UNIR - Universidad Internacional de la Rioja

Ciudad de México

**ACTIVIDAD 1:**

**Uso de MongoDB**

Alumno: Cuenca Roa, Leonard Jose

Grupo: 1001

**ÍNDICE**

Carga de Datos 3

Evidencia carga de datos 3

Explora las colecciones 4

Identifica todas las distintas categorías (categories) de la colección books 4

Identifica los distintos estados (status) de la colección books 5

Describe brevemente qué arroja la siguiente consulta 5

Utiliza la condición de la consulta anterior para recuperar aquellos libros que posean exactamente 2 autores y que estén publicados 6

Describe brevemente qué ocurre si a la consulta del punto anterior le añades al final la siguiente instrucción 6

Consulta la colección 1 7

Consulta la colección 2 7

Carga de Datos

De la carga de datos, deseo resaltar lo que pude aprender y comprender qué el objetivo es organizar y almacenar datos sobre libros y compañías en una base de datos NoSQL, utilizando MongoDB y su herramienta de importación.

Resaltando la satisfacción y la ejecución de las siguientes actividades:

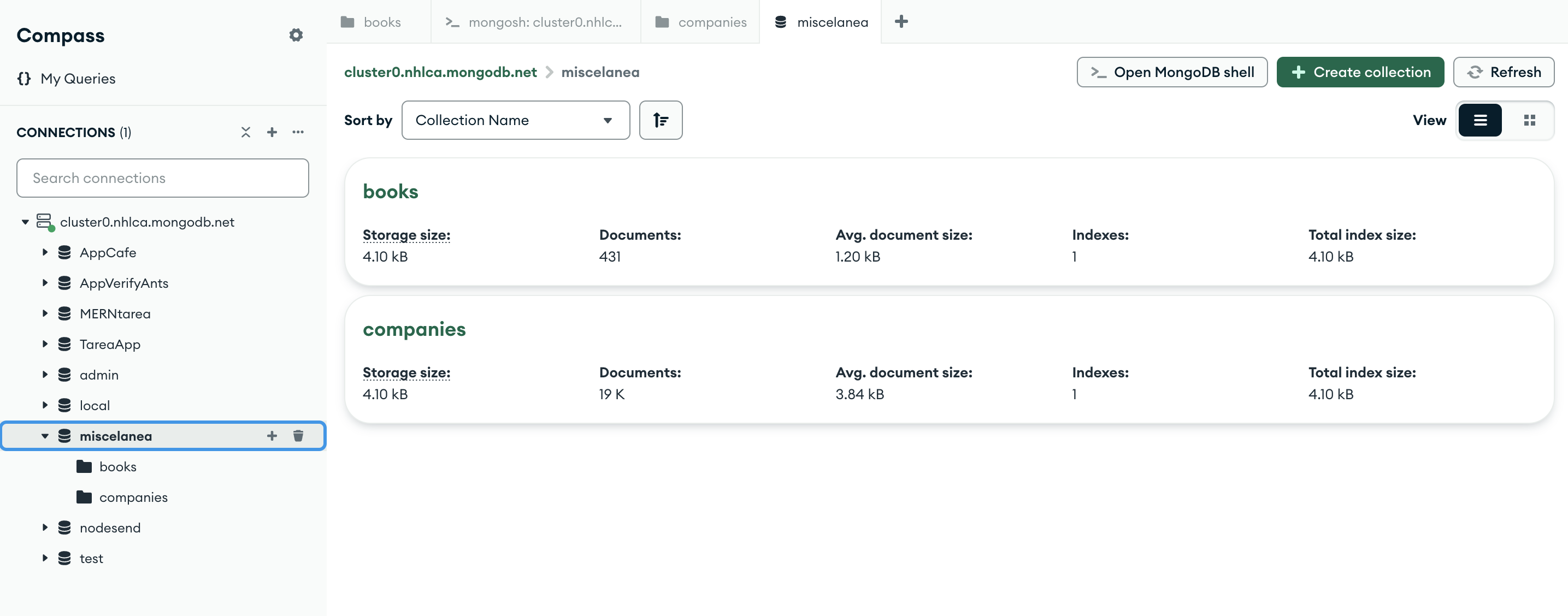
**Creación de una base de datos**: Se debe crear una nueva base de datos con el nombre "miscelanea".

**Creación de colecciones**: Dentro de la base de datos, se creó las dos colecciones: "books" y "companies".

**Importación de datos**: Los datos se cargó desde archivos JSON a las colecciones correspondientes utilizando la herramienta.

**Herramienta a utilizar**: Se adoptó una estrategia híbrida que integra la línea de comandos de MongoDB con las capacidades visuales de MongoDB Compass. Esta metodología fue seleccionada para llevar a cabo la importación y validación de datos, aprovechando las herramientas de última generación ofrecidas por Atlas. Al combinar la precisión de la línea de comandos con la facilidad de uso de la interfaz gráfica, se logró un proceso más eficiente y controlado.

Evidencia carga de datos



Evidencia de base de datos y sus colecciones creadas.

Explora las colecciones

Identifica todas las distintas categorías (categories) de la colección books

Para resolver este ejercicio, se implementó una serie de funciones que ofrece MongoDB. En primer lugar, identifico la colección con la que quiero trabajar utilizando **getCollection**(), especificando la colección **books**. Luego, empleo el método **aggregate**(), que permite realizar operaciones complejas sobre una colección. Luego, utilizo el operador $**unwind** para descomponer el campo '**categories**' de cada documento en registros individuales. Una vez hecho esto, procedo a agrupar los resultados utilizando el operador $**group** para agrupar por el campo deseado utilizando la propiedad $**addToSet** ya que, toma el valor del campo **categories** de cada documento y lo agrega a un nuevo conjunto **(set),** con el propósito de que solo agrega un valor al conjunto si este aún no existe en caso contrario si un mismo valor de categoría aparece en múltiples documentos, solo se incluirá una vez en el conjunto final y para finalizar, utilizo el operador $**project** para seleccionar los campos que quiero incluir en el resultado final, ocultando aquellos que no son necesarios.

| Código | Evidencia |
| --- | --- |
| db.getCollection('books').aggregate([  { $unwind: '$categories' },  { $group: {\_id: null, categories: { $addToSet: '$categories' }}},  { $project: { \_id: 0, categories: 1 } }  ]); |  |
|
|
|

Identifica los distintos estados (status) de la colección books

En este ejercicio, identificamos tres posibles soluciones para obtener registros únicos a partir de una colección. Utilizando el operador **distinct**: Este operador nos permite listar todos los campos de una colección, eliminando los duplicados y mostrando únicamente los valores únicos. Consultando directamente la colección de manera similar al primer caso, podemos obtener los mismos resultados accediendo directamente a la colección utilizando **db.getCollection()** otra opción es agrupando los resultados con **$group,** para agrupar los datos por un campo específico para este caso el **estatus** y obtener valores únicos, podemos emplear el operador $**group** dentro de una etapa de **agregación.**

| Código | Evidencia |
| --- | --- |
| //Opción Uno  db.books.distinct("status")  //Opción Dos  db.getCollection('books').distinct("status")  //Opción Tres  db.getCollection('books').aggregate(  [{ $group: { \_id: '$status' } }]  ); |  |
|
|
|

Describe brevemente qué arroja la siguiente consulta

Esta consulta busca en la colección **books** todos los libros cuyo campo **longDescription** comience por una letra que esté entre la letra "A" y la letra "B" sin incluir la letra "B". De todos los **books** que cumplan esta condición, solo se mostrarán los campos **title** y **longDescription.**

| Código | Evidencia |
| --- | --- |
| db.getCollection('books').find({longDescription: {$gte: "A", $lt: "B"}}, {title: 1, longDescription: 1}) | Captura de pantalla 2024-11-15 a la(s) 0.33.30.png |
|
|
|

Utiliza la condición de la consulta anterior para recuperar aquellos libros que posean exactamente 2 autores y que estén publicados

Muestra solo los campos: title, longDescription, status y authors.

| Código | Evidencia |
| --- | --- |
| db.getCollection('books').find({  status: "PUBLISH",  authors: { $size: 2 }  }, {  title: 1,  longDescription: 1,  status: 1,  authors: 1  }) |  |
|
|
|

Describe brevemente qué ocurre si a la consulta del punto anterior le añades al final la siguiente instrucción

.toArray()

Al añadirle la propiedad este genera un arreglo de objetos con todos los resultados, en cambio la consulta anterior genera un cursor para iterar sobre los resultados, esta pequeña diferencia depende de la necesidades que se desees trabajar, por ejemplo si necesitas procesar los resultados de forma incremental ó cuando tienes un gran número de resultados y quieres evitar cargarlos todos a la memoria a la vez ó cuando quieres iterar sobre los resultados varias veces **usas el cursor** pero cuando necesitas trabajar con todos los resultados a la vez o cuando quieres pasar los resultados a otra función o biblioteca debes usar **un arreglo de objetos**.

| Código | Evidencia |
| --- | --- |
| db.getCollection('books').find({  status: "PUBLISH",  authors: { $size: 2 }  }, {  title: 1,  longDescription: 1,  status: 1,  authors: 1  }).toArray() |  |
|
|
|

Consulta la colección 1

Consulta la colección 2