

**Informe del Modelo de Datos – Sistema Integrado de Gestión de Biblioteca (DGAC)**

**Equipo alumnos:**

Cristian Adrian Martinez Alcantara

Renato Toro Silva

Rodolfo Morales

Miguel Angel Donoso

**Docente:**

Fabián Alejandro Alcántara Guajardo

**Sede:**

San Bernardo/Duoc UC

[**1. Contexto General 2**](#_heading=h.vkw4efsmoa74)

[**2. Objetivo del Modelo de Datos 3**](#_heading=h.p1f0pwkd016j)

[**3. Descripción de Tablas Principales 4**](#_heading=h.a6kz85t4q64f)

[3.1. USUARIO: 4](#_heading=h.kqwvksvxj7xl)

[3.2. PERFIL: 4](#_heading=h.czgw85hus4ex)

[3.3. LIBRO: 4](#_heading=h.d4j1vfeesgzf)

[3.4. MATERIAL: 4](#_heading=h.q5kqlq6eotnf)

[3.5. PRÉSTAMO: 4](#_heading=h.elw2i3r9tkuv)

[3.6. FAVORITO: 5](#_heading=h.esmgu8hpcrne)

[3.7. HISTORIAL: 5](#_heading=h.3lwpg1jlxhyj)

[3.8. REPORTES: 5](#_heading=h.jrm5eg9uyae)

[**4. Diagrama del Modelo Entidad–Relación (MER) 6**](#_heading=h.m6ocr2443gzo)

[**5. Relaciones Clave del Modelo 6**](#_heading=h.42yfr08somnx)

[**6. Justificación del Diseño 7**](#_heading=h.xcs5qjjxfq9d)

[**7. Conclusiones y Justificación Tecnológica 8**](#_heading=h.5dxa65uwcadx)

[7.1. Justificación de Django y SQLite para Prototipado Local 9](#_heading=h.2dlto5k90tvk)

[7.2. Escalabilidad (Transición a Producción) 9](#_heading=h.mgjz5pb84hdu)

# 1. Contexto General

La Dirección General de Aeronáutica Civil (DGAC) ha identificado la necesidad crítica de modernizar y automatizar la gestión de su acervo bibliotecario, una tarea que actualmente se realiza de forma manual. Este proceso manual es inherentemente propenso a errores, consume tiempo y dificulta la obtención de métricas de uso de los recursos.

El presente Modelo de Datos es la base estructural propuesta para el Sistema Integrado de Gestión de Biblioteca. Su diseño ha seguido rigurosos criterios de normalización (3FN), buscando asegurar la integridad referencial, minimizar la redundancia y garantizar la escalabilidad futura. Este modelo permitirá al sistema gestionar eficientemente:

* El inventario de materiales bibliográficos (libros, manuales, etc.).
* El registro de usuarios y sus roles.
* Las operaciones transaccionales como préstamos y devoluciones.
* La trazabilidad de todas las acciones importantes dentro del sistema.

# 2. Objetivo del Modelo de Datos

El objetivo principal de este diseño es establecer una estructura relacional robusta y flexible que soporte todas las funcionalidades requeridas para la gestión bibliotecaria de la DGAC:

* Gestión de Usuarios con Roles Diferenciados: Permitir el registro y administración de usuarios con distintos niveles de acceso y permisos, tales como Administrador (gestión total), Cargador (ingreso de material) y Lector (consulta y operaciones de préstamo).
  + *Ejemplo:* Un usuario con el rol 'Administrador' podrá eliminar un registro de libro, mientras que un 'Lector' sólo podrá marcarlo como favorito.
* Control Riguroso de Préstamos y Devoluciones: Implementar un flujo de trabajo que asegure el seguimiento preciso del estado de cada material (disponible, prestado, en mora), incluyendo la gestión de renovaciones.
  + *Ejemplo:* Registrar la fecha\_devolucion y el número de renovaciones para un material específico.
* Administración Diversificada de Colecciones: Distinguir entre diferentes tipos de recursos, permitiendo un manejo de inventario más granular para libros, manuales, revistas o el material aeronáutico especializado.
  + *Ejemplo:* Separar la gestión de un "Manual de Vuelo" (tabla MATERIAL) de la de un "Libro de Historia de la Aviación" (tabla LIBRO).
* Control de Favoritos y Trazabilidad Histórica: Ofrecer funcionalidades de valor añadido al usuario (lista de favoritos) y al administrador (auditoría de acciones).
* Generación de Reportes Estadísticos: Almacenar datos para el análisis de la utilización de la biblioteca, esencial para la toma de decisiones.
  + *Ejemplo:* Un reporte podría indicar los 10 libros más prestados del mes.

# 3. Descripción de Tablas Principales

El modelo se compone de ocho tablas, diseñadas para evitar la duplicidad de datos y mantener la coherencia.

## 3.1. USUARIO:

* **Propósito:** Almacenar los datos fundamentales de las personas que interactúan con el sistema.
* **Atributos Clave:** nombre, correo (clave de acceso), contraseña (cifrada por seguridad), rol (Administrador, Cargador, Lector) y fecha\_creacion.
* **Relaciones:** Se vinculan 1:1 con **PERFIL** y 1:N (Uno a Muchos) con las tablas **PRÉSTAMO**, **FAVORITO** e **HISTORIAL**, ya que un usuario puede realizar múltiples de estas acciones.

## 3.2. PERFIL:

* **Propósito:** Almacenar información opcional o visual, separándose de los datos críticos de autenticación (**USUARIO**).
* **Atributos Clave:** avatar (URL o ruta de la foto de perfil).
* **Relaciones:** Relación **1:1** con **USUARIO** a través de la clave foránea usuario\_id.

## 3.3. LIBRO:

* **Propósito:** Registrar materiales bibliográficos estándar con un formato de libro.
* **Atributos Clave:** título, autor, descripción, publicado (fecha), y cantidad disponible.
* **Relaciones:** Se relaciona 1:N con **PRÉSTAMO** y **FAVORITO**.

## 3.4. MATERIAL:

* **Propósito:** Almacenar recursos bibliográficos distintos a los libros (e.g., manuales aeronáuticos, revistas, mapas, videos).
* **Atributos Específicos:** tipo (e.g., "Revista", "Manual"), ubicación (física o digital), formato (e.g., "PDF", "Impreso") y código de inventario.
* **Relaciones:** Relación **1:N** con **PRÉSTAMO** y **FAVORITO**.

## 3.5. PRÉSTAMO:

* **Propósito:** Gestionar las transacciones de préstamo y devolución.
* **Atributos Clave:** fecha\_prestamo, fecha\_devolucion, renovaciones realizadas y el estado actual (e.g., "Activo", "Devuelto", "Vencido").
* **Relaciones:** Es la tabla transaccional central. Posee claves foráneas a **USUARIO**, **LIBRO** y **MATERIAL**, reflejando la operación (Muchos a 1).

## 3.6. FAVORITO:

* **Propósito:** Permitir a los usuarios marcar recursos para un acceso rápido, mejorando su experiencia.
* **Atributos Clave:** creado (timestamp).
* **Relaciones:** Une al **USUARIO** con un recurso, ya sea un **LIBRO** o un **MATERIAL**, en una relación N:M (implementada como tabla intermedia 1:N desde cada entidad).

## 3.7. HISTORIAL:

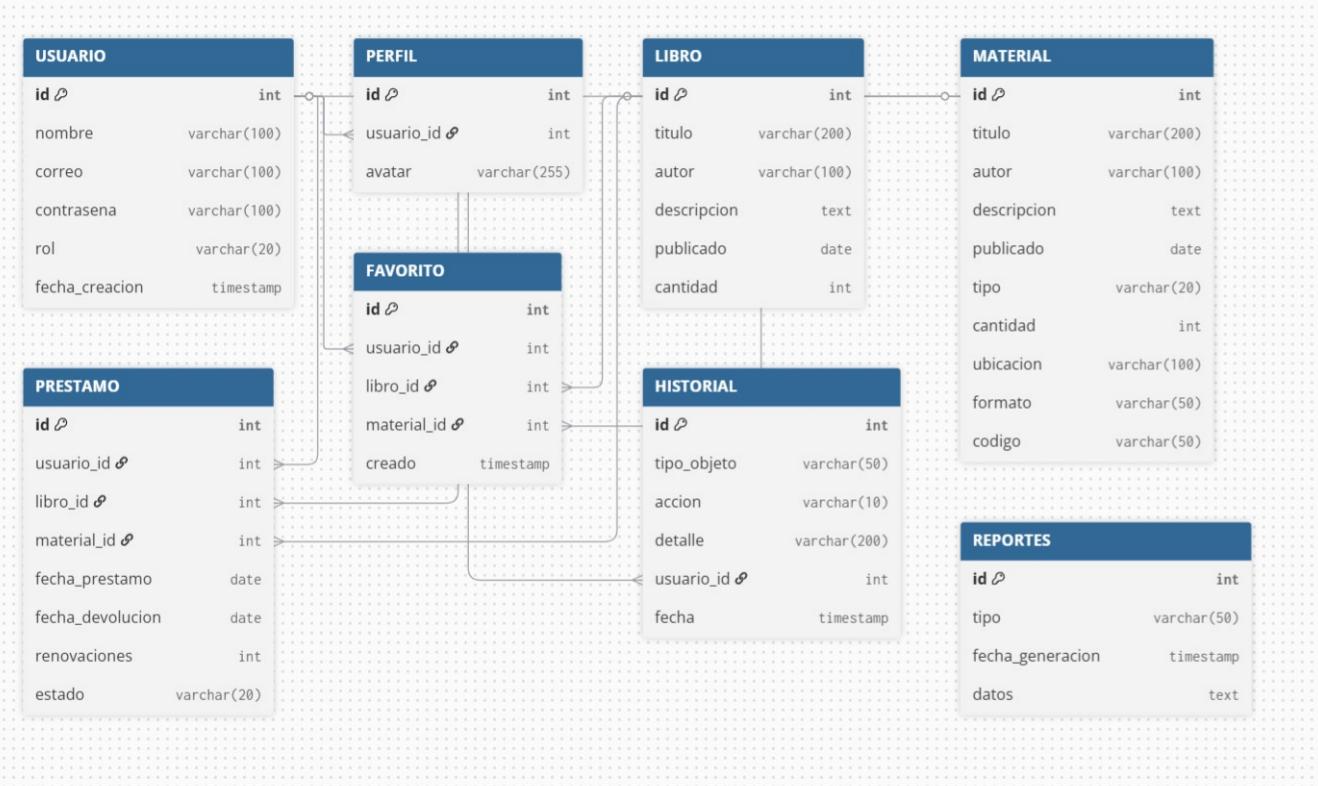
* **Propósito:** Servir como log de auditoría. Registra las acciones significativas del usuario dentro del sistema, proporcionando trazabilidad.
* **Atributos Clave:** tipo\_objeto (tabla afectada), accion (e.g., "CREAR", "MODIFICAR", "ELIMINAR"), detalle (descripción del cambio) y fecha.

## 3.8. REPORTES:

* **Propósito:** Almacenar los resultados de la generación de informes estadísticos, optimizando el rendimiento al evitar generar reportes complejos con frecuencia.

**Atributos Clave:** tipo (e.g., "Top 10 Préstamos"), fecha\_generacion y datos (el contenido del reporte en formato texto/JSON).

# 4. Diagrama del Modelo Entidad–Relación (MER)



En el diagrama MER se observan 8 entidades principales con relaciones 1:1 y 1:N bien definidas. Destacan las entidades transaccionales (PRÉSTAMO e HISTORIAL) y las de soporte (USUARIO y PERFIL), asegurando modularidad y trazabilidad.

# 5. Relaciones Clave del Modelo

| **Relación** | **Tipo** | **Descripción** |
| --- | --- | --- |
| **USUARIO – PERFIL** | **1:1** | Cada usuario registrado tiene un único registro de perfil asociado. |
| **USUARIO – PRÉSTAMO** | **1:N** | Un usuario puede realizar múltiples operaciones de préstamo a lo largo del tiempo. |
| **USUARIO – FAVORITO** | **1:N** | Un usuario puede marcar una colección de recursos como favoritos. |
| **USUARIO – HISTORIAL** | **1:N** | Un usuario genera múltiples registros de acciones auditables en el sistema. |
| **LIBRO – PRÉSTAMO** | **1:N** | Un mismo libro (por su ID) puede ser objeto de múltiples transacciones de préstamo. |
| **MATERIAL – PRÉSTAMO** | **1:N** | Un material no-libro puede ser prestado varias veces. |
| **LIBRO/MATERIAL – FAVORITO** | **1:N** | Un recurso puede ser marcado como favorito por varios usuarios. |

# 6. Justificación del Diseño

El diseño propuesto se sustenta en principios de ingeniería de datos para garantizar un sistema eficiente, mantenible y escalable:

* Tercera Forma Normal (3FN): El modelo ha sido construido bajo la 3FN, que exige que todos los atributos no clave dependen directamente de la clave primaria. Esto se traduce en una eliminación efectiva de la redundancia de datos.
  + *Ejemplo:* Los datos personales del usuario solo se almacenan una vez en USUARIO, y solo su id se repite en PRÉSTAMO, evitando almacenar su nombre y correo en cada transacción.
* Separación Lógica de Entidades (LIBRO y MATERIAL): La decisión de separar los recursos bibliográficos en dos tablas distintas (LIBRO y MATERIAL) optimiza la gestión. Permite que la tabla MATERIAL posea atributos únicos y relevantes (como código o ubicación) que no son necesarios para un LIBRO, facilitando consultas y reportes específicos.
* Trazabilidad y Control (HISTORIAL): La tabla HISTORIAL es crucial para la auditoría y el seguimiento de cambios. Registra quién, qué, y cuándo se realizó una acción, esencial para un sistema institucional donde la integridad de los datos es prioritaria.
* Diseño Modular y Extensible: La estructura relacional y la clara separación de las responsabilidades de cada tabla aseguran la facilidad para ampliaciones futuras.
  + *Ejemplo:* Para añadir un módulo de NOTIFICACIONES (alerta de devolución), solo se requeriría una nueva tabla con una relación 1:N con USUARIO, sin alterar la estructura base.

# 

# 7. Conclusiones y Justificación Tecnológica

El **Modelo de Datos** aquí presentado establece una estructura sólida, segura y escalable que cumple con la totalidad de los requerimientos funcionales para la automatización de la gestión bibliotecaria de la DGAC. Al adherirse a principios de normalización, el modelo garantiza la **integridad de los datos**, esencial para el control de inventario y transacciones de préstamo/devolución.

Además, la elección de **Django** como framework web y **SQLite** como motor de base de datos para la fase de prototipado en ambiente **localhost** es fundamental y estratégicamente justificada:

## 7.1. Justificación de Django y SQLite para Prototipado Local

* **Desarrollo Rápido con Django:** Django, un *framework* escrito en Python, acelera drásticamente el desarrollo de aplicaciones web (**MVPs**) gracias a sus componentes "listos para usar". Proporciona una solución robusta "fuera de la caja" para la **autenticación de usuarios** y el **panel de administración** (ideal para la gestión de libros y materiales).
* **Uso Ideal de SQLite en Localhost:** **SQLite** es la base de datos por defecto de Django, lo que minimiza la configuración inicial. Es un motor **ligero** que guarda toda la información en un **único archivo**, facilitando su uso, portabilidad y la configuración en un entorno de desarrollo **localhost** (sin necesidad de un servidor de base de datos dedicado). Su simplicidad permite a los desarrolladores enfocarse en la lógica del negocio (el modelo bibliotecario).

## 7.2. Escalabilidad (Transición a Producción)

El diseño del modelo y la elección tecnológica garantizan que el prototipo sea **escalable** en caso de pasar a un entorno de **producción** con más usuarios:

* **Migración Sencilla de la Base de Datos:** El **ORM (Mapeador Objeto-Relacional)** de Django desacopla el modelo de datos del motor de base de datos. Esto significa que, si el sistema requiere manejar una alta concurrencia de usuarios (limitación de SQLite), la migración a bases de datos robustas como **PostgreSQL** o **MySQL** se realiza simplemente ajustando el archivo de configuración de Django, sin necesidad de reescribir la lógica de la aplicación. El modelo relacional se mantiene intacto.

En resumen, la combinación de este **Modelo de Datos normalizado** con **Django y SQLite** permite la construcción de un prototipo **funcional, eficiente y de rápida iteración** en el entorno local, sentando una base sólida y flexible para el despliegue a producción en el futuro.