

Actividad:

creación de los objetos de la base de datos

GA6-220501096-AA1-EV04

Aprendiz:

Wilmer Jair Espinosa Silva

CC: 1.095.910.391

Instructor:

ISRAEL ARBONA GUERRERO

Servicio Nacional de aprendizaje-SENA

Curso: TECNOLOGÍA EN ANÁLISIS Y DESARROLLO DE SOFTWARE

Ficha: 2455285

Con base en las características del *software* a desarrollar realice el modelo relacional indicando la cardinalidad y cumpliendo con las reglas de normalización según los conceptos y ejemplos vistos en el componente Modelo entidad relación – Modelo relacional.

- Cree una base de datos NoSQL.

RTA: Para crear una base de datos en MongoDB llamada ADSO, puedes usar el comando **use** en la consola de MongoDB. Por ejemplo:

```
use ADSO
```

```
mongosh mongodb://127.0.0.1:27017/27017?directConnection=true&serverSelectionTimeoutMS=2000
Please enter a MongoDB connection string (Default: mongodb://localhost/): 27017
27017
Current Mongosh Log ID: 63cd5fdc2fa9d88ee5dafed2
Connecting to:      mongodb://127.0.0.1:27017/27017?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.6.2
Using MongoDB:      6.0.4
Using Mongosh:      1.6.2

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

-----
The server generated these startup warnings when booting
2023-01-22T10:55:41.488-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

-----
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----

27017> use adso
switched to db adso
adso>
```

Si la base de datos ya existe, esto simplemente cambiará a esa base de datos. Si no existe, se creará automáticamente al agregar una colección o al insertar un documento en ella.

Si prefieres crear la base de datos antes de usarla, puedes crearla usando el comando **db.createDatabase()** en la consola de MongoDB:

```
db.createDatabase("ADSO")
```

Es importante tener en cuenta que, para MongoDB 3.4 y versiones anteriores, el comando **db.createDatabase()** no está disponible y debería usar **use** y **db.createCollection()** en su lugar.

- Cree una colección de datos llamada "parque".

RTA: Para crear una colección de datos llamada "parque" en MongoDB, puedes usar el comando **db.createCollection()**. Por ejemplo: **db.createCollection('parque')**

```
use ADSO
db.createCollection("parque")
```

Con el primer comando estamos seleccionando la base de datos y con el segundo creamos la colección "parque" dentro de la base de datos seleccionada.

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Connecting to:  mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.6.2
Using MongoDB:  6.0.4
Using Mongosh:  1.6.2

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

-----
The server generated these startup warnings when booting
2023-01-22T10:55:41.488-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----

27017> use adso
switched to db adso
adso> db.createCollection("parque")
{ ok: 1 }
```

- Inserte cinco (5) documentos con la estructura JSON Creada.

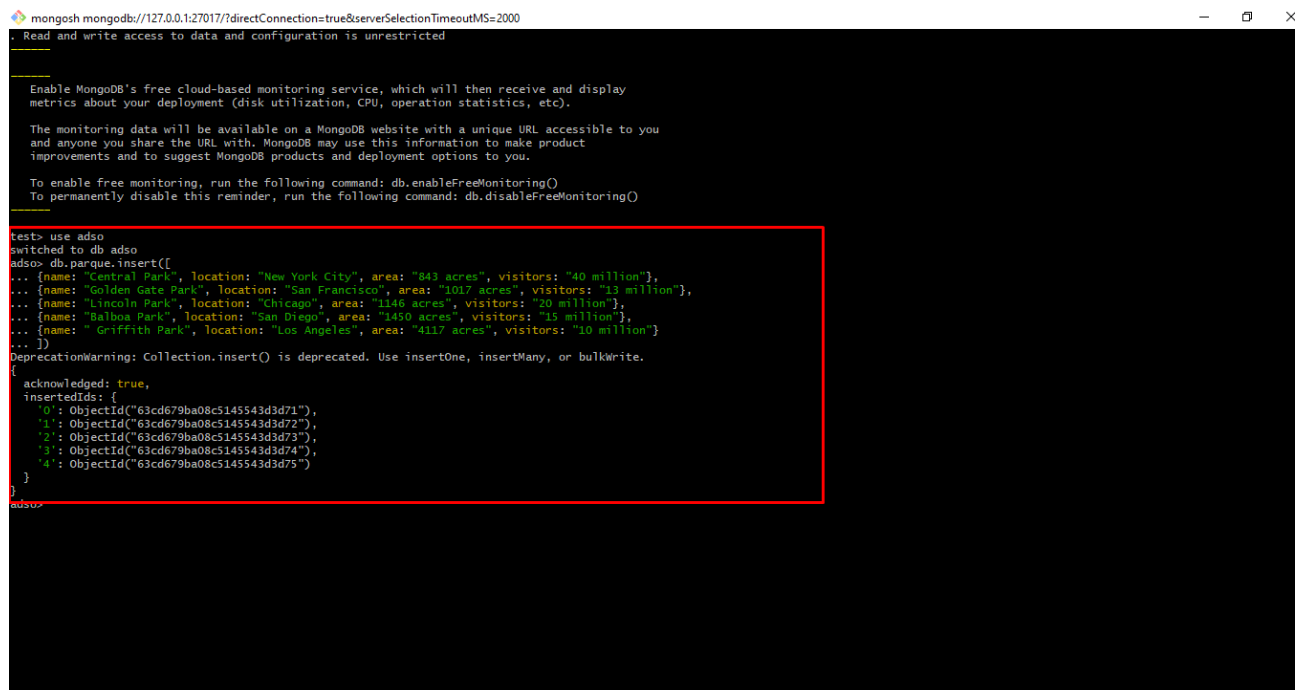
RTA: Para insertar cinco documentos con una estructura JSON específica en la colección "parque" en MongoDB, puedes usar el comando **db.collection.insert()**. Por ejemplo, si tu estructura JSON es la siguiente:

```
{
  name: "",
  location: "",
  area: "",
  visitors: ""
}
```

Podemos insertar 5 documentos con la siguiente sintaxis:

```
db.parque.insert([
  {name: "Central Park", location: "New York City", area: "843 acres", visitors:
    "40 million"},
  {name: "Golden Gate Park", location: "San Francisco", area: "1017 acres",
    visitors: "13 million"},
  {name: "Lincoln Park", location: "Chicago", area: "1146 acres", visitors: "20
    million"},
  {name: "Balboa Park", location: "San Diego", area: "1450 acres", visitors: "15
    million"},
  {name: "Griffith Park", location: "Los Angeles", area: "4117 acres", visitors:
    "10 million"}
])
```

Después de insertarse podemos validar la siguiente información



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Read and write access to data and configuration is unrestricted

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()

test> use adso
switched to db adso
adso> db.parque.insert([
... {name: "Central Park", location: "New York City", area: "843 acres", visitors: "40 million"},
... {name: "Golden Gate Park", location: "San Francisco", area: "1017 acres", visitors: "13 million"},
... {name: "Lincoln Park", location: "Chicago", area: "1146 acres", visitors: "20 million"},
... {name: "Balboa Park", location: "San Diego", area: "1450 acres", visitors: "15 million"},
... {name: "Griffith Park", location: "Los Angeles", area: "4117 acres", visitors: "10 million"}
... ])
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63cd679ba08c5145543d3d71"),
    '1': ObjectId("63cd679ba08c5145543d3d72"),
    '2': ObjectId("63cd679ba08c5145543d3d73"),
    '3': ObjectId("63cd679ba08c5145543d3d74"),
    '4': ObjectId("63cd679ba08c5145543d3d75")
  }
}
```

Cada uno de los documentos creados tiene las propiedades "name", "location", "area", "visitors" con sus respectivos valores.

Cabe mencionar que el comando **insert()** inserta uno o varios documentos en una colección, especificando el conjunto de documentos con una sintaxis similar a la de un objeto JSON.

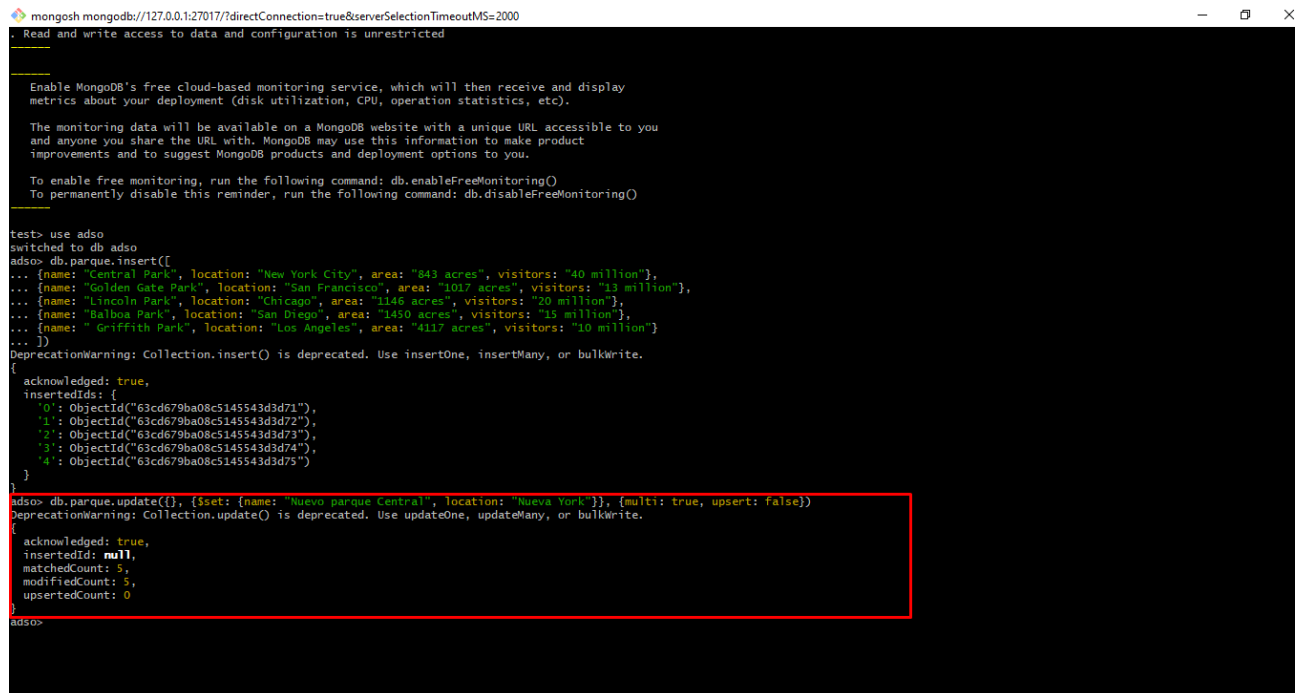
- Actualizar los datos del primer y último registro.

RTA: Para actualizar el primer registro, puedes usar el comando "db.parque.updateOne({}, {\$set: {datos_nuevos}})"

ejemplo:

```
db.parque.update({}, {$set: {name: "Nuevo parque Central", location: "Nueva York"}}, {multi: true, upsert: false})
```

Y esta acción la podemos validar de la siguiente manera



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Read and write access to data and configuration is unrestricted

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()

tests> use adso
switched to db adso
adso> db.parque.insert([
... {name: "Central Park", location: "New York City", area: "843 acres", visitors: "40 million"},
... {name: "Golden Gate Park", location: "San Francisco", area: "1017 acres", visitors: "13 million"},
... {name: "Lincoln Park", location: "Chicago", area: "1146 acres", visitors: "20 million"},
... {name: "Balboa Park", location: "San Diego", area: "1450 acres", visitors: "15 million"},
... {name: "Griffith Park", location: "Los Angeles", area: "4117 acres", visitors: "10 million"}
... ])
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63cd679ba08c5145543d3d71"),
    '1': ObjectId("63cd679ba08c5145543d3d72"),
    '2': ObjectId("63cd679ba08c5145543d3d73"),
    '3': ObjectId("63cd679ba08c5145543d3d74"),
    '4': ObjectId("63cd679ba08c5145543d3d75")
  }
}
adso> db.parque.update({}, {$set: {name: "Nuevo parque Central", location: "Nueva York"}}, {multi: true, upsert: false})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 5,
  upsertedCount: 0
}
adso>
```

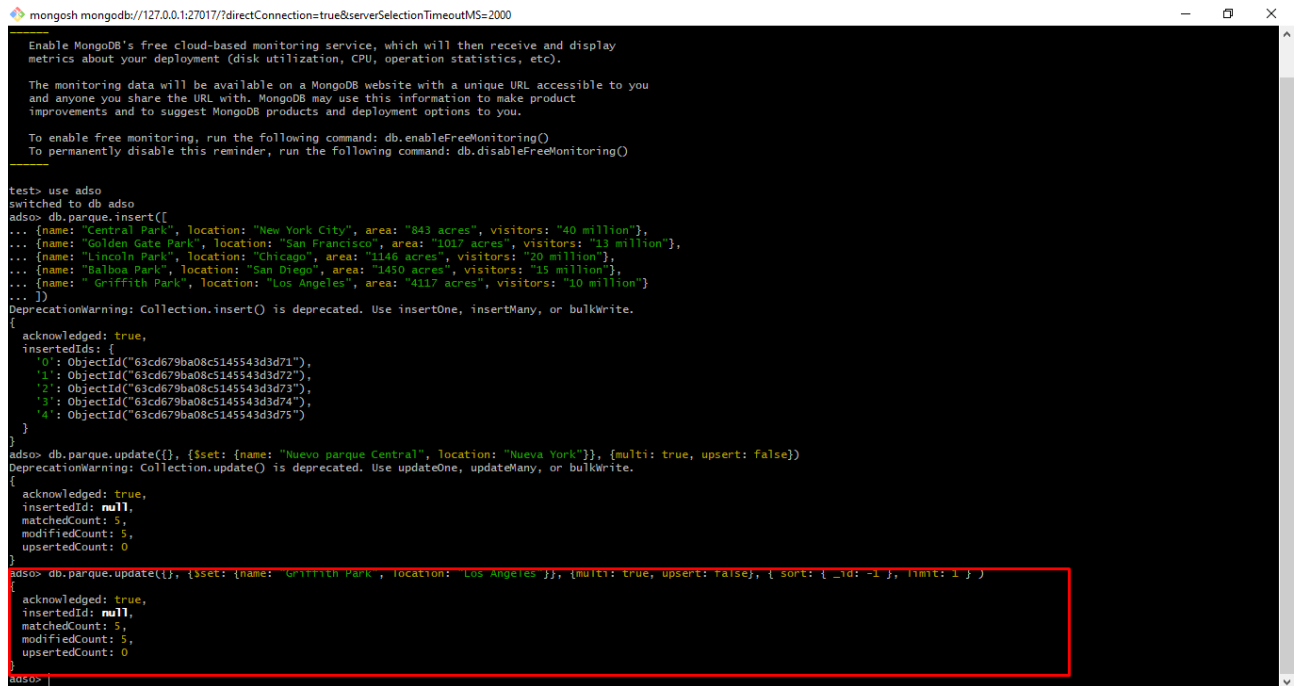
Este comando actualizará el primer registro de la colección "parque" estableciendo el nombre del parque a "Nuevo parque Central" y la ubicación a "Nueva York". el primer argumento es una consulta vacía, es decir selecciona todos los documentos, el segundo es el nuevo valor de los campos que queremos cambiar y el tercer argumento es una opción para cambiar todos los documentos.

Para actualizar el último registro, puedes usar el comando "db.parque.updateOne({}, {\$set: {datos_nuevos}}, {sort: {_id: -1}, limit: 1})"

ejemplo:

```
db.parque.updateOne({}, {$set: {"nombre": "Parque Nacional de Rocky Mountain", "ubicacion": "Colorado, Estados Unidos", "area": "4.5 km²", "visitantes_anuales": "3.4 millones"}}, {sort: {_id: -1}, limit: 1})
```

Y esta acción la podemos visualizar de la siguiente manera



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()

test> use adso
switched to db adso
adso> db.parque.insert([
... {name: "Central Park", location: "New York City", area: "843 acres", visitors: "40 million"},
... {name: "Golden Gate Park", location: "San Francisco", area: "1017 acres", visitors: "13 million"},
... {name: "Lincoln Park", location: "Chicago", area: "1146 acres", visitors: "20 million"},
... {name: "Balboa Park", location: "San Diego", area: "1450 acres", visitors: "15 million"},
... {name: "Griffith Park", location: "Los Angeles", area: "4117 acres", visitors: "10 million"}
... ])
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63cd679ba08c5145543d3d71"),
    '1': ObjectId("63cd679ba08c5145543d3d72"),
    '2': ObjectId("63cd679ba08c5145543d3d73"),
    '3': ObjectId("63cd679ba08c5145543d3d74"),
    '4': ObjectId("63cd679ba08c5145543d3d75")
  }
}
adso> db.parque.update({}, {$set: {name: "Nuevo parque Central", location: "Nueva York"}}, {multi: true, upsert: false})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 5,
  upsertedCount: 0
}
adso> db.parque.update({}, {$set: {name: "Griffith Park", location: "Los Angeles"}}, {multi: true, upsert: false}, {sort: {_id: -1}, limit: 1})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 5,
  upsertedCount: 0
}
```

- Liste la colección completa.

RTA: Para listar todos los documentos en una colección en MongoDB, puedes usar el comando `db.collection.find()` sin ningún argumento. Por ejemplo, para listar todos los documentos en la colección "parque":

```
db.parque.find()
```

Y se visualiza de la siguiente manera

mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000

```
{
  acknowledged: true,
  insertedIds: {
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 5,
      modifiedCount: 5,
      upsertedCount: 0
    }
  }
}
ads> db.parque.find()
[
  {
    _id: ObjectId("63cd679ba08c5145543d3d71"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '943 acres',
    visitors: '40 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d72"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '1017 acres',
    visitors: '13 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d73"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '1146 acres',
    visitors: '20 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d74"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '1450 acres',
    visitors: '15 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d75"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '4117 acres',
    visitors: '10 million'
  }
]
ads> |
```

También puedes utilizar el comando **db.collection.find().pretty()** para mostrar los resultados en un formato más legible:

```
db.parque.find().pretty()
```

En ambos casos, se mostrarán todos los documentos en la colección "parque", con todos sus campos y valores.

Es importante mencionar que el comando **find()** devuelve un cursor, el cual es un objeto especial que permite iterar sobre los resultados de una consulta. si deseas mostrar todos los resultados del cursor en una sola vez, puedes utilizar el método **toArray()** sobre el cursor.

```
db.parque.find().toArray()
```

También puedes limitar los resultados utilizando el método **limit()**.

```
db.parque.find().limit(3)
```

Este comando devolverá solo los primeros 3 documentos de la colección.

- Borre el tercer documento de la colección parque.

RTA: Para eliminar un documento específico en una colección en MongoDB, puedes usar el comando **db.collection.remove()** y especificar un criterio para identificar el documento a eliminar. Por ejemplo, para eliminar el tercer documento de la colección "parque", puedes usar la siguiente sintaxis:

```
db.parque.remove({}, { justOne: true }, { sort: { _id: 1 }, skip: 2 } )
```

Este comando eliminará el tercer documento en la colección "parque". El primer argumento es una consulta vacía, es decir selecciona todos los documentos, el segundo argumento indica que solo queremos eliminar un solo documento y el tercer argumento especifica que queremos seleccionar el tercer documento ordenado por `_id` y saltando los primeros dos documentos.

Es importante mencionar que el comando **remove()** elimina de forma permanente los documentos y no tiene opción de deshacer la acción. Si quieres mantener un registro de los documentos eliminados, puedes considerar utilizar el comando **update()** y agregar un campo "eliminado" con el valor "si" o similar.

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 5,
  modifiedCount: 5,
  upsertedCount: 0
}
ads@> db.parque.find()
[
  {
    _id: ObjectId("63cd679ba08c5145543d3d71"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '843 acres',
    visitors: '40 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d72"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '1017 acres',
    visitors: '13 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d73"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '1146 acres',
    visitors: '20 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d74"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '1450 acres',
    visitors: '15 million'
  },
  {
    _id: ObjectId("63cd679ba08c5145543d3d75"),
    name: 'Griffith Park',
    location: 'Los Angeles',
    area: '4117 acres',
    visitors: '10 million'
  }
]
ads@> db.parque.remove({}, { justOne: true }, { sort: { _id: 1 }, skip: 2 })
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 1 }
ads@>
```


- Liste la colección de datos completa.

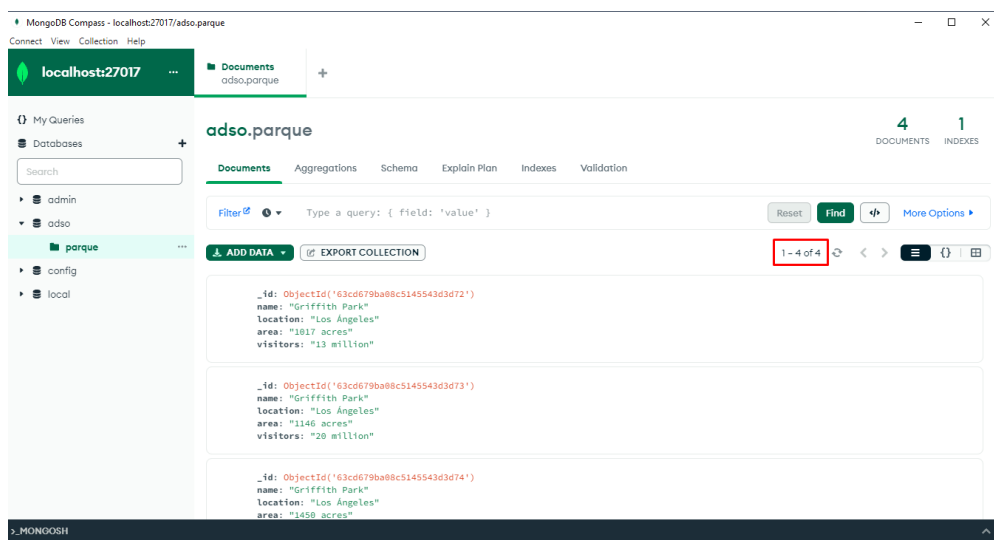
RTA: Para listar todos los documentos en una colección en MongoDB, puedes usar el comando `db.collection.find()` sin ningún argumento. Por ejemplo, para listar todos los documentos en la colección "parque":

```
db.parque.find()
```

Y al ejecutar el comando el resultado es el siguiente

```
mongosh mongodb://127.0.0.1:27017/?directConnections=true&serverSelectionTimeoutMS=2000
{
  "_id": ObjectId("63cd679ba08c5145543d3d74"),
  "name": "Griffith Park",
  "location": "Los Angeles",
  "area": "1450 acres",
  "visitors": "15 million"
},
{
  "_id": ObjectId("63cd679ba08c5145543d3d75"),
  "name": "Griffith Park",
  "location": "Los Angeles",
  "area": "4117 acres",
  "visitors": "10 million"
}
}
adso> db.parque.remove({}, { justOne: true }, { sort: { _id: 1 }, skip: 2 })
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 1 }
adso> db.parque.find()
[
  {
    "_id": ObjectId("63cd679ba08c5145543d3d72"),
    "name": "Griffith Park",
    "location": "Los Angeles",
    "area": "1017 acres",
    "visitors": "13 million"
  },
  {
    "_id": ObjectId("63cd679ba08c5145543d3d73"),
    "name": "Griffith Park",
    "location": "Los Angeles",
    "area": "1146 acres",
    "visitors": "20 million"
  },
  {
    "_id": ObjectId("63cd679ba08c5145543d3d74"),
    "name": "Griffith Park",
    "location": "Los Angeles",
    "area": "1450 acres",
    "visitors": "15 million"
  },
  {
    "_id": ObjectId("63cd679ba08c5145543d3d75"),
    "name": "Griffith Park",
    "location": "Los Angeles",
    "area": "4117 acres",
    "visitors": "10 million"
  }
]
```

Al validar el proceso por Mongo DB Compass podemos visualizar que el proceso se realizó con éxito



También puedes utilizar el comando `db.collection.find().pretty()` para mostrar los resultados en un formato más legible:

```
db.parque.find().pretty()
```

En ambos casos, se mostrarán todos los documentos en la colección "parque", con todos sus campos y valores.

Es importante mencionar que el comando **find()** devuelve un cursor, el cual es un objeto especial que permite iterar sobre los resultados de una consulta. si deseas mostrar todos los resultados del cursor en una sola vez, puedes utilizar el método **toArray()** sobre el cursor.

```
db.parque.find().toArray()
```

También puedes limitar los resultados utilizando el método **limit()**.

```
db.parque.find().limit(3)
```

Este comando devolverá solo los primeros 3 documentos de la colección.

Cabe mencionar que si no deseas mostrar todos los campos de cada documento, puedes especificar los campos específicos que deseas mostrar en el comando **find()** utilizando el parametro **projection** y especificando los campos que deseas mostrar.

```
db.parque.find({}, {name:1, location:1, _id:0})
```

Este comando devolverá solo los campos "name" y "location" de todos los documentos en la colección "parque" sin mostrar el campo "_id".