Imagen que contiene Diagrama

El contenido generado por IA puede ser incorrecto.

Universidad Tecnológica de Durango

ING. Desarrollo y Gestión de Software

Administración de Base de Datos

*“Documentación”*

Alumno:

* Contreras Arévalos José Uriel
* Forester López José Bernardo
* Ramírez Martínez Natalia
* Rocha Chacón Adrián
* Soto Diaz José Miguel

8°A

Docente:

* ISC Alberto Antonio Vásquez Orozco

Marzo de 2025

# Introducción

## Descripción de la base de datos

La base de datos del servicio de streaming combina PostgreSQL y MongoDB para optimizar rendimiento y flexibilidad. PostgreSQL gestiona datos estructurados que requieren integridad referencial, como usuarios, reseñas e historial de visualización. Cada usuario tiene un perfil que le permite acceder al catálogo y registrar sus actividades.

MongoDB almacena datos dinámicos y semiestructurados, como películas, series y log de actividades, lo que permite mayor flexibilidad en la gestión de metadatos y el seguimiento de eventos en la plataforma.

Esta arquitectura híbrida optimiza el almacenamiento y acceso a la información, asignando cada tipo de dato al sistema más eficiente según su estructura y escalabilidad.

## Propósito

El propósito de esta base de datos es proporcionar una infraestructura eficiente para gestionar grandes volúmenes de datos en un servicio de streaming, optimizando consultas, minimizando redundancias y asegurando la integridad de la información. También facilita la administración del catálogo de contenido y el seguimiento de las preferencias de los usuarios, garantizando una experiencia fluida.

## Objetivo

El objetivo de la base de datos es proporcionar una solución eficiente y estructurada para almacenar y gestionar la información de usuarios, contenido y actividades dentro del servicio de streaming, garantizando una experiencia fluida para cada espectador. Se busca que la base de datos permita una gestión rápida y segura del acceso a la plataforma, así como un registro detallado de las interacciones de los usuarios con el contenido. Además, pretende garantizar la seguridad y privacidad de la información, protegiendo los datos personales de los usuarios y asegurando que solo el contenido adecuado sea accesible de acuerdo con los permisos establecidos.

## Tecnologías utilizadas

Para el desarrollo y gestión de la base de datos del servicio de streaming, se están utilizando dos gestores de bases de datos: PostgreSQL y MongoDB. Esta combinación permite evaluar el desempeño y las ventajas de cada uno en diferentes aspectos del sistema, obteniendo una comparación práctica entre un modelo relacional y uno NoSQL.

*PostgreSQL: Base de Datos Relacional*

PostgreSQL es un sistema de gestión de bases de datos relacional que se ha elegido debido a su robustez y cumplimiento con estándares SQL. Su modelo relacional facilita la normalización de datos, asegurando integridad, reducción de redundancias y cumplimiento de reglas de negocio. Además, su capacidad para manejar transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) garantiza que todas las operaciones de la base de datos sean consistentes, incluso en entornos de alta concurrencia.

En este proyecto, PostgreSQL se está utilizando exclusivamente para almacenar datos estructurados y críticos que requieren un modelo altamente relacional. Específicamente, PostgreSQL maneja las siguientes tablas principales:

1. **Usuarios**: Contiene la información de los usuarios, sus credenciales y roles de acceso.
2. **Reseñas**: Almacena las calificaciones y comentarios que los usuarios dejan sobre las películas y series.
3. **Historial de Visualización**: Registra qué contenido ha visto cada usuario y en qué momento.

Para garantizar la integridad referencial, se han definido claves primarias y foráneas entre estas tablas, asegurando que cada usuario tenga un historial de visualización correcto y que las reseñas estén asociadas a usuarios existentes.

*MongoDB: Base de Datos NoSQL*

MongoDB es un gestor de bases de datos NoSQL orientado a documentos. Se ha incorporado a este proyecto con el propósito de evaluar su flexibilidad y rendimiento en el almacenamiento de información que no requiere una estructura rígida y que puede crecer dinámicamente.

MongoDB se ha utilizado para las siguientes colecciones de datos:

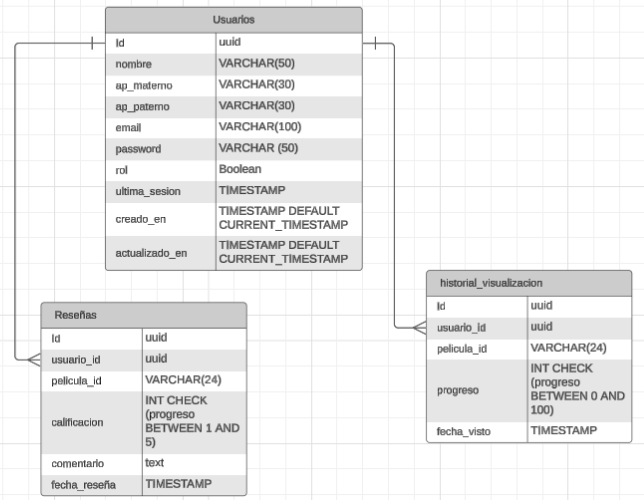
1. **Películas**: Contiene la información de las películas disponibles en la plataforma, incluyendo título, género, duración y descripciones.
2. **Series**: Similar a la colección de películas, pero estructurada para permitir múltiples episodios por serie.
3. **Log de Actividades**: Registra eventos del sistema, como intentos de inicio de sesión, cambios de perfil y otras acciones relevantes de los usuarios.

MongoDB permite almacenar estos datos en documentos JSON flexibles, lo que facilita consultas eficientes y escalabilidad. Esto es especialmente útil para el log de actividades, ya que los eventos pueden variar en estructura y no requieren la rigidez de un esquema relacional.

El uso combinado de PostgreSQL y MongoDB permite aprovechar lo mejor de cada tecnología, **postgreSQL** gestiona la información que requiere **integridad referencial y consistencia**, como usuarios, reseñas e historial de visualización, **mongoDB** maneja datos que pueden cambiar de estructura con el tiempo, como películas, series y el log de actividades.

Esta estrategia permite optimizar el rendimiento del sistema, asegurando que cada tipo de dato se almacene en la tecnología más adecuada, logrando un equilibrio entre rendimiento, flexibilidad y seguridad.

# Diagramas

[](https://lucid.app/lucidchart/e5a5acc6-41ed-43bb-9613-6460da4393fc/edit?viewport_loc=179%2C104%2C1714%2C797%2C0_0&invitationId=inv_60f48ccb-2f73-4cff-afe8-9c41c49c2dd0)

[Diagrama 1 Modelo ER de la BD](https://lucid.app/lucidchart/e5a5acc6-41ed-43bb-9613-6460da4393fc/edit?viewport_loc=179%2C104%2C1714%2C797%2C0_0&invitationId=inv_60f48ccb-2f73-4cff-afe8-9c41c49c2dd0)

El modelo de la base de datos para el servicio de streaming sigue un enfoque relacional en PostgreSQL, estructurando los datos en tablas normalizadas para optimizar el almacenamiento y las consultas. Las principales entidades del sistema incluyen **usuarios, contenido, historial de actividad y reseñas**, estableciendo relaciones entre ellas para garantizar la coherencia e integridad de los datos.

1. **Usuarios** – Almacena la información de los usuarios registrados.
2. **Historial de Visualización** – Registra las visualizaciones del usuario.
3. **Reseñas** – Guarda las opiniones dadas por los usuarios.

Este modelo de la base de datos se estructura en las entidades principales mencionadas anteriormente con relaciones bien definidas:

* **Usuarios** interactúan con contenido y generan registros en **Historial de Visualizaciones**.
* **Usuarios** pueden dejar **Reseñas** sobre el contenido que han visto.

Este modelo permite almacenar la información de manera eficiente, garantizando integridad referencial y facilitando consultas optimizadas para mejorar la experiencia del usuario dentro del servicio de streaming.

# Estructura de las tablas

## Tabla usuarios

La tabla usuarios almacena la información de los usuarios registrados en la plataforma, incluyendo sus credenciales de acceso y nivel de permisos dentro del sistema.

|  |  |  |  |
| --- | --- | --- | --- |
| Campo | Tipo de Dato | Restricciones | Descripción |
| idUsuario | uuid | Primary Key | Identificador único del usuario |
| nombre | varchar (50) | |  | | --- | | Not Null |  |  | | --- | |  | | |  | | --- | | Nombre del usuario |  |  | | --- | |  | |
| ap\_paterno | varchar (30) | |  | | --- | | Not Null |  |  | | --- | |  | | |  | | --- | | Apellido paterno del usuario |  |  | | --- | |  | |
| ap\_materno | Varchar(30) |  | Apellido materno del usuario |
| Email | varchar (100) | Unique, Not Null | Correo electrónico del usuario |
| Password | varchar (50) | Not Null | Contraseña encriptada |
| rol | boolean |  | Rol del usuario y nivel de permisos |
| ultima\_sesion | timestamp | Default Now() | Fecha de ultimo inicio de sesión de la cuenta |
| creado\_en | timestamp | Default Now() | Fecha de creación de la cuenta |
| actualizado\_en | timestamp | Default Now() | Fecha de actualización de datos de la cuenta |

## Relaciones de usuarios

* Un usuario puede ver múltiples contenidos, registrando sus interacciones en el historial de visualizaciones.
* Un usuario puede escribir múltiples reseñas sobre películas o series.

## Tabla historial\_visualizacion

La tabla historial\_visualizacion registra las interacciones de los usuarios con el contenido, permitiendo hacer seguimiento de lo que han visto y en qué momento.

|  |  |  |  |
| --- | --- | --- | --- |
| Campo | Tipo de Dato | Restricciones | Descripción |
| idVisualizacion | uuid | Primary Key | Identificador único de la actividad |
| id\_usuario | uuid | |  | | --- | | FOREIGN KEY -> usuarios(idUsuario) |  |  | | --- | |  | | |  | | --- | | Usuario que realizó la acción |  |  | | --- | |  | |
| |  | | --- | | id\_contenido |  |  | | --- | |  | | varchar(24) | |  | | --- | |  | | |  | | --- | | Contenido interactuado |  |  | | --- | |  | |
| Progreso | varchar (50) | INT CHECK (progreso BETWEEN 0 AND 100) | |  | | --- | |  |  |  | | --- | | Progreso del contenido | |
| fecha\_visto | timestamp | Default Now() | Fecha y hora de la visualización |

## Relaciones de historial\_visualizacion

* Un usuario puede tener múltiples registros en el historial con diferentes contenidos.
* Un contenido puede estar en el historial de múltiples usuarios.

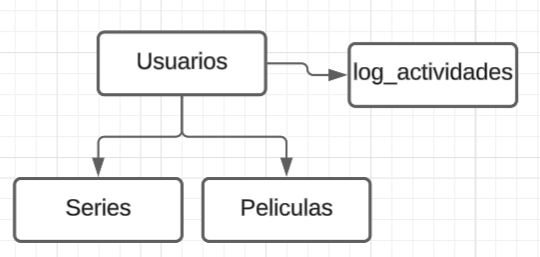
## Tabla reseñas

La tabla reseñas almacena las opiniones y calificaciones que los usuarios dan a las películas y series vistas.

|  |  |  |  |
| --- | --- | --- | --- |
| Campo | Tipo de Dato | Restricciones | Descripción |
| idResena | Serial | Primary Key | Identificador único de la reseña |
| |  | | --- | | id\_usuario |  |  | | --- | |  | | Integer | |  | | --- | | FOREIGN KEY -> usuarios(idUsuario) |  |  | | --- | |  | | |  | | --- | | Usuario que hizo la reseña |  |  | | --- | |  | |
| |  | | --- | | id\_contenido |  |  | | --- | |  | | Integer | |  |  |  | | --- | --- | --- | | |  | | --- | |  |  |  | | --- | | FOREIGN KEY -> contenido(id\_contenido) | |  |  | | --- | |  | | |  |  |  | | --- | --- | --- | | |  | | --- | | Contenido reseñado |  |  | | --- | |  | |  |  | | --- | |  | |
| |  | | --- | | calificacion |  |  | | --- | |  | | Integer | CHECK (calificacion BETWEEN 1 AND 5) | Puntuación del contenido (1-5) |
| |  | | --- | | comentario |  |  | | --- | |  | | Text |  | Opinión del usuario sobre el contenido |
| |  | | --- | | fecha\_resena |  |  | | --- | |  | | timestamp | Default Now() | |  | | --- | |  |  |  | | --- | | Fecha de la reseña | |

## Relaciones de reseñas

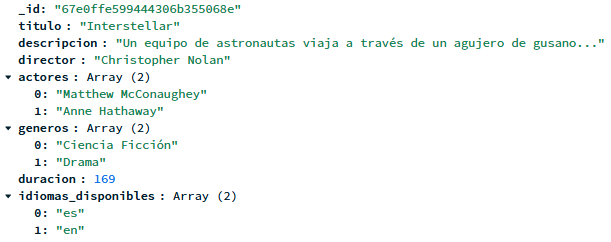
* Un usuario puede escribir múltiples reseñas, pero solo una por contenido.
* Un contenido puede tener múltiples reseñas de distintos usuarios.

[](https://lucid.app/lucidchart/e5a5acc6-41ed-43bb-9613-6460da4393fc/edit?viewport_loc=147%2C813%2C1106%2C489%2C0_0&invitationId=inv_60f48ccb-2f73-4cff-afe8-9c41c49c2dd0)

[Diagrama 2 Modelo de BD](https://lucid.app/lucidchart/e5a5acc6-41ed-43bb-9613-6460da4393fc/edit?viewport_loc=147%2C813%2C1106%2C489%2C0_0&invitationId=inv_60f48ccb-2f73-4cff-afe8-9c41c49c2dd0)

## Colección Peliculas

La colección películas en la base de datos de MongoDB almacena información detallada sobre cada película disponible en el servicio de streaming. Cada documento dentro de esta colección representa una película específica y contiene una variedad de atributos que facilitan la organización, búsqueda y reproducción del contenido.





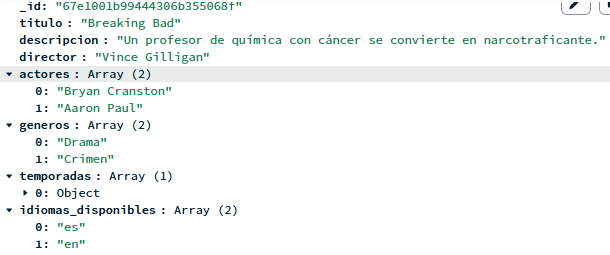
* **\_id (ObjectId)**: Identificador único de la película dentro de la colección. Generado automáticamente por MongoDB.
* **titulo (String)**: Nombre de la película.
* **descripcion (String)**: Breve sinopsis de la película que proporciona contexto sobre su trama.
* **director (String)**: Nombre del director o directores responsables de la película.
* **actores (Array de objetos)**: Lista de actores principales en la película. Cada objeto dentro del array contiene:
  + nombre: Nombre del actor o actriz.
* **generos (Array de Strings)**: Géneros a los que pertenece la película (Ej: Ciencia ficción, Drama, Acción).
* **duracion (Number)**: Duración de la película en minutos.
* **idiomas\_disponibles (Array de Strings)**: Lista de idiomas en los que la película está disponible.
* **subtitulos (Array de Strings)**: Lista de idiomas en los que la película tiene subtítulos disponibles.
* **calificacion (Number)**: Calificación promedio de la película basada en reseñas de los usuarios (valor entre 1 y 5, o 1 y 10 según el sistema de evaluación utilizado).
* **url\_streaming (String)**: Enlace directo a la reproducción de la película dentro de la plataforma de streaming.
* **vistas (Number)**: Cantidad de veces que la película ha sido reproducida en la plataforma.

***Consideraciones sobre el Diseño de la Colección***

1. **Uso de Arrays y Objetos**
   * Se optó por almacenar actores en un array de objetos, permitiendo acceder a información detallada sin necesidad de hacer consultas adicionales.
   * Los idiomas y los subtítulos se manejan en arrays de strings, facilitando búsquedas y filtrado de contenido por idioma.
2. **Eficiencia en Consultas**
   * Se pueden indexar campos como título, géneros y calificación para acelerar las consultas más frecuentes.
   * Se pueden aplicar índices compuestos si se desean realizar búsquedas combinadas, por ejemplo, genero + calificación.
3. **Escalabilidad**
   * Almacenar datos de la película en MongoDB permite manejar cambios en la estructura sin afectar otras entidades.
   * Se pueden agregar campos opcionales en el futuro sin necesidad de modificar documentos existentes.

## Colección Series

La colección series en la base de datos de MongoDB almacena información detallada sobre cada serie de televisión disponible en la plataforma de streaming. Cada documento dentro de esta colección representa una serie específica e incluye atributos que facilitan su organización, búsqueda y reproducción dentro del sistema.





* **\_id (ObjectId)**: Identificador único de la serie dentro de la colección.
* **titulo (String)**: Nombre de la serie.
* **descripcion (String)**: Breve sinopsis de la serie que proporciona contexto sobre su trama.
* **director (String)**: Nombre del director o creadores de la serie.
* **actores (Array de objetos)**: Lista de actores principales en la serie. Cada objeto dentro del array contiene:
  + nombre: Nombre del actor o actriz.
* **generos (Array de Strings)**: Géneros a los que pertenece la serie (Ej: Ciencia ficción, Drama, Acción).
* **temporadas (Array de objetos)**: Cada objeto representa una temporada de la serie e incluye:
  + numero: Número de la temporada.
* **idiomas\_disponibles (Array de Strings)**: Lista de idiomas en los que la serie está disponible.
* **subtitulos (Array de Strings)**: Lista de idiomas en los que la serie tiene subtítulos disponibles.
* **calificacion (Number)**: Calificación promedio de la serie basada en reseñas de los usuarios (valor entre 1 y 5, o 1 y 10 según el sistema de evaluación utilizado).

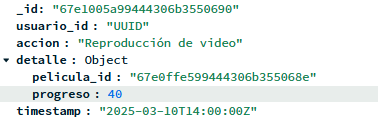
***Consideraciones sobre el Diseño de la Colección***

1. **Estructura Jerárquica con Anidación**
   * Se optó por anidar temporadas dentro del mismo documento de la serie. Esto reduce la necesidad de hacer múltiples consultas para obtener la información completa de una serie.
   * Se evita el uso de múltiples documentos relacionados, lo que simplifica la consulta de datos.
2. **Eficiencia en Consultas**
   * Indexar titulo y generos para mejorar la velocidad de búsqueda de series por nombre o género.
   * Se pueden aplicar índices compuestos si se desean realizar búsquedas combinadas, por ejemplo, genero + calificación.
3. **Escalabilidad**
   * Este diseño permite agregar nuevas temporadas y episodios fácilmente sin necesidad de restructurar la base de datos.
   * Se pueden incluir metadatos adicionales en los episodios, como fechas de estreno o enlaces a plataformas de streaming.

La colección series en MongoDB está diseñada para almacenar información completa sobre cada serie dentro de la plataforma de streaming. Su estructura anidada permite un acceso eficiente a temporadas y episodios sin necesidad de múltiples consultas a diferentes documentos.

## Colección log\_actividades

La colección log\_actividades en la base de datos MongoDB se encarga de almacenar un historial detallado de todas las acciones realizadas por los usuarios dentro de la plataforma de streaming. Este registro permite monitorear la actividad de cada usuario, lo que es útil tanto para la personalización de la experiencia de usuario como para la auditoría de seguridad y la optimización del servicio.



* **\_id (ObjectId)**: Identificador único de cada evento registrado.
* **usuario\_id (ObjectId)**: Identificador del usuario que realizó la acción, enlazado a la colección usuarios en MongoDB.
* **accion (String)**: Tipo de acción realizada por el usuario. Algunos ejemplos incluyen:
  + "Inicio de sesión"
  + "Reproducción iniciada"
  + "Pausa"
  + "Reanudación de reproducción"
  + "Finalización de reproducción"
  + "Reseña publicada"
  + "Búsqueda de contenido"
* **detalle (String)**: Descripción adicional de la acción. Puede incluir detalles sobre la película o serie visualizada, el término de búsqueda usado o cualquier otra información relevante.
* **pelicula\_id (ObjectId, Opcional)**: Identificador de la película o serie relacionada con la acción (si aplica). En caso de acciones generales como el inicio de sesión, este campo puede estar ausente.
* **progreso (Number, Opcional)**: Indica el porcentaje de la película o serie que ha sido reproducido antes de pausar, salir o finalizar la visualización. Su valor varía entre 0.0 y 1.0 (Ej: 0.05 significa que el usuario ha visto el 5% del contenido). Este campo es útil para la funcionalidad de "Continuar viendo".
* **timestamp (ISODate)**: Registro de la fecha y hora exacta en la que ocurrió la acción, almacenado en formato ISO 8601.

***Consideraciones sobre el Diseño de la Colección***

1. **Escalabilidad y Rendimiento**
   * Dado que los registros de actividad pueden crecer de manera exponencial con el número de usuarios y sus interacciones, MongoDB permite manejar estos datos sin comprometer el rendimiento del sistema.
   * Se pueden aplicar políticas de retención de datos para eliminar automáticamente registros más antiguos después de cierto tiempo, reduciendo el almacenamiento innecesario.
2. **Índices para Consultas Eficientes**
   * Se recomienda indexar usuario\_id para mejorar la velocidad de búsqueda de registros de actividad de un usuario en particular.
   * También es útil indexar timestamp para optimizar consultas basadas en rangos de fechas.
3. **Seguridad y Auditoría**
   * La recolección de estos registros no solo permite personalizar la experiencia del usuario, sino que también es fundamental para la auditoría de seguridad.
   * Se puede implementar monitoreo en tiempo real para detectar comportamientos sospechosos, como múltiples intentos fallidos de inicio de sesión en un corto periodo de tiempo.

La colección log\_actividades desempeña un papel crucial en el seguimiento de las interacciones de los usuarios dentro de la plataforma de streaming. Su estructura flexible permite registrar diversas acciones con un nivel de detalle suficiente para mejorar la experiencia del usuario y garantizar la seguridad del sistema.

* usuarios → log\_actividades: Un usuario puede generar muchas actividades registradas en el log.
* peliculas y series están separadas porque tienen estructuras similares, pero diferencias clave (ej., temporadas en series).

Por lo tanto, en nuestras bases de datos **películas y series** se almacenan en MongoDB porque su estructura es flexible y puede cambiar con el tiempo (ej., una serie puede agregar nuevas temporadas), el **log de actividades** también se almacena en MongoDB para manejar grandes volúmenes de eventos sin afectar el rendimiento del sistema y **usuarios, historial de visualización y reseñas** se manejan en PostgreSQL porque requieren integridad referencial.

# Seguridad y Permisos

# Respaldo y Recuperación

 **Backups automáticos** programados diariamente.

 **Registro de cambios** para mantener un historial de modificaciones.