EJERCICIO 1

1. Partiendo del escritorio de tu ordenador, ejecuta los pasos necesarios para poder usar la consola de mongodb (abrir un terminal o consola, iniciar/parar servicios, ejecutar la shell de mongodb)

```
Microsoft Windows [Versión 10.0.19044.2130]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Andrea>mongosh
Current Mongosh Log ID: 6362b2fc22646fabae3aca32
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.6.0
Using MongoDB:          6.0.2
Using Mongosh:          1.6.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2022-10-26T19:23:34.116+02:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

------
   Free Monitoring URL:
   https://cloud.mongodb.com/freemonitoring/cluster/X6RSNWXNG6DBEZR7W6YE6CXFMLEJIXJO
------

test> show dbs
admin          72.00 KiB
concesionario  80.00 KiB
config         72.00 KiB
local          40.00 KiB
mongosh         8.00 KiB
test>
```

2. Crea una base de datos llamada concesionario. Recuerda insertar al menos un documento en una colección que se llame coches. Como propiedades del documento json, puedes usar matrícula, marca, modelo, versiones (sport, confort), kms y fecha de matriculación.

```
concesionario> db.coches.insertMany([{"matricula":"9153LDH","marca":"peugeot", "modelo":"308", "version":"sport","kms":15000,"fMatriculacion":new Date()},{"matricula":"1563
CPK","marca":"citroen","modelo":"xsara", "version":"confort", "kms":198000,"fMatriculacion": new Date()}, {"matricula":"7876MBB", "marca":"audi", "modelo":"a6", "version":"
sport","kms":3400,"fMatriculacion":new Date()}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("6359804108c8c30f9e1b27a9"),
    '1': ObjectId("6359804108c8c30f9e1b27aa"),
    '2': ObjectId("6359804108c8c30f9e1b27ab")
  }
}
```

3. Crea una colección que se llame clientes. Como propiedades del documento json, puedes usar nombre, f_nacimiento, e_mail y telefono.

```
concesionario> db.createCollection("clientes")
{ ok: 1 }
concesionario> db.clientes.insertOne({"nombre":"andrea", "fNacimiento":new Date(), "email":"andrea@correo.com", "telefono":"655789877"})
{
  acknowledged: true,
  insertedId: ObjectId("6359814608c8c30f9e1b27ac")
}
concesionario> db.clientes.insertOne({"nombre":"david", "fNacimiento":new Date(), "email":"david@correo.com", "telefono":"655218276"})
{
  acknowledged: true,
  insertedId: ObjectId("6359816408c8c30f9e1b27ad")
}
concesionario>
```

4. Visualiza el listado de todas las colecciones de datos disponibles y su contenido.

```
concesionario> show collections
clientes
coches
concesionario>
```

5. Operaciones básicas

```
concesionario> db.clientes.find().pretty()
[
  {
    _id: ObjectId("6359814608c8c30f9e1b27ac"),
    nombre: 'andrea',
    fNacimiento: ISODate("2022-10-26T18:49:42.438Z"),
    email: 'andrea@correo.com',
    telefono: '655789877'
  },
  {
    _id: ObjectId("6359816408c8c30f9e1b27ad"),
    nombre: 'david',
    fNacimiento: ISODate("2022-10-26T18:50:12.500Z"),
    email: 'david@correo.com',
    telefono: '655218276'
  }
]
concesionario>
```

```
concesionario> db.coches.find().pretty()
[
  {
    _id: ObjectId("6359804108c8c30f9e1b27a9"),
    matricula: '9153LDH',
    marca: 'peugeot',
    modelo: '308',
    version: 'sport',
    kms: 15000,
    fMatriculacion: ISODate("2022-10-26T18:45:21.388Z")
  },
  {
    _id: ObjectId("6359804108c8c30f9e1b27aa"),
    matricula: '1563CPK',
    marca: 'citroen',
    modelo: 'xsara',
    version: 'confort',
    kms: 198000,
    fMatriculacion: ISODate("2022-10-26T18:45:21.388Z")
  },
  {
    _id: ObjectId("6359804108c8c30f9e1b27ab"),
    matricula: '7876MBB',
    marca: 'audi',
    modelo: 'a6',
    version: 'sport',
    kms: 3400,
    fMatriculacion: ISODate("2022-10-26T18:45:21.388Z")
  }
]
concesionario>
```

Filtrado por versión

```
concesionario> db.coches.find({"version":"sport"},{"marca":1, "modelo":1, _id:0})
[
  { marca: 'peugeot', modelo: '308' },
  { marca: 'audi', modelo: 'a6' }
]
concesionario>
```

Limit, skip y sort
(el skip me quita los primeros registros, limit solo me muestra esa cantidad de registros, sort ordena con 1 y -1)

Operaciones de comparación
$gt o $gte
$lt o $lte

```
concesionario> db.coches.find({"kms":{$lt:4000}},{"matricula":1, "fMatriculacion":1, "kms":1, _id:0})
[
  {
    matricula: '7876MBB',
    kms: 3400,
    fMatriculacion: ISODate("2022-10-26T18:45:21.388Z")
  }
]
concesionario>
```

EJERCICIO 2

1. Crea la base de datos Retail.

```
test> use retail
switched to db retail
```

2. Inserta en la colección productos con los siguientes documentos de uno en uno:

```
retail> db.createCollection("productos")
{ ok: 1 }
```

```
retail> db.productos.insertOne({"referencia":"P0001", "tipo":"camisa","paraHombre":true,"talla": "XS", "precio":20.99});
{
  acknowledged: true,
  insertedId: ObjectId("6362b9c5db7dc310d04b3dd2")
}
retail> db.productos.insertOne({"referencia":"P0002", "tipo":"camisa","paraHombre":true,"talla": "XL", "precio":30.25});
{
  acknowledged: true,
  insertedId: ObjectId("6362b9f0db7dc310d04b3dd3")
}
retail> db.productos.insertOne({"referencia":"P0003", "tipo":"pantalon","paraMujer":true,"talla": "L", "precio":20.99});
{
  acknowledged: true,
  insertedId: ObjectId("6362ba27db7dc310d04b3dd4")
}
retail>
```

3. Elimínalos todos.

```
retail> db.productos.drop()
true
retail>
```

4. Ahora créalos, pero todos a la vez.

```
retail> db.productos.insertMany([{"referencia":"P0001", "tipo":"camisa","paraHombre":true,"talla": "XS", "precio":20.99},{"referencia":"P0002", "tipo":"camisa","paraHombre"
:true,"talla": "XL", "precio":30.25},{"referencia":"P0003", "tipo":"pantalon","paraMujer":true,"talla": "L", "precio":20.99}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("6362bf2e46a8eb94c17dd9fb"),
    '1': ObjectId("6362bf2e46a8eb94c17dd9fc"),
    '2': ObjectId("6362bf2e46a8eb94c17dd9fd")
  }
}
retail>
```

5. Actualiza todos los documentos con un precio inferior a 25 para que tenga un precio un 10% más caro.

```
retail> db.productos.update({"precio":{$lte:25}},{$mul:{precio:1.10}},{multi:true})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

```
retail> db.productos.find().pretty()
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    tipo: 'camisa',
    paraHombre: true,
    talla: 'XS',
    precio: 23.089
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fc"),
    referencia: 'P0002',
    tipo: 'camisa',
    paraHombre: true,
    talla: 'XL',
    precio: 30.25
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fd"),
    referencia: 'P0003',
    tipo: 'pantalon',
    paraMujer: true,
    talla: 'L',
    precio: 23.089
  }
]
retail>
```

6. Reemplaza cada documento que sea para hombre con la misma estructura pero añadiendo una propiedad nueva: “paraMujer”: false.

```
retail> db.productos.updateMany({"paraHombre":true},{$set:{"paraMujer":false}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
retail>
```

```
retail> db.productos.find().pretty()
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    tipo: 'camisa',
    paraHombre: true,
    talla: 'XS',
    precio: 23.089,
    paraMujer: false
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fc"),
    referencia: 'P0002',
    tipo: 'camisa',
    paraHombre: true,
    talla: 'XL',
    precio: 30.25,
    paraMujer: false
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fd"),
    referencia: 'P0003',
    tipo: 'pantalon',
    paraMujer: true,
    talla: 'L',
    precio: 23.089
  }
]
retail>
```

7. Reemplaza cada documento que sea para mujer con la misma estructura pero añadiendo una propiedad nueva: “paraHombre”: false.

```
retail> db.productos.updateMany({"paraMujer":true},{$set:{"paraHombre":false}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
retail>
```

```
retail> db.productos.find().pretty()
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    tipo: 'camisa',
    paraHombre: true,
    talla: 'XS',
    precio: 23.089,
    paraMujer: false
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fc"),
    referencia: 'P0002',
    tipo: 'camisa',
    paraHombre: true,
    talla: 'XL',
    precio: 30.25,
    paraMujer: false
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fd"),
    referencia: 'P0003',
    tipo: 'pantalon',
    paraMujer: true,
    talla: 'L',
    precio: 23.089,
    paraHombre: false
  }
]
retail>
```

8. Actualiza cada documento, filtrando por sus referencia. Las propiedades cambiando el tipo camisa por chaqueta y la talla debe ser igual a M.

```
retail> db.productos.updateMany({"tipo":"camisa"},{$set:{"tipo":"chaqueta"}, $set:{"talla":"M"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
retail>
```

```
retail> db.productos.find().pretty()
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    tipo: 'chaqueta',
    paraHombre: true,
    talla: 'M',
    precio: 23.089,
    paraMujer: false
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fc"),
    referencia: 'P0002',
    tipo: 'chaqueta',
    paraHombre: true,
    talla: 'M',
    precio: 30.25,
    paraMujer: false
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fd"),
    referencia: 'P0003',
    tipo: 'pantalon',
    paraMujer: true,
    talla: 'L',
    precio: 23.089,
    paraHombre: false
  }
]
retail>
```

EJERCICIO 3

1. **Actualiza la colección productos con las siguientes propiedades:**

● P0001, añade las siguientes propiedades: tienda: [“Jerez”, “Sevilla”, “Cordoba”] proveedor: {nombre: “Camiseros SA”, nif:”B12345678”, contacto:”Jose”}

```
retail> db.productos.update({"referencia":"P0001"},{$set:{"tienda":["Jerez", "Sevilla", "Cordoba"], "proveedor":{nombre:"Camiseros SA", nif:"B123
45678", contacto:"Jose"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
retail>
```

● P0002, añade las siguientes propiedades: tienda: [“Jerez”, “Chucena”, “Cordoba”] proveedor: {nombre: “Camiseros SA”, nif:”B12345678”, contacto:”Juan”}

```
retail> db.productos.update({"referencia":"P0002"},{$set:{"tienda":["Jerez", "Chucena", "Cordoba"], "proveedor":{nombre:"Camiseros SA", nif:"B123
45678", contacto:"Juan"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

● P0003, añade las siguientes propiedades: tienda: [“Rota”, “Sevilla”] proveedor: {nombre: “Pantaloneros SA”, nif:”B87654321”, contacto:”Jose”}

```
retail> db.productos.update({"referencia":"P0001"},{$set:{"tienda":["Jerez", "Sevilla", "Cordoba"], "proveedor":{nombre:"Camiseros SA", nif:"B123
45678", contacto:"Jose"}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
retail>
```

2. **Realiza las siguientes consultas sobre la colección productos:**

● De cada producto mostrar su referencia, las tiendas en las que se vende y el nombre del proveedor que lo suministra.

```
retail> db.productos.find({},{"referencia":1, "tiendas":1,"proveedor.nombre":1})
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    proveedor: { nombre: 'Camiseros SA' }
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fc"),
    referencia: 'P0002',
    proveedor: { nombre: 'Camiseros SA' }
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fd"),
    referencia: 'P0003',
    proveedor: { nombre: 'Pantaloneros SA' }
  }
]
retail>
```

● Mostrar la referencia y talla de todos los productos que se venden en la tienda de Jerez.

```
db.productos.find({"tienda":"Jerez"},{"referencia":1,"talla":1, "_id":0})
```

```
retail> db.productos.find({"tienda":"Jerez"},{"referencia":1,"talla":1, "_id":0})
[
  { referencia: 'P0001', talla: 'M' },
  { referencia: 'P0002', talla: 'M' }
]
retail>
```

● Mostrar los productos que se venden en las tiendas de Jerez y Córdoba.
db.productos.find({ $and:[{"tienda":"Jerez"},{"tienda":"Cordoba"}]})

```
retail> db.productos.find({ $and:[{"tienda":"Jerez"},{"tienda":"Cordoba"}]})
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    tipo: 'chaqueta',
    paraHombre: true,
    talla: 'M',
    precio: 23.089,
    paraMujer: false,
    proveedor: { nombre: 'Camiseros SA', nif: 'B12345678', contacto: 'Jose' },
    tienda: [ 'Jerez', 'Sevilla', 'Cordoba' ]
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fc"),
    referencia: 'P0002',
    tipo: 'chaqueta',
    paraHombre: true,
    talla: 'M',
    precio: 30.25,
    paraMujer: false,
    proveedor: { nombre: 'Camiseros SA', nif: 'B12345678', contacto: 'Juan' },
    tienda: [ 'Jerez', 'Chucena', 'Cordoba' ]
  }
]
retail>
```

● Mostrar la referencia, talla y precio de los productos suministrados por el proveedor cuyo nif es ”B12345678”.
db.productos.find({"proveedor.nif”:”B12345678”},{"referencia":1,"talla":1, “precio”:1, "_id":0})

```
retail> db.productos.find({"proveedor.nif":"B12345678"},{"referencia":1,"talla":1, "precio":1, "_id":0})
[
  { referencia: 'P0001', talla: 'M', precio: 23.089 },
  { referencia: 'P0002', talla: 'M', precio: 30.25 }
]
retail>
```

● Mostrar los productos cuyo contacto del proveedor sea “Jose”.
db.productos.find({“proveedor.contacto”:”Jose”})

```
retail> db.productos.find({"proveedor.contacto":"Jose"})
[
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fb"),
    referencia: 'P0001',
    tipo: 'chaqueta',
    paraHombre: true,
    talla: 'M',
    precio: 23.089,
    paraMujer: false,
    proveedor: { nombre: 'Camiseros SA', nif: 'B12345678', contacto: 'Jose' },
    tienda: [ 'Jerez', 'Sevilla', 'Cordoba' ]
  },
  {
    _id: ObjectId("6362bf2e46a8eb94c17dd9fd"),
    referencia: 'P0003',
    tipo: 'pantalon',
    paraMujer: true,
    talla: 'L',
    precio: 23.089,
    paraHombre: false,
    proveedor: { nombre: 'Pantaloneros SA', nif: 'B87654321', contacto: 'Jose' },
    tienda: [ 'Rota', 'Sevilla' ]
  }
]
retail>
```

EJERCICIO 4

1. **Consultas con Profesores**
   - Obtener todos los profesores

```
universidad> db.profesores.find()
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51
  },
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54
  }
]
universidad>
```

● Obtener todos los profesores asociados

```
universidad> db.profesores.find({"esAsociado":true})
[
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  }
]
universidad>
```

● Obtener todos los profesores titulares y mayores de 50

```
universidad> db.profesores.find( { $and: [{"edad":{$gte:50}},{"esTitular":true}]})
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54
  }
]
universidad>
```

● Obtener todos los profesores con edades entre 50 y 60

```
universidad> db.profesores.find( { $and: [{"edad":{$gte:50}},{"edad":{$lte:60}}]})
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54
  }
]
universidad>
```

● Número total de profesores

```
universidad> db.profesores.find().count()
3
universidad>
```

● Número total de profesores asociados

```
universidad> db.profesores.find({"esAsociado":true}).count()
1
universidad>
```

● Obtener solo los dos primeros resultados

```
universidad> db.profesores.find().limit(2)
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51
  },
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  }
]
universidad>
```

● Obtener todos los profesores con especialidad física o biología

```
universidad> db.profesores.find({$or:[{"especialidad":"física"},{"especialidad":"biología"}]})
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51
  },
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54
  }
]
universidad>
```

● Obtener todos los profesores con especialidad biología o que sea asociado

```
universidad> db.profesores.find({$or:[{"esAsociado":true},{"especialidad":"biología"}]})
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51
  },
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  }
]
universidad>
```

● Obtener todos los profesores que imparten exáctamente matemáticas y física (exáctamente en ese orden)

```
universidad> db.profesores.find({$and:[{"especialidad":"matemáticas"},{"especialidad":"física"}]})
[
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  }
]
universidad>
```

● Obtener todos los profesores que imparten matemáticas y física (sin orden)

```
universidad> db.profesores.find({$and:[{"especialidad":"física"},{"especialidad":"matemáticas"}]})
[
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39
  }
]
```

● Número total de profesores que imparten biología.

```
universidad> db.profesores.find({"especialidad":"biología"}).count()
1
universidad>
```

2. Consultas con Profesores y Asignaturas II

● Actualiza la colección Profesores con los datos de las asignaturas que imparten.

```
universidad> db.profesores.updateOne({"nombre":"María"}, {$set:{"asignatura":{"id":"A0001", "nombre":"Biología molecular", "creditos":6}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
universidad> db.profesores.updateOne({"nombre":"José Luis"}, {$set:{"asignatura":{"id":"A0002", "nombre":"Termodinámica", "creditos":9}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
universidad> db.profesores.updateOne({"nombre":"Antonio"}, {$set:{"asignatura":{"id":"A0002", "nombre":"Termodinámica", "creditos":9}}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

● Obtener todos los profesores que imparten la asignatura A0002.

```
universidad> db.profesores.find({"asignatura.id":"A0002"})
[
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  }
]
universidad>
```

● Obtener todos los profesores que imparten termodinámica.

```
universidad> db.profesores.find({"asignatura.nombre":"Termodinámica"})
[
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  }
]
universidad>
```

● Obtener todos los profesores que den asignaturas de más de 6 créditos.

```
universidad> db.profesores.find({"asignatura.creditos":{$gt:6}})
[
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  },
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  }
]
universidad>
```

● Obtener todos los que den asignaturas de más de 6 créditos y mayores de 40.

```
universidad> db.profesores.find({$and:[{"asignatura.creditos":{$gt:6}},{"edad":{$gt:40}}]})
[
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 }
  }
]
universidad>
```

3. Consultas con Profesores y Vehículos

● Añadir propiedades (vehículo) y registros a la colección Profesores

● Obtener todos los profesores que tienen un SUV.

```
universidad> db.profesores.find({"vehiculo.tipo":"SUV"})
[
  {
    _id: ObjectId("63637e7114c7cdca4b9a7782"),
    nombre: 'María',
    apellidos: 'Suárez Manrique',
    especialidad: [ 'biología' ],
    esTitular: true,
    esAsociado: false,
    edad: 51,
    asignatura: { id: 'A0001', nombre: 'Biología molecular', creditos: 6 },
    vehiculo: {
      matricula: '1111ABC',
      tipo: 'SUV',
      marca: 'Lexus',
      caballos: 190,
      uso: 3
    }
  },
  {
    _id: ObjectId("63637f0f14c7cdca4b9a7783"),
    nombre: 'José Luis',
    apellidos: 'López Pérez',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: false,
    esAsociado: true,
    edad: 39,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 },
    vehiculo: {
      matricula: '2222DEF',
      tipo: 'SUV',
universidad>
      caballos: 120,
      uso: 13
    }
  }
]
universidad>
```

● Obtener todos los profesores que tienen una moto y un uso de menos de 5 años.

```
universidad> db.profesores.find({$and:[{"vehiculo.tipo":"moto"},{"vehiculo.uso":{$lt:5}}]})
[
  {
    _id: ObjectId("6363969814c7cdca4b9a7786"),
    nombre: 'Elena',
    apellidos: 'Hérnandez Serafín',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: true,
    esAsociado: false,
    vehiculo: {
      matricula: '5555NOP',
      tipo: 'moto',
      marca: 'Honda',
      caballos: 170,
      uso: 4
    }
  }
]
universidad>
```

● Obtener todos los profesores con vehículo de más de 150 caballos y que impartan física.

```
universidad> db.profesores.find({$and:[{"especialidad":"física"},{"vehiculo.caballos":{$gt:150}}]})
[
  {
    _id: ObjectId("63637fc114c7cdca4b9a7784"),
    nombre: 'Antonio',
    apellidos: 'Munguía Arteche',
    especialidad: [ 'física', 'química' ],
    esTitular: true,
    esAsociado: false,
    edad: 54,
    asignatura: { id: 'A0002', nombre: 'Termodinámica', creditos: 9 },
    vehiculo: {
      matricula: '3333HIJ',
      tipo: 'Turismo',
      marca: 'Volvo',
      caballos: 176,
      uso: 6
    }
  },
  {
    _id: ObjectId("6363969814c7cdca4b9a7786"),
    nombre: 'Elena',
    apellidos: 'Hérnandez Serafín',
    especialidad: [ 'matemáticas', 'física' ],
    esTitular: true,
    esAsociado: false,
    vehiculo: {
      matricula: '5555NOP',
      tipo: 'moto',
      marca: 'Honda',
      caballos: 170,
      uso: 4
    }
  }
]
universidad>
```

● Obtener el nº de profesores con más de 50 años, que tienen moto.

```
universidad> db.profesores.find({$and:[{"vehiculo.tipo":"moto"},{"edad":{$gt:50}}]}).count()
0
universidad>
```

(Los nuevos insert no incluían edad)

● Obtener el nº de profesores titulares que tienen un SUV.

```
universidad> db.profesores.find({$and:[{"vehiculo.tipo":"SUV"},{"esTitular":true}]}).count()
1
universidad>
```