

CONFIGURACIÓN DEL ENTORNO MongoDB

Preparación, configuración e inicialización de Mongo DB en el puerto C:\data\db

```
Símbolo del sistema - Mongo
Microsoft Windows [Versión 10.0.26100.3775]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\bryan>mongod.exe --version
db version v8.0.8
Build Info: {
  "version": "8.0.8",
  "gitVersion": "7f52660c14217ed2c8d3240f823a2291a4fe6abd",
  "modules": [],
  "allocator": "tcmalloc-gperf",
  "environment": {
    "distmod": "windows",
    "distarch": "x86_64",
    "target_arch": "x86_64"
  }
}

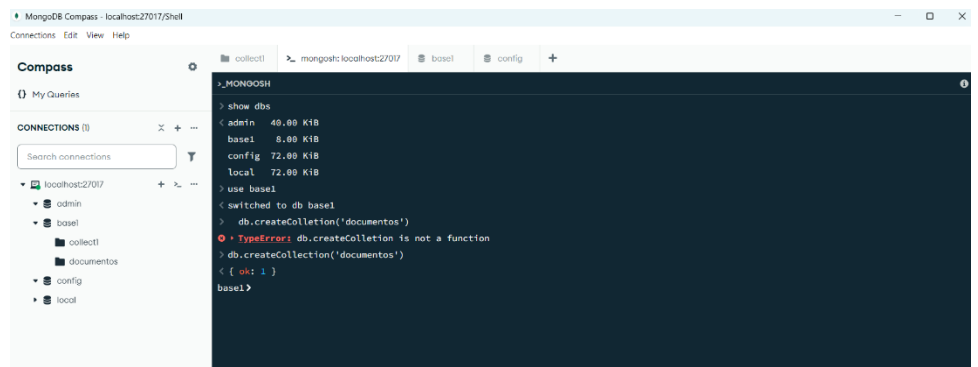
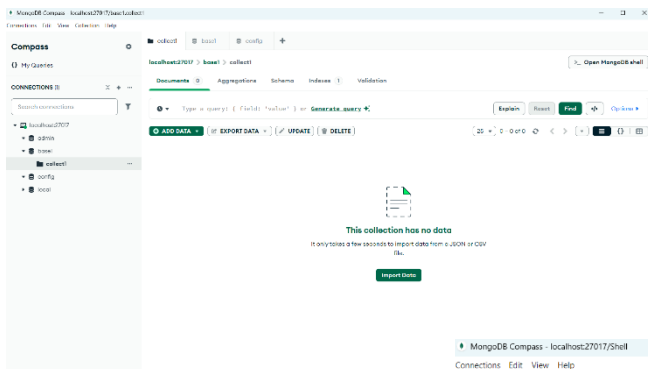
C:\Users\bryan>mkdir C:\data\db
Ya existe el subdirectorio o el archivo C:\data\db.

C:\Users\bryan>cd "C:\Program Files\MongoDB\Server\8.0\bin"

C:\Program Files\MongoDB\Server\8.0\bin>Mongo
{"t":{"$date":"2025-04-29T23:26:28.406-06:00"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"thread1", "msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2025-04-29T23:26:28.406-06:00"},"s":"I", "c":"CONTROL", "id":5945603, "ctx":"thread1", "msg":"Multi threading initialized"}
{"t":{"$date":"2025-04-29T23:26:28.406-06:00"},"s":"I", "c":"NETWORK", "id":4648601, "ctx":"thread1", "msg":"Implicit TCP FastOpen unavailable. If TCP FastOpen is required, set at least one of the related parameters, 'attr':{'relatedParameters':{'tcpFastOpenServer','tcpFastOpenClient','tcpFastOpenQueueSize'}}}"
```

DISEÑO DE LA BASE DE DATOS

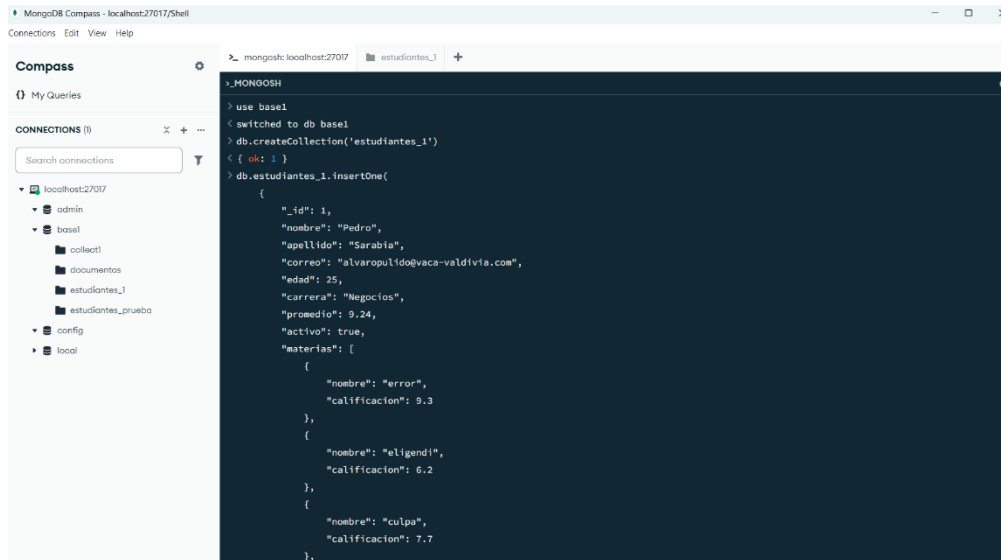
Creación de una nueva conexión en MongoDB compass y creación de una base de datos prueba, llamada documentos y otra de estudiantes que contiene un **registro de estudiantes** (id, nombre, apellido, carrera, promedio y las materias que cursa), utilizando los comandos de Mongosh



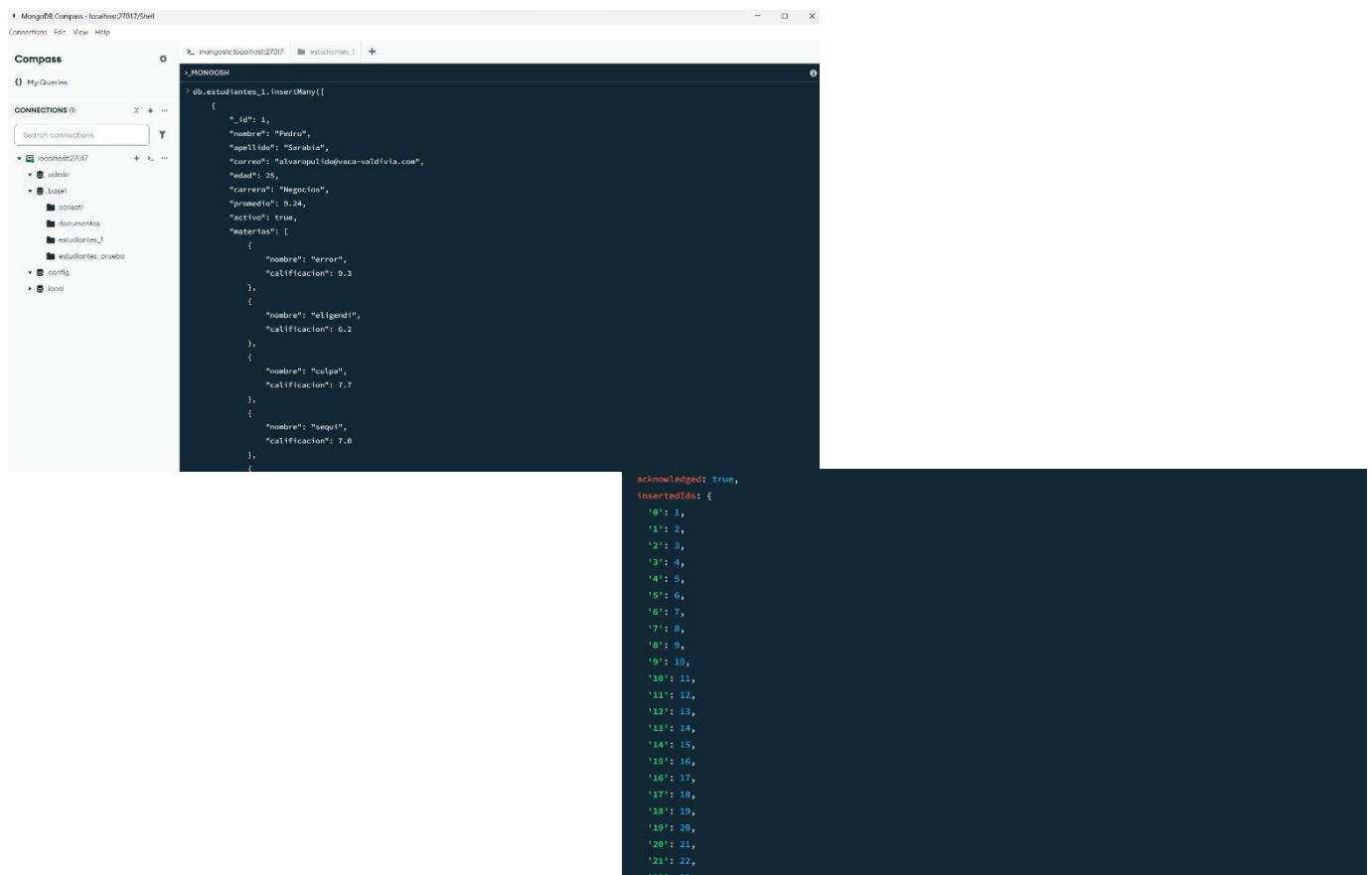
OPERACIONES CRUD

CREATE

Aquí se usa el comando para crear una collection dentro de la base de datos seleccionada y para insertar valores a la misma collection usamos comando `db.collection_name.insertOne()`

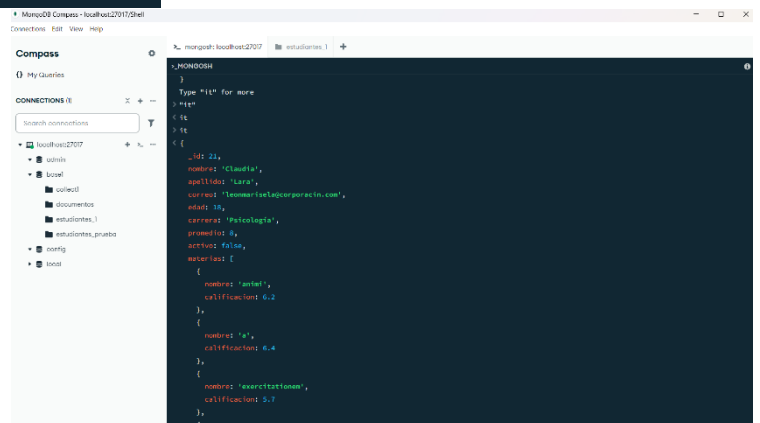
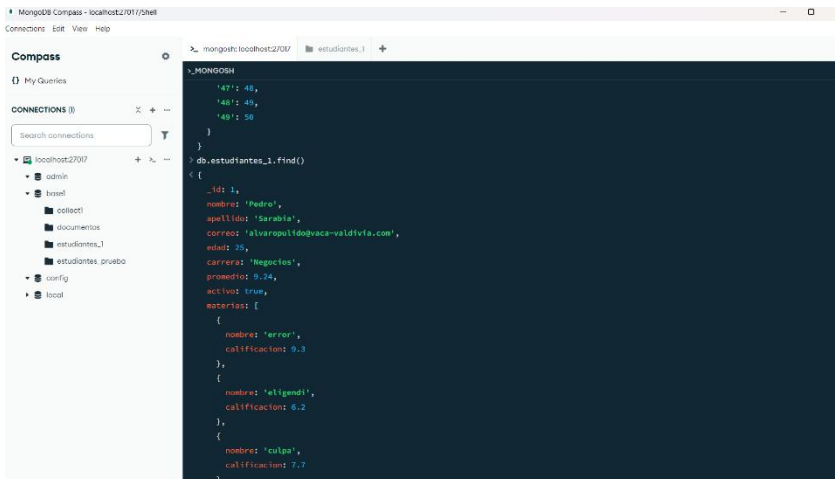


Para agregar diferentes valores en la collection creada dentro de la base de datos utilizada, se usa el comando `db.collection_name.insertMany()`

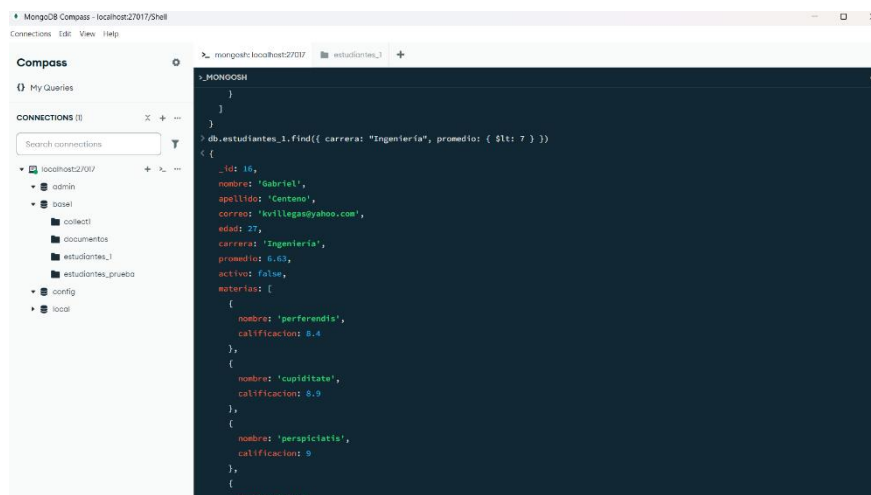


READ

Usar el comando `db.estudiantes_1.find()` para ver todos los estudiantes, no obstante, mongosh te va mostrando 20 id a la vez, y se debe escribir “it” para seguir leyendo los siguientes 20 datos.

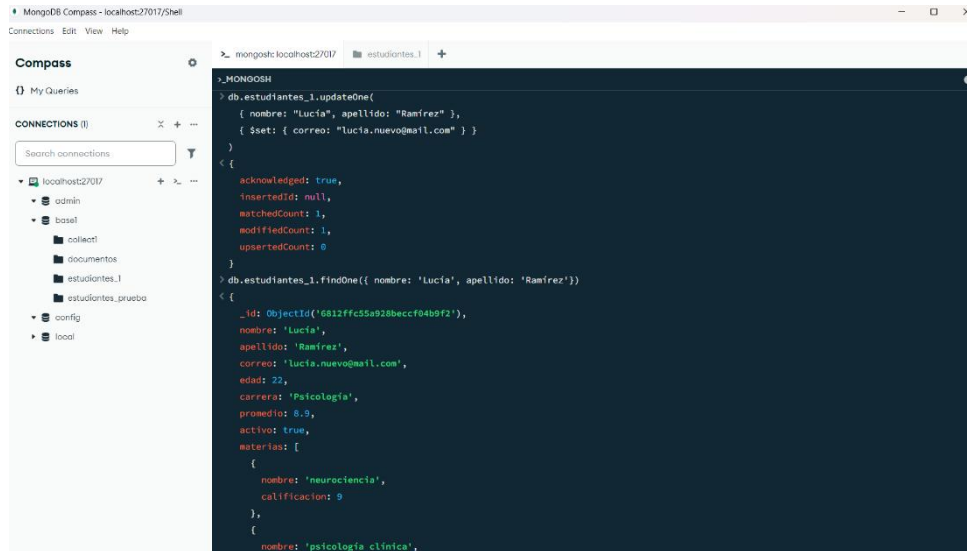


Existen algunos operadores utilizados en comandos mongosh para leer datos mayores, menores o iguales a ciertos parámetros, por ejemplo, usando el comando `db.estudiantes_1.find({ carrera: "Ingeniería", promedio: { $lt: 7 } })`, se mostrarán los estudiantes que tengan un promedio menor a 7 en la carrera de ingeniería.



UPDATE

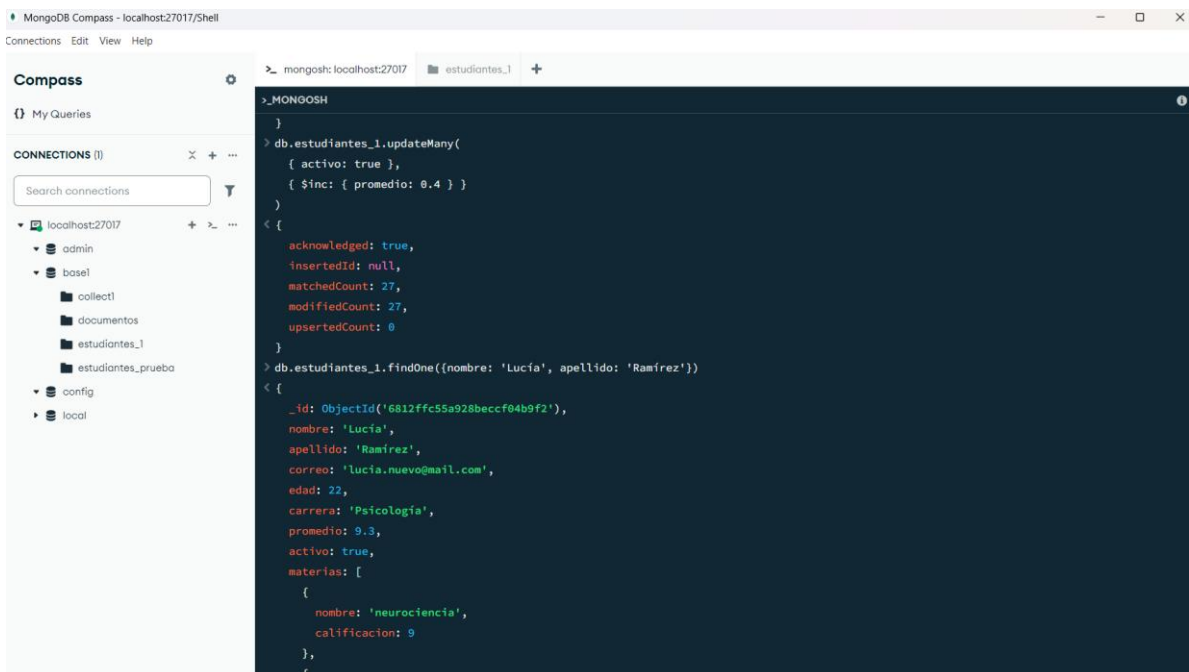
El comando `.updateOne()` y el operador `$set` funcionan para poder cambiar y actualizar un valor o un dato dentro de la collection que estamos utilizando.



The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' panel shows a connection to 'localhost:27017' with a database named 'estudiantes_1'. The main panel displays the following command and its result:

```
>_MONGOOSH
> db.estudiantes_1.updateOne(
  { nombre: "Lucia", apellido: "Ramirez" },
  { $set: { correo: "lucia.nuevo@mail.com" } }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.estudiantes_1.findOne({ nombre: 'Lucia', apellido: 'Ramirez'})
< {
  _id: ObjectId('6812ffc55a928beccf04b9f2'),
  nombre: 'Lucia',
  apellido: 'Ramirez',
  correo: 'lucia.nuevo@mail.com',
  edad: 22,
  carrera: 'Psicología',
  promedio: 8.9,
  activo: true,
  materias: [
    {
      nombre: 'neurociencia',
      calificacion: 9
    },
    {
      nombre: 'psicología clínica',
```

En caso de querer hacer una modificación general, en este caso, subir 0.4 décimas en el promedio a todos los estudiantes activos, se puede utilizar el comando `.updateMany()` acompañado del operador `$inc`

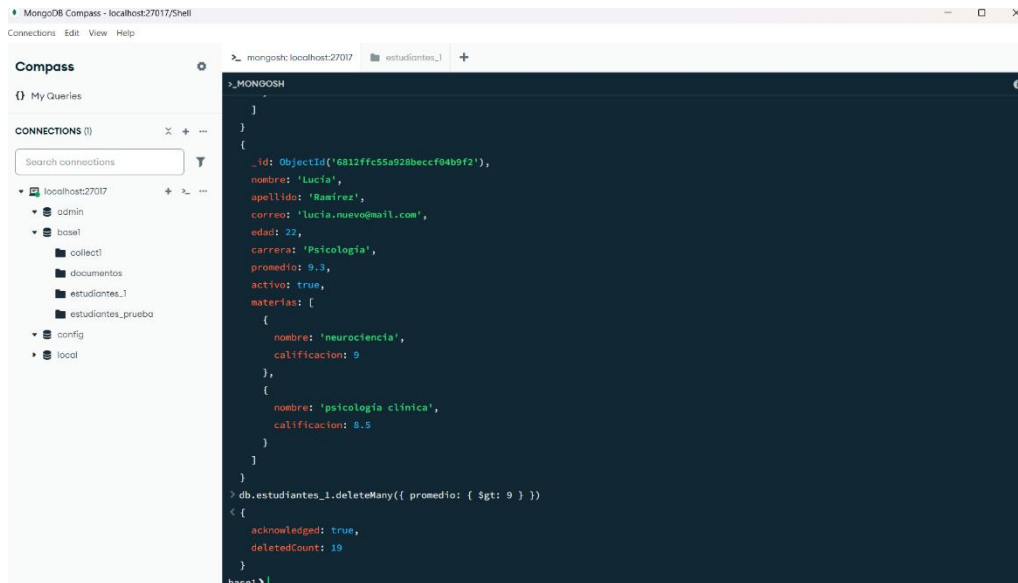


The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' panel shows a connection to 'localhost:27017' with a database named 'estudiantes_1'. The main panel displays the following command and its result:

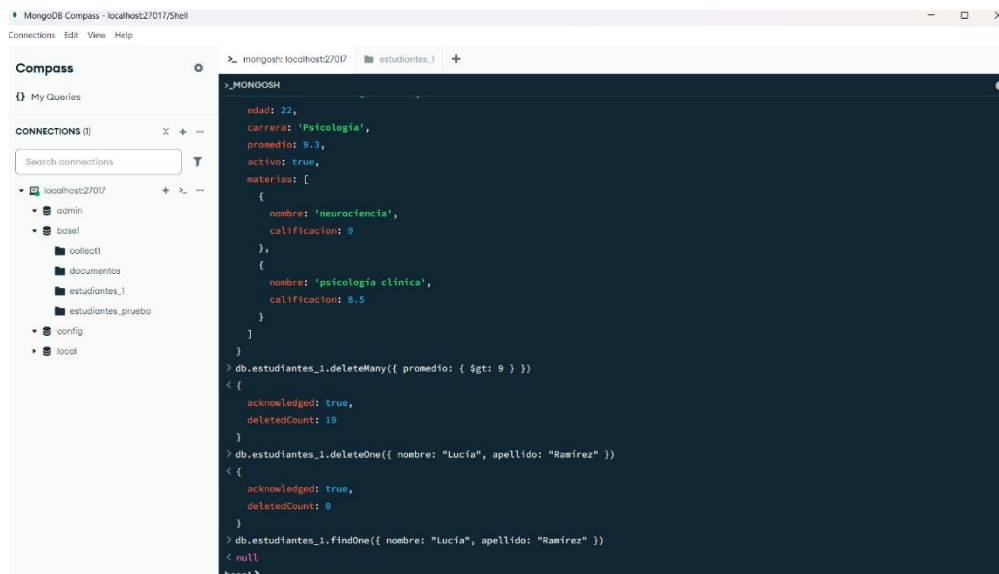
```
>_MONGOOSH
> db.estudiantes_1.updateMany(
  { activo: true },
  { $inc: { promedio: 0.4 } }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 27,
  modifiedCount: 27,
  upsertedCount: 0
}
> db.estudiantes_1.findOne({nombre: 'Lucia', apellido: 'Ramirez'})
< {
  _id: ObjectId('6812ffc55a928beccf04b9f2'),
  nombre: 'Lucia',
  apellido: 'Ramirez',
  correo: 'lucia.nuevo@mail.com',
  edad: 22,
  carrera: 'Psicología',
  promedio: 9.3,
  activo: true,
  materias: [
    {
      nombre: 'neurociencia',
      calificacion: 9
    },
    {
```

DELETE

Se puede utilizar el comando `.deleteMany()` acompañado de algún operador para eliminar cierto valor específico dentro de la collection. Por ejemplo, en el ejemplo se va a eliminar a todos los estudiantes que tengan un promedio mayor a 9, es decir, que están exentos del examen final.



Otra transformación a la base de datos es eliminar un dato previamente cargado, utilizando el comando `.deleteOne()`, en este caso se va a eliminar por nombre el alumno que se agregó al final, “Lucía Ramírez”.

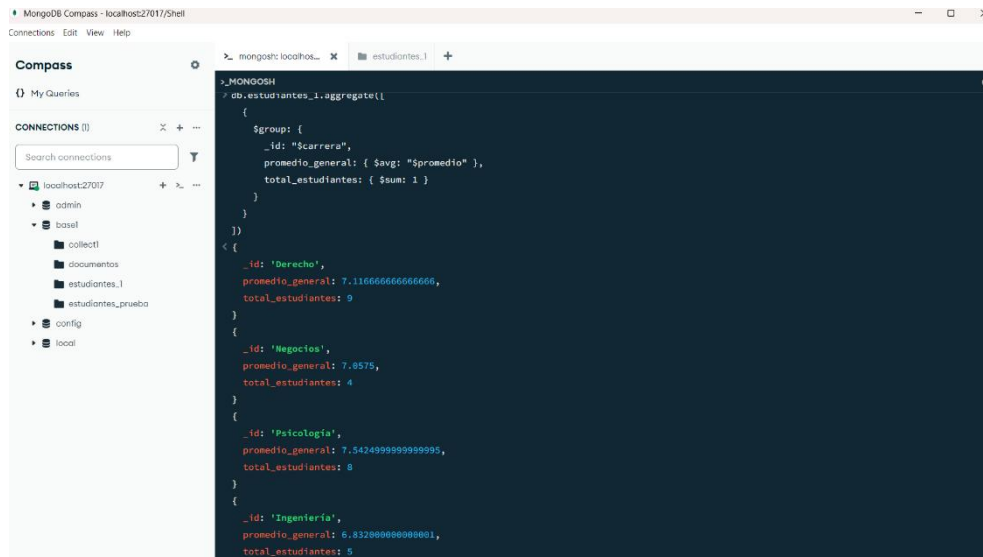


CONSULTAS AVANZADAS

AGRUPAR Y ORDENAR (pipeline)

A través del comando `.aggregate()` se pueden realizar consultas que incluyan agrupaciones, filtrados, conteos, ordenar, entre otras consultas.

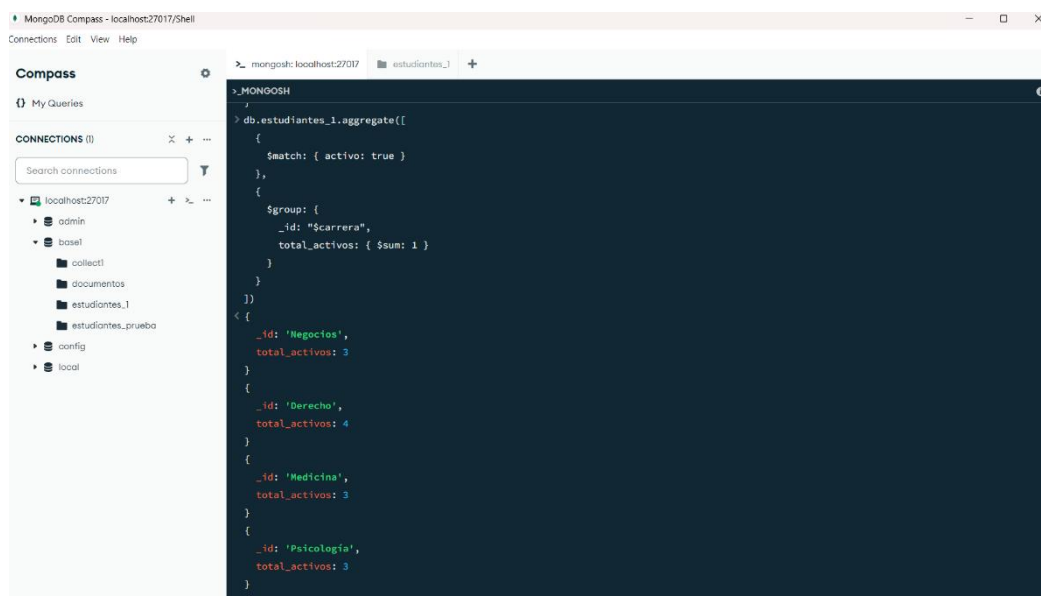
- Se utiliza `$group` que permite agrupar documentos que tienen valores comunes en un campo específico, en este caso es el promedio de estudiantes por carrera.



```
> use mongodb
> use estudiantes_1
> db.estudiantes_1.aggregate([
  {
    $group: {
      _id: "$carrera",
      promedio_general: { $avg: "$promedio" },
      total_estudiantes: { $sum: 1 }
    }
  }
])
< [
  {
    "_id": "Derecho",
    "promedio_general": 7.116666666666666,
    "total_estudiantes": 9
  },
  {
    "_id": "Negocios",
    "promedio_general": 7.8575,
    "total_estudiantes": 4
  },
  {
    "_id": "Psicología",
    "promedio_general": 7.5424999999999995,
    "total_estudiantes": 8
  },
  {
    "_id": "Ingeniería",
    "promedio_general": 6.832899999999999,
    "total_estudiantes": 5
  }
]
```

FILTRAR

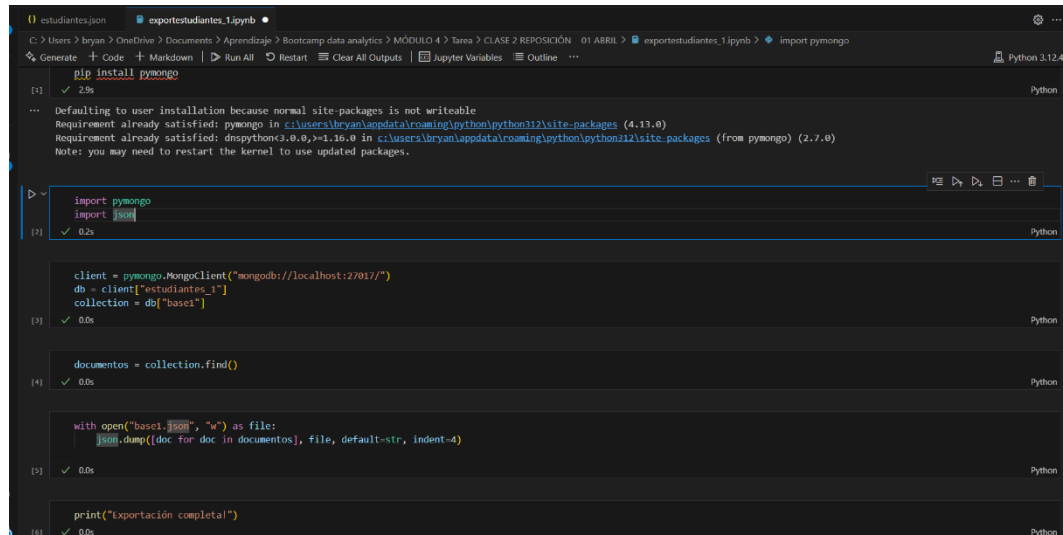
Dentro del pipeline de agregación se usa `$match` y `$group` por ejemplo para filtrar dentro de los estudiantes que están activos por carrera.



```
> use mongodb
> use estudiantes_1
> db.estudiantes_1.aggregate([
  {
    $match: { activo: true }
  },
  {
    $group: {
      _id: "$carrera",
      total_activos: { $sum: 1 }
    }
  }
])
< [
  {
    "_id": "Negocios",
    "total_activos": 3
  },
  {
    "_id": "Derecho",
    "total_activos": 4
  },
  {
    "_id": "Medicina",
    "total_activos": 3
  },
  {
    "_id": "Psicología",
    "total_activos": 3
  }
]
```

EXPORTAR USANDO PYTHON

Se genera una conexión de Python con MongoDB para poder gestionar la base de datos a través de códigos de Python. Se instaló y luego se importó la biblioteca de Python para interactuar con bases de datos MongoDB (pymongo) y finalmente se guardó el archivo en formato JSON.



```
estudiantes.json | exportestudiantes_1.ipynb
C:\Users\bryan > OneDrive > Documentos > Aprendizaje > Bootcamp data analytics > MÓDULO 4 > Tarea > CLASE 2 REPOSICIÓN 01 ABRIL > exportestudiantes_1.ipynb > import pymongo
Generate + Code + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline ... Python 3.12.4

[1] | pip install pymongo | Python
✓ 2.9s
...
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pymongo in c:\users\bryan\appdata\roaming\python\python312\site-packages (4.13.0)
Requirement already satisfied: dnspython<3.0.0,>=1.16.0 in c:\users\bryan\appdata\roaming\python\python312\site-packages (from pymongo) (2.7.0)
Note: you may need to restart the kernel to use updated packages.

[2] | import pymongo | Python
import json | Python
✓ 0.2s

[3] | client = pymongo.MongoClient("mongodb://localhost:27017/") | Python
db = client["estudiantes_1"] | Python
collection = db["base1"] | Python
✓ 0.0s

[4] | documentos = collection.find() | Python
✓ 0.0s

[5] | with open("base1.json", "w") as file: | Python
    json.dump([doc for doc in documentos], file, default=str, indent=4) | Python
✓ 0.0s

[6] | print("Exportación completa!") | Python
✓ 0.0s
```

REFLEXIÓN

El proceso de gestión de bases de datos no relacionales (NoSQL) es una habilidad muy importante también dentro del análisis de datos pues no siempre se van a analizar datos exclusivamente del tipo INT, puede haber ciertos features que de igual manera deben agruparse, filtrarse u ordenarse. Justamente creo que es una herramienta que como en nuestro ejemplo, docentes o universidades pueden utilizar para gestionar los datos de todos sus alumnos. La plataforma de compass de MongoDB es muy buena, y además la sintaxis no es tan compleja.

Algo que me pareció interesante en la sintaxis usando MongoDB es el mayor que, menor que o igual que. Estos son totalmente distintos a Python o SQL.

Algunas de las complicaciones que se tuvieron en esta tarea estuvieron relacionadas con la inicialización de Mongo, pues en mi caso tuve que usar CMD como administrador.