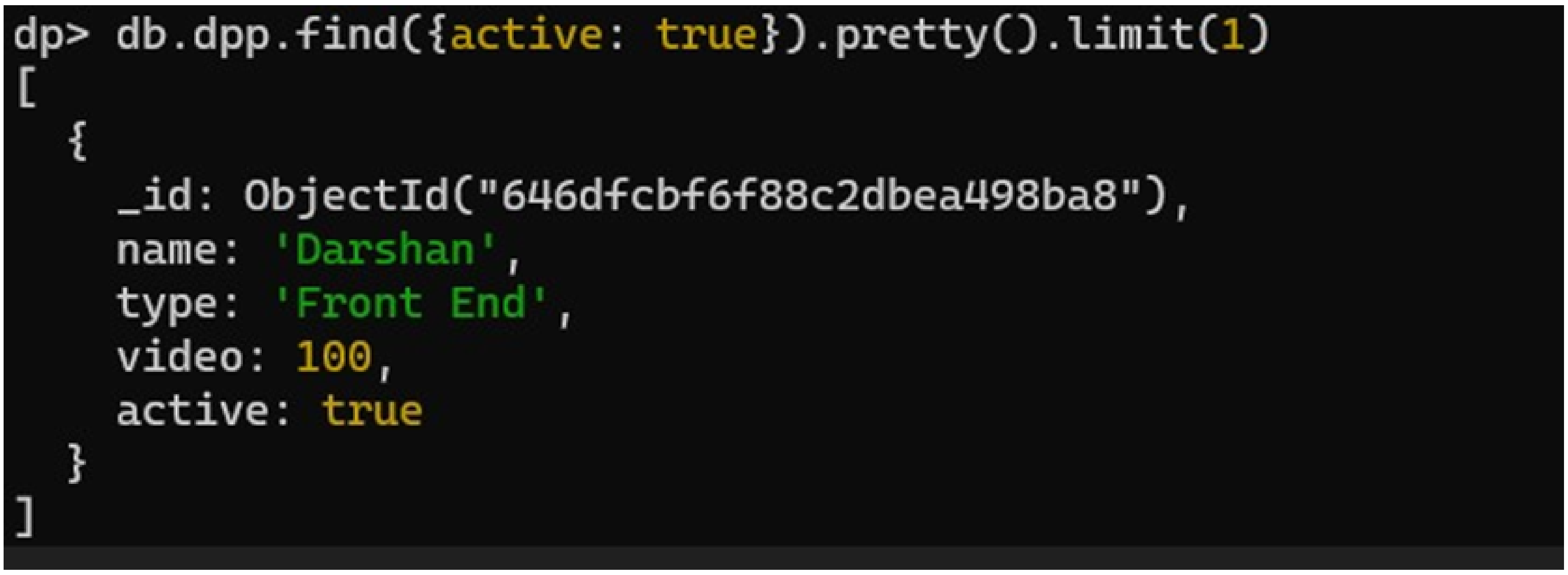
```
dp> db.dpp.find({video: 1010},{video:1})
[ { _id: ObjectId("646e11916f88c2dbea498baa"), video: 1010 } ]
```

5. Get only video:1010 as output withonly name field without id



```
dp> db.dpp.find({video: 1010},{_id:0, video:1})
[ { video: 1010 } ]
```

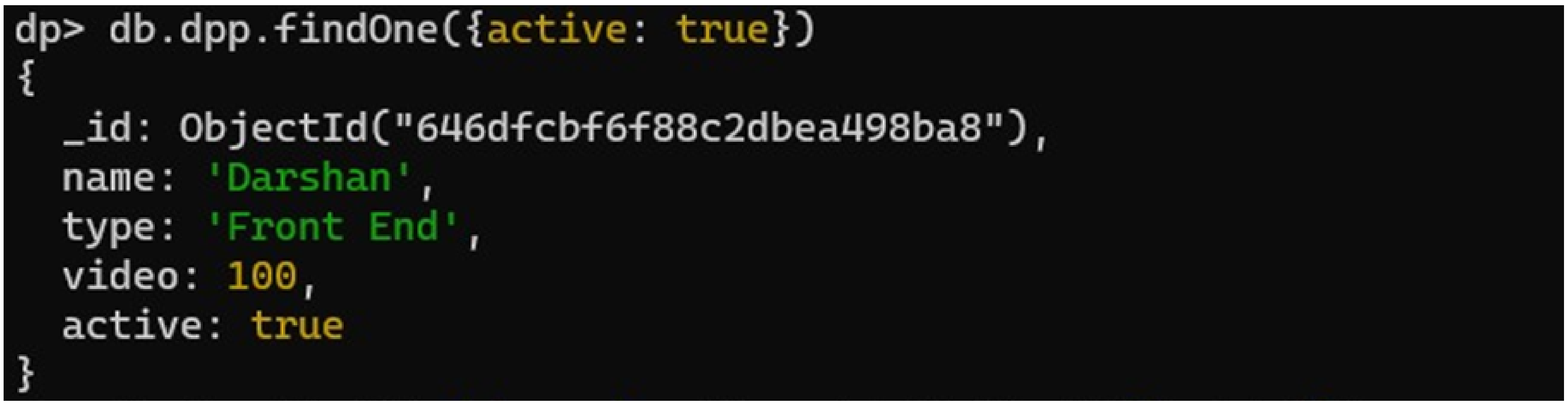
6. Set filter to "active":true and get only the first field with "active":true value.



```
dp> db.dpp.find({active: true}).pretty().limit(1)
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Front End',
    video: 100,
    active: true
  }
]
```

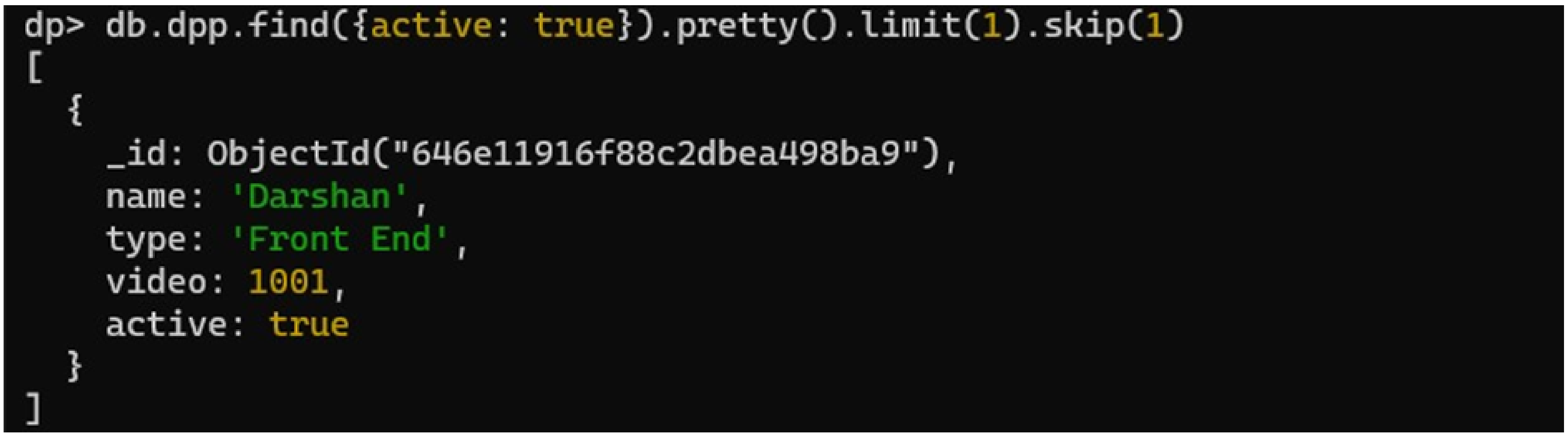
7. Do same question with different method.

- db.<collection>.findOne(): Retrieves a single document from the specified collection.



```
dp> db.dpp.findOne({active: true})
{
  _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
  name: 'Darshan',
  type: 'Front End',
  video: 100,
  active: true
}
```

8.Do as Same 6th question but at this time, get second field with "active":true by skipping the 1st field.

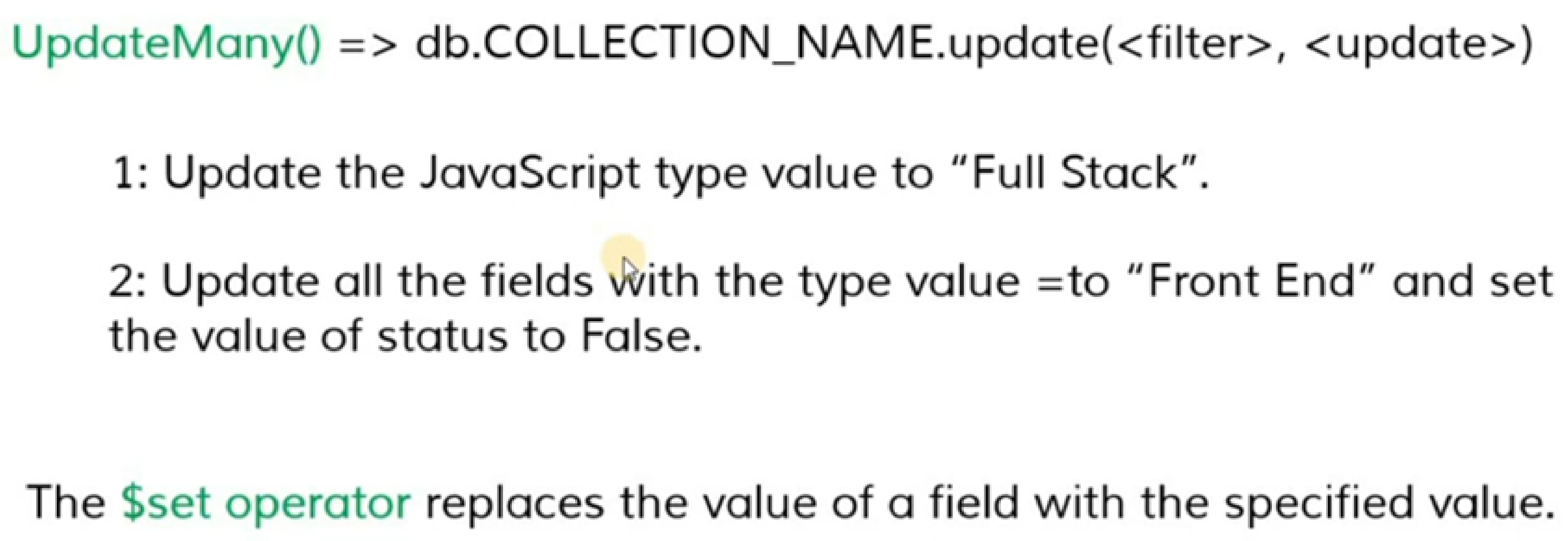


```
dp> db.dpp.find({active: true}).pretty().limit(1).skip(1)
[
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Front End',
    video: 1001,
    active: true
  }
]
```

✏**Update Operation:**

- Syntax: db.collectionName.update(query, update, options)
- Example: db.users.update({ name: "John" }, { $set: { age: 35 } })
- This will update the "age" field of the document(s) in the "users" collection where the name is "John" and set it to 35.

UpdateOne() => db.COLLECTION_NAME.updateOne(<filter>, <update>)



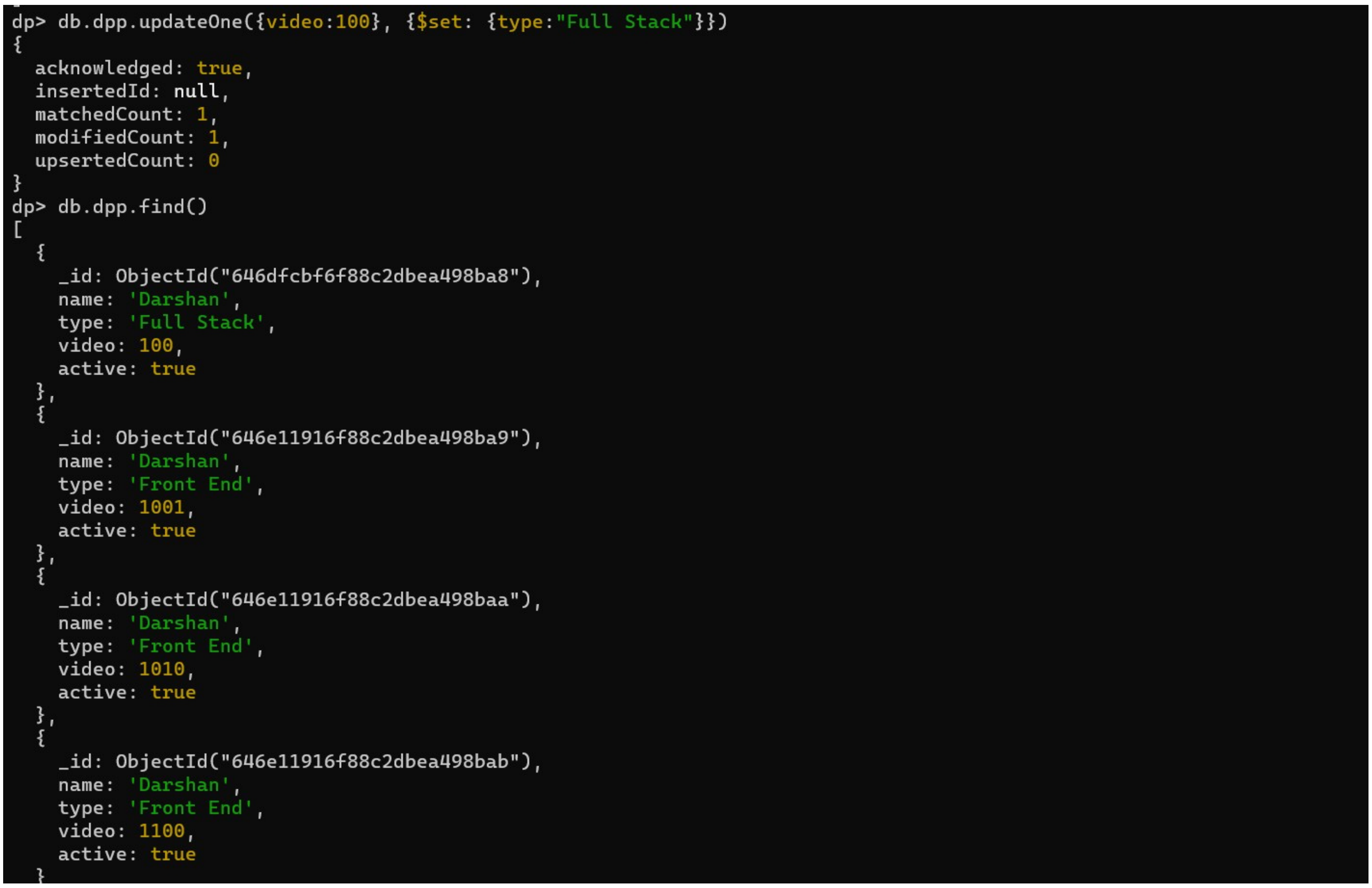
UpdateMany() => db.COLLECTION_NAME.update(<filter>, <update>)

1: Update the JavaScript type value to “Full Stack”.

2: Update all the fields with the type value =to “Front End” and set the value of status to False.

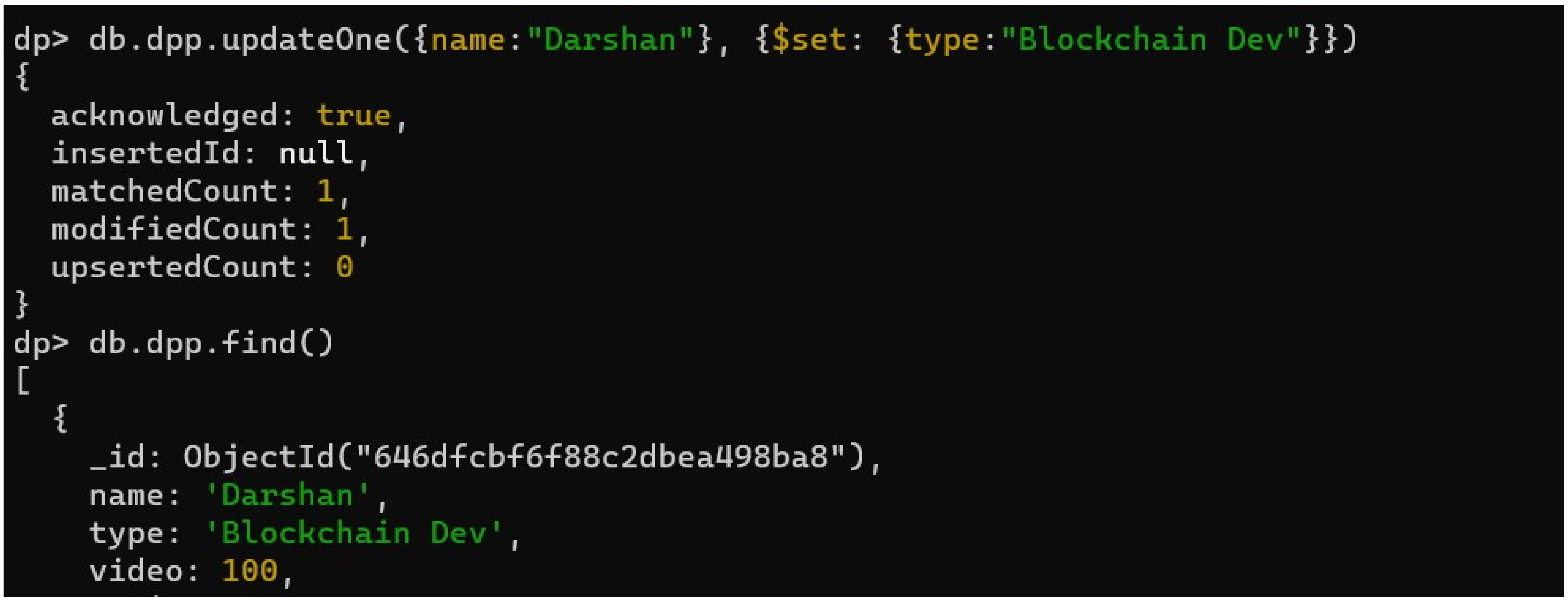
The $set operator replaces the value of a field with the specified value.

- **updateOne(😎)**

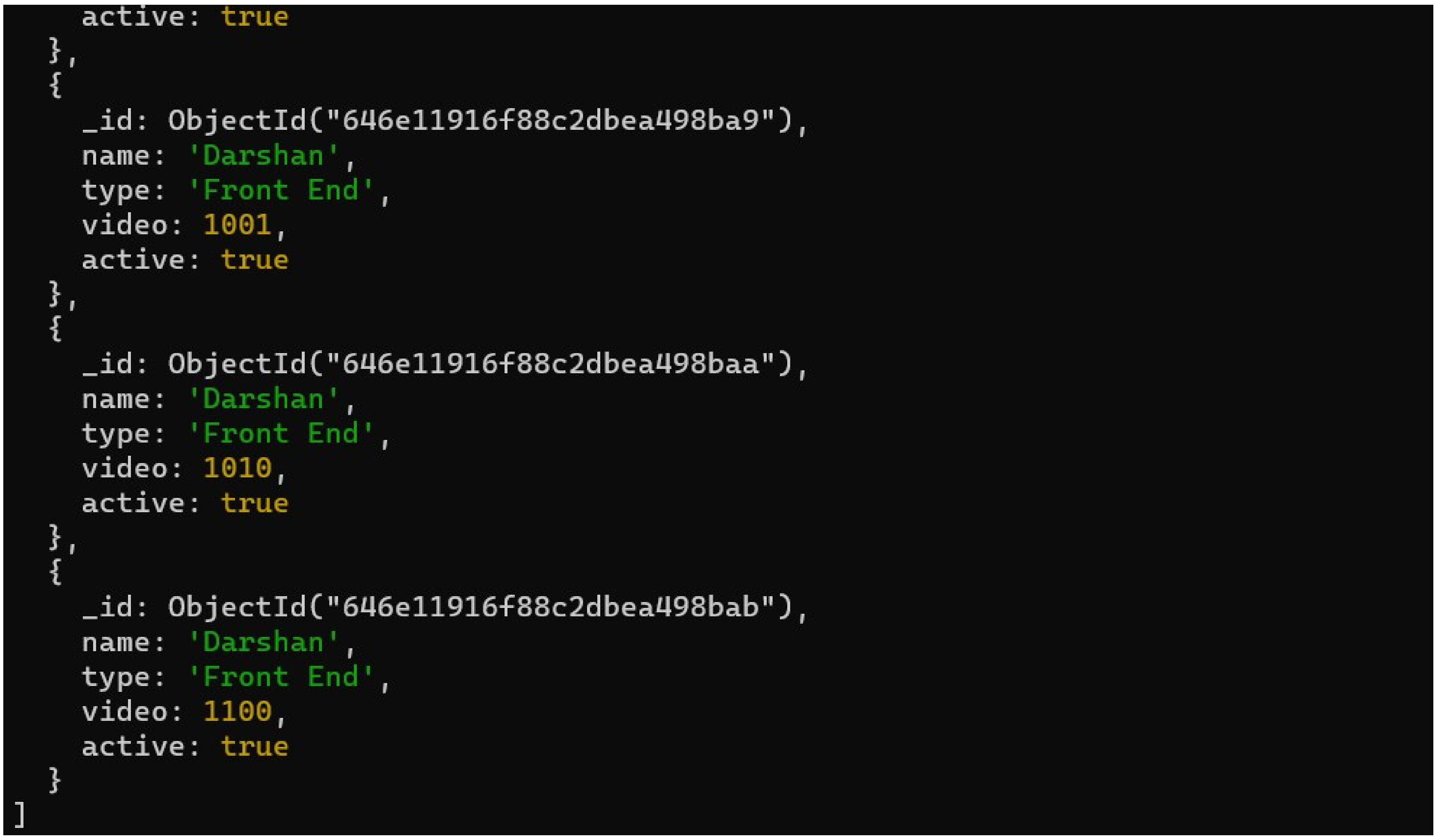


```
dp> db.dpp.updateOne({video:100}, {$set: {type:"Full Stack"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
dp> db.dpp.find()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Full Stack',
    video: 100,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Front End',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Front End',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498bab"),
    name: 'Darshan',
    type: 'Front End',
    video: 1100,
    active: true
  }
```

Example 2 With Output 😇

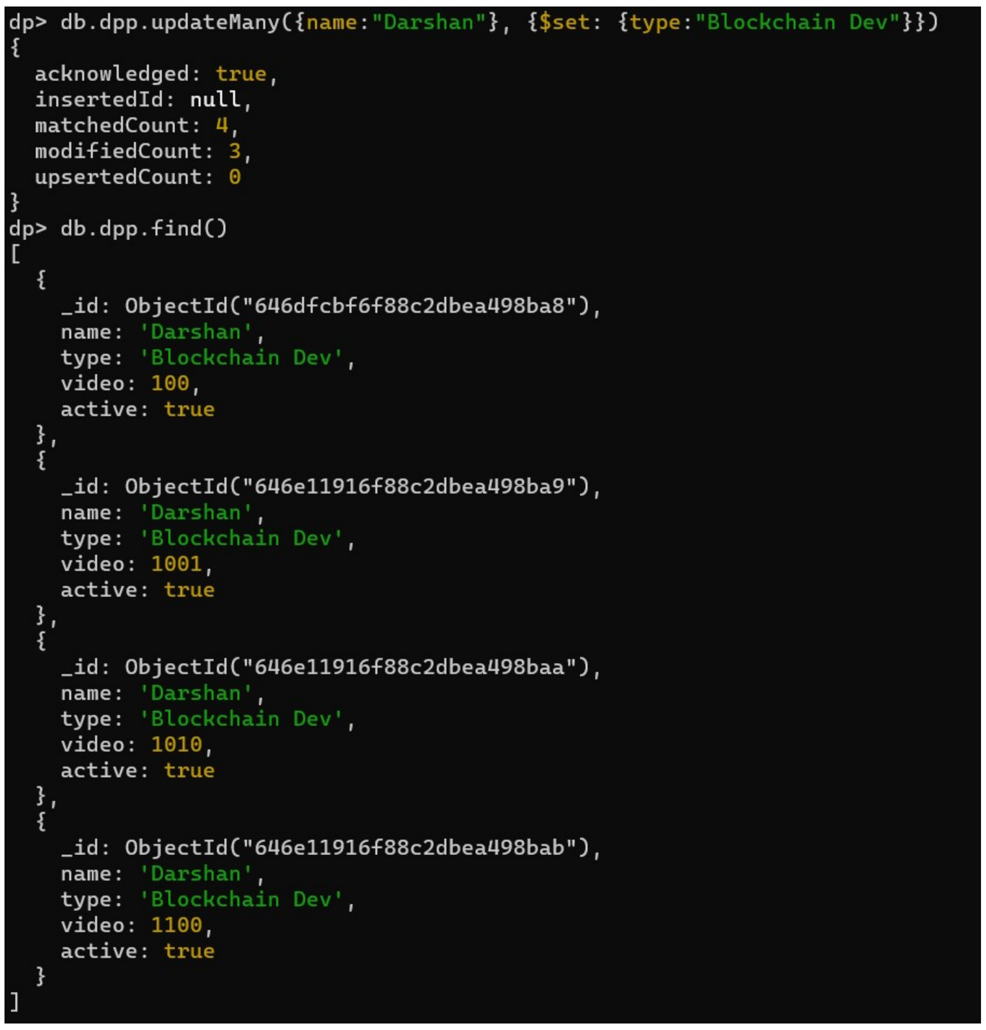


```
dp> db.dpp.updateOne({name:"Darshan"}, {$set: {type:"Blockchain Dev"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
dp> db.dpp.find()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 100,
```



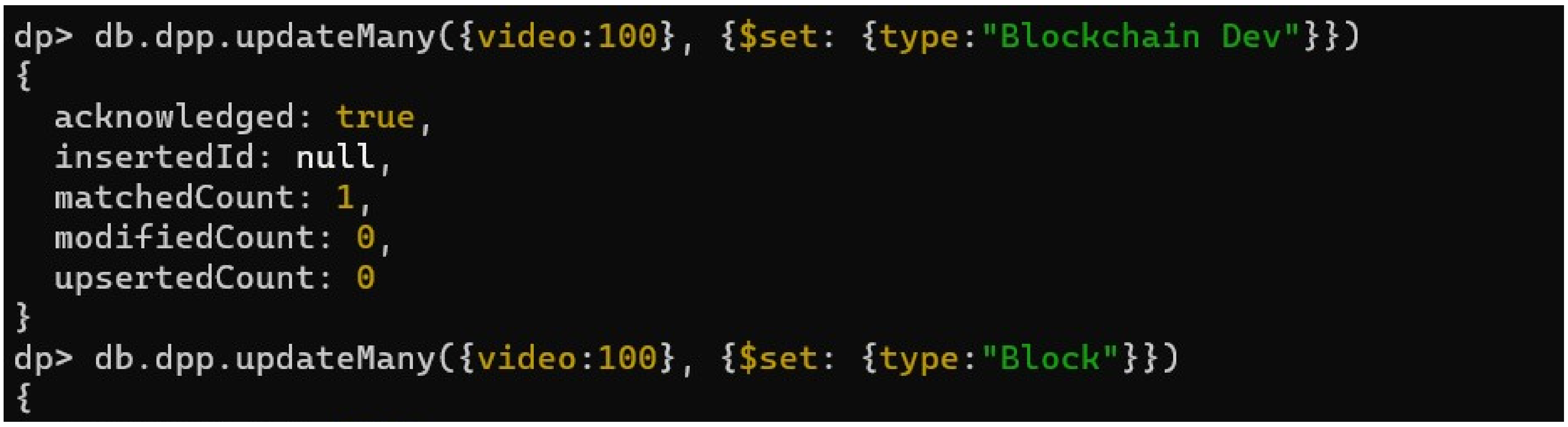
```
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Front End',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Front End',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498bab"),
    name: 'Darshan',
    type: 'Front End',
    video: 1100,
    active: true
  }
]
```

- **UpdateMany(😎)**

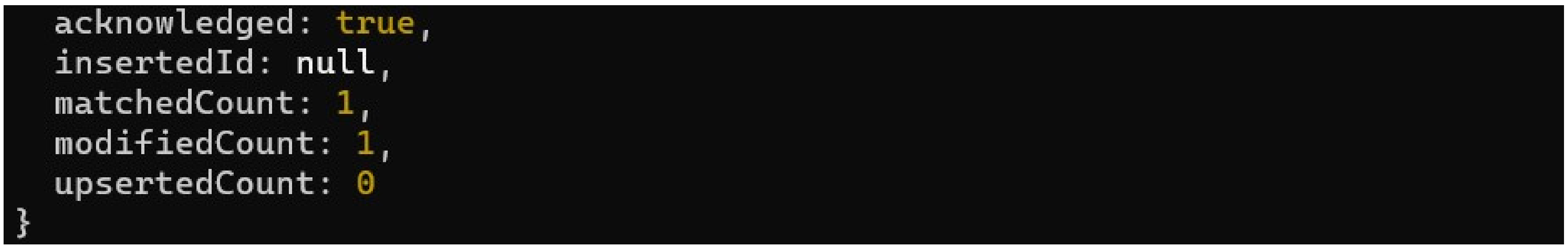


```
dp> db.dpp.updateMany({name:"Darshan"}, {$set: {type:"Blockchain Dev"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 4,
  modifiedCount: 3,
  upsertedCount: 0
}
dp> db.dpp.find()
[
  {
    _id: ObjectId("646dfcbf6f88c2dbea498ba8"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 100,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498bab"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1100,
    active: true
  }
]
```

Example: 2



```
dp> db.dpp.updateMany({video:100}, {$set: {type:"Blockchain Dev"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
dp> db.dpp.updateMany({video:100}, {$set: {type:"Block"}})
{
```

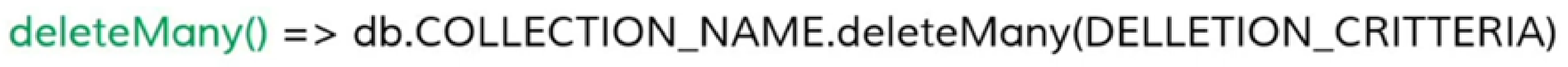


```
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

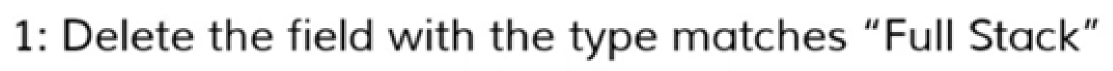
✏**Delete Operation:**

- Syntax: db.collectionName.remove(query)
- Example: db.users.remove({ name: "John" })
- This will remove the document(s) from the "users" collection where the name is "John".

- For Delete One Document Use **: deleteOne(😄)**



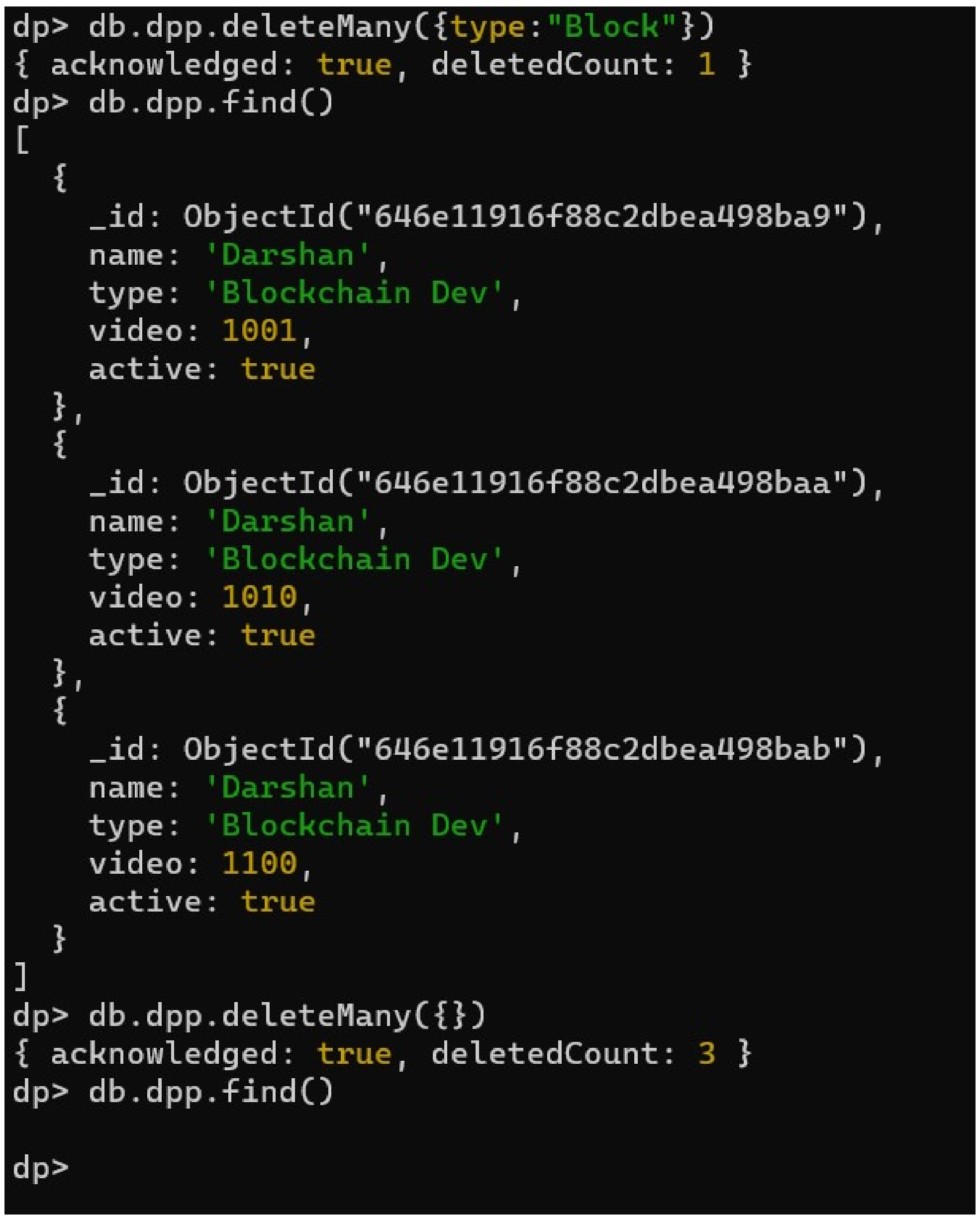
deleteMany() => db.COLLECTION_NAME.deleteMany(DELLETION_CRITTERIA)



1: Delete the field with the type matches "Full Stack"

We also have the remove() method to perform the delete operation but it's deprecated as per documents.

- **deleteMany(😄)**



```
dp> db.dpp.deleteMany({type:"Block"})
{ acknowledged: true, deletedCount: 1 }
dp> db.dpp.find()
[
  {
    _id: ObjectId("646e11916f88c2dbea498ba9"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1001,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498baa"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1010,
    active: true
  },
  {
    _id: ObjectId("646e11916f88c2dbea498bab"),
    name: 'Darshan',
    type: 'Blockchain Dev',
    video: 1100,
    active: true
  }
]
dp> db.dpp.deleteMany({})
{ acknowledged: true, deletedCount: 3 }
dp> db.dpp.find()

dp>
```