**🔎 ¿Qué es $lookup en MongoDB?**

En MongoDB no hay *joins* como en SQL, pero con **$lookup** podés relacionar documentos de diferentes colecciones dentro de un aggregate.

📌 **Sintaxis básica:**

db.coleccion1.aggregate([

{

$lookup: {

from: "coleccion2", // colección con la que quiero unir

localField: "campo1", // campo en coleccion1

foreignField: "campo2", // campo en coleccion2

as: "resultado" // nombre del array donde guardo la unión

}

}

])

👉 $lookup siempre genera un **array** (resultado). Si querés “aplanarlo” usás $unwind.

**📚 Ejemplo con datos de la API (libros OpenLibrary)**

Supongamos que en la API insertaste en **libros** algo así:

{

"titulo": "The Hobbit",

"autores": ["J.R.R. Tolkien"],

"anio": 1937,

"temas": ["fantasy", "adventure"]

}

Y además en otra colección **autores** cargaste:

{

"nombre": "J.R.R. Tolkien",

"nacionalidad": "Reino Unido",

"premios": ["Orden del Imperio Británico"]

}

**Ejemplo 1 – Relacionar libros con autores**

db.libros.aggregate([

{

$lookup: {

from: "autores",

localField: "autores", // viene como array

foreignField: "nombre", // campo en autores

as: "infoAutores"

}

}

])

🔎 Resultado: cada libro tendrá un array infoAutores con los datos de los autores.

**Ejemplo 2 – Desarmar con $unwind**

db.libros.aggregate([

{

$lookup: {

from: "autores",

localField: "autores",

foreignField: "nombre",

as: "infoAutores"

}

},

{ $unwind: "$infoAutores" } // cada libro-autor queda como documento separado

])

**📝 Ejercicios con $lookup (para hacer en consola)**

Estos los armé en función de los datos de **libros de la API** más los que ya tenés de **empleados** y **películas** en tus documentos de práctica.

**🔹 Ejercicio 1**

Unir **libros** con **autores** y mostrar:

* título del libro,
* nombre del autor,
* nacionalidad.

**🔹 Ejercicio 2**

Unir **películas** con **empleados**, considerando que algunos empleados son críticos y su nombre aparece en criticas.autor.

* Mostrar película, autor de la crítica y legajo del empleado.

💡 pista: $lookup + $unwind.

**🔹 Ejercicio 3**

Relacionar **empleados** con **libros**, asumiendo que agregamos un campo ciudadTrabajo en empleados que puede coincidir con la editorial.ciudad.

* Mostrar empleados y cuántos libros se editaron en su ciudad.

💡 pista: $lookup + $group.

**🔹 Ejercicio 4**

Hacer un **lookup encadenado**:

* Unir empleados → críticas de películas → películas → autores de esas películas.
* Mostrar: *Empleado – Crítica – Director de la película*.

**🔹 Ejercicio 5 (Desafío final con pipeline largo)**

1. Desarmar críticas de películas ($unwind).
2. Hacer $lookup con empleados (autor = nombre).
3. Agrupar por empleado y contar cuántas críticas escribió.
4. Ordenar de mayor a menor y mostrar los **3 críticos más activos**.

**===============================**

**📌 5 Ejercicios Simples de $lookup**

**🔹 Ejercicio 1 – Libros y Editoriales**

Relacionar libros con su editorial (ya está embebida, pero usalo como práctica).  
👉 Mostrar título del libro y ciudad de la editorial.

**🔹 Ejercicio 2 – Películas y Premios**

Unir películas con otras películas que tengan el mismo valor en premios.oscar.  
👉 Mostrar título y si ganó o no Oscar.

**🔹 Ejercicio 3 – Empleados y Tareas**

Unir empleados con otros empleados que compartan al menos una tarea.  
👉 Mostrar nombre del empleado y cantidad de compañeros con misma tarea.

**🔹 Ejercicio 4 – Libros y Años de Publicación**

Unir libros con otros libros publicados en el mismo año.  
👉 Mostrar título y cuántos libros salieron ese año.

**🔹 Ejercicio 5 – Películas y Directores**

Unir películas con otras películas del mismo director.  
👉 Mostrar título y cantidad de películas que hizo ese director.

**===============================**

**Práctica de MongoDB – Agregación, Lookups y Expresiones Regulares**

**📌 Parte 1 – Carga de datos**

En esta sección se cargan datos desde una API pública hacia MongoDB local.  
El límite se puede ajustar para traer 100, 500 o 1000 registros.

import requests

from pymongo import MongoClient, UpdateOne

# 🔗 Conexión a tu cluster Atlas

client = MongoClient("mongodb+srv://user:user@casa.tgh4en9.mongodb.net/?retryWrites=true&w=majority&appName=Casa")

db = client["333practica"]

# ============================================

# 1) Colección LIBROS desde OpenLibrary API

# ============================================

url = "https://openlibrary.org/subjects/fantasy.json?limit=100"

data = requests.get(url).json()

libros = []

autores = []

for libro in data.get("works", []):

lista\_autores = [a["name"] for a in libro.get("authors", [])]

libros.append({

"titulo": libro["title"],

"autores": lista\_autores,

"anio": libro.get("first\_publish\_year"),

"temas": libro.get("subject", []),

"editorial": {

"ciudad": "Londres", # valor de ejemplo

"nombre": "Editorial Fantasía"

}

})

# Autores para colección separada

for a in libro.get("authors", []):

autores.append({

"nombre": a["name"],

"id": a["key"],

# campo extra para practicar

"nacionalidad": "Desconocida"

})

if libros:

db.libros.drop()

db.libros.insert\_many(libros)

print(f"✅ Insertados {len(libros)} libros en MongoDB")

if autores:

db.autores.drop()

ops = []

for autor in autores:

ops.append(UpdateOne({"id": autor["id"]}, {"$set": autor}, upsert=True))

db.autores.bulk\_write(ops)

print(f"✅ Insertados/actualizados {len(autores)} autores en MongoDB")

# ============================================

# 2) Colección PELICULAS (datos fijos de práctica)

# ============================================

peliculas = [

{

"titulo": "Inception",

"director": "Christopher Nolan",

"estreno": "2010-07-16",

"criticas": [

{"autor": "Juan Perez", "comentario": "Brillante"},

{"autor": "Lucía Gómez", "comentario": "Un poco confusa"}

]

},

{

"titulo": "Titanic",

"director": "James Cameron",

"estreno": "1997-12-19",

"criticas": [

{"autor": "Lucía Gómez", "comentario": "Emotiva"},

{"autor": "Martín Díaz", "comentario": "Muy larga"}

]

},

{

"titulo": "El Padrino",

"director": "Francis Ford Coppola",

"estreno": "1972-03-24",

"criticas": [

{"autor": "Carlos", "comentario": "Una obra maestra"},

{"autor": "Juan Perez", "comentario": "Clásico absoluto"}

]

}

]

db.peliculas.drop()

db.peliculas.insert\_many(peliculas)

print(f"✅ Insertadas {len(peliculas)} películas en MongoDB")

# ============================================

# 3) Colección EMPLEADOS (datos fijos de práctica)

# ============================================

empleados = [

{

"nombre": "Juan Perez",

"legajo": 10001,

"ciudadTrabajo": "Bogotá",

"tareas": ["planillas", "reportes"],

"activo": True

},

{

"nombre": "Lucía Gómez",

"legajo": 10002,

"ciudadTrabajo": "Londres",

"tareas": ["atención al cliente"],

"activo": True

},

{

"nombre": "Martín Díaz",

"legajo": 10003,

"ciudadTrabajo": "Buenos Aires",

"tareas": ["reuniones", "soporte técnico"],

"activo": False

},

{

"nombre": "Carla Ruiz",

"legajo": 10004,

"ciudadTrabajo": "Mar del Plata",

"tareas": ["reportes", "reuniones"],

"activo": True

}

]

db.empleados.drop()

db.empleados.insert\_many(empleados)

print(f"✅ Insertados {len(empleados)} empleados en MongoDB")

print("🚀 Base de datos lista con colecciones: libros, autores, peliculas, empleados")

**📌 Parte 2 – Agregaciones Básicas $match** → libros publicados después de 2015

1. **$project** → mostrar solo título y primer autor
2. **$sort** → ordenar empleados por fecha de ingreso
3. **$limit + $skip** → traer los 5 libros más nuevos, saltando los primeros 2
4. **$count** → contar cuántos empleados están activos
5. **$sortByCount** → contar películas por género

**📌 Parte 3 – Expresiones Regulares**

1. Buscar libros cuya ciudad editorial contenga "mar" → /mar/i
2. Empleados cuyo nombre empiece con "Lu"
3. Películas cuyos comentarios contengan exactamente la palabra "obra"
4. Críticas que incluyan "obra" seguido de "maestra" → /obra(?=.\*maestra)/i
5. Libros cuyo título **no** contenga vocales repetidas → /^(?!.\*(aa|ee|ii|oo|uu))/i

**📌 Parte 4 – Lookup y Combinaciones**

1. $lookup: empleados ↔ libros (ciudad editorial = ciudadTrabajo)
2. $lookup + $unwind: unir empleados con críticas de películas donde el autor = nombre empleado
3. $lookup + $group: contar cuántas críticas hizo cada empleado
4. $lookup encadenado: libros ↔ películas ↔ empleados (mostrar conexiones indirectas)
5. **Desafío largo:**
   * Obtener las 3 ciudades que más empleados críticos tienen
   * Para esas ciudades, calcular el promedio de páginas de libros publicados
   * Mostrar solo si el promedio > 250 páginas

**📌 Parte 5 – Proyecto Final**

Pipeline grande que combine todo:

1. **$unwind** críticas de películas
2. **$match** críticas con más de 100 likes
3. **$lookup** con empleados (autor ↔ nombre)
4. **$group** por empleado y contar cantidad de críticas
5. **$sort** descendente
6. **$limit** a top 5 críticos

**📌 6 Ejercicios Combinados**

**🔹 Ejercicio 1 – Empleados y Libros en la misma ciudad**

Unir empleados con libros publicados en su ciudadTrabajo.  
👉 Mostrar empleado, ciudad y cantidad de libros encontrados.

**🔹 Ejercicio 2 – Películas y Empleados críticos**

Unir películas con empleados que escribieron críticas.  
👉 Mostrar película, autor de crítica y si ese autor está en empleados.

**🔹 Ejercicio 3 – Top 3 directores más criticados por empleados**

1. Desarmar críticas.
2. Relacionar con empleados (si el autor es empleado).
3. Agrupar por director y contar.
4. Ordenar y limitar a 3.

**🔹 Ejercicio 4 – Libros por década**

1. Agrupar libros por década (año).
2. Mostrar cantidad de libros por década.

**🔹 Ejercicio 5 – Regex en ciudades (con $lookup)**

Unir empleados con libros cuando ciudadTrabajo contiene "mar" (ej: "Mar del Plata").  
👉 Mostrar empleado y títulos de libros de esa ciudad.

**🔹 Ejercicio 6 – Regex en críticas de películas**

Unir películas con empleados que hicieron críticas que **contienen la palabra “obra”**.  
👉 Mostrar empleado, película y el comentario.