Convierte el ER a modelo relacional (tablas con columnas):

1. **Libro:**

Nombre de Columna:	Tipo de Dato	Descripción
ID (único, Libro)	INT	Clave primaria: Identificador único del libro.
Título	VARCHAR(200)	Título del libro, con un máximo de 200 caracteres.
ISBN	VARCHAR(20)	ISBN único del libro, con un máximo de 20 caracteres.
Año de publicación	INT	Año de publicación.
ID_Autor	INT	Clave foránea a la tabla Autor.

2. **Autor:**

Nombre de Columna:	Tipo de Dato	Descripción
ID (único, Autor)	INT	Clave primaria: Identificador único del autor.
Nombre	VARCHAR(125)	Nombre del autor, con un máximo de 125 caracteres.
Nacionalidad	VARCHAR(50)	Nacionalidad del autor, con un máximo de 50 caracteres.

3. **Préstamo:**

Nombre de Columna:	Tipo de Dato	Descripción
ID (único, Préstamo)	INT	Clave primaria: Identificador único del préstamo.
ID_Libro	INT	Clave foránea a la tabla Libro.
ID_Estudiante	INT	Clave foránea a la tabla Estudiante.

Nombre de Columna:	Tipo de Dato	Descripción
Fecha préstamo	DATE	Fecha del préstamo, para tener referencia de cuando comienza.
Fecha devolución	DATE	Fecha de devolución (puede ser nula).

4. **Estudiante:**

Nombre de Columna:	Tipo de Dato	Descripción
ID (único, Estudiante)	INT	Clave primaria: Identificador único del estudiante.
Nombre	VARCHAR(125)	Nombre del estudiante, máximo 125 caracteres.
Grado	VARCHAR(75)	Grado del estudiante (por ejemplo, "2º Bachillerato"), máximo 75 caracteres.

- **Normaliza hasta Tercera Forma Normal (3FN): Elimina redundancias (e.g., no repetir nombre de autor en tabla de libros).**

Primera Forma Normal (1FN):

La 1FN establece que cada columna en una tabla debe contener solo valores atómicos, es decir, no debe haber múltiples valores en una sola celda. También asegura que no haya filas duplicadas y que cada columna contenga solo un tipo de dato (por ejemplo, no mezclar texto con números). 1FN es importante porque asegura que los datos estén bien estructurados y que no haya redundancia o complejidad innecesaria en las tablas.

Mi modelo cumple con 1FN porque todos los atributos son atómicos, no tienes valores compuestos ni columnas con listas o múltiples valores, y no hay filas duplicadas.

Segunda Forma Normal (2FN):

La **2FN** establece que, además de cumplir con **1FN**, los atributos en una tabla deben depender completamente de la **clave primaria**. En el caso de una clave primaria compuesta (de más de una columna), todos los atributos deben depender de **toda** la clave primaria y no solo de una parte de ella. Si algún atributo depende solo de una parte de la clave primaria, debe moverse a otra tabla. La **2FN** es necesaria para **eliminar dependencias parciales** que pueden generar **redundancias** o relaciones incorrectas dentro de la base de datos, lo que hace que los datos sean más consistentes y eficientes.

Mi modelo cumple con **2FN** porque no tengo claves primarias compuestas, y todos los atributos dependen completamente de la clave primaria de sus respectivas tablas. No existen dependencias parciales en ninguna de las entidades de mi modelo.

Tercera Forma Normal (3FN):

La **3FN** establece que, además de cumplir con **2FN**, no puede haber **dependencias transitivas**. Es decir, si un atributo no clave depende de otro atributo no clave, se debe reestructurar la base de datos para que todos los atributos no clave dependan únicamente de la **clave primaria**. Esto elimina dependencias innecesarias que podrían generar inconsistencias en los datos. La **3FN** es crucial para eliminar relaciones indirectas innecesarias y hacer que la base de datos sea más eficiente, fácil de mantener y consistente.

Mi modelo cumple con **3FN**, ya que, aunque podría entenderse en la parte de relaciones 1 a 1, que existe una tabla llamada detalles, esta no existe, detalles solo son un conjunto de datos asociados a un estudiante como podría ser su nombre y grado, esta tabla en si no existe, solo es mencionada como un detalle que tiene que ir relacionado solo a un estudiante, además aprovechando esta estructura podemos reducir redundancias innecesarias o dependencias de datos secundarios y así tener un código más limpio y sencillo.

Explicación de al menos un ejemplo de desnormalización evitada:

En lugar de almacenar el nombre del autor repetidamente en cada fila de la tabla Libro, lo que podría generar redundancia de datos y hacer el sistema más lento al actualizar los registros, se decidió almacenar el nombre del autor solo en la tabla Autor. Esta decisión fue posible gracias a la 3FN, ya que nos permitió eliminar la repetición del nombre del autor en varias filas de Libro, asegurando que, si un autor cambia su nombre, solo se necesita actualizar una fila en la tabla Autor. Esto mejora la eficiencia, reduce el uso de recursos y asegura que los datos sean más consistentes y fáciles de mantener.

Otro problema que se solucionó fue la gestión de "Detalles". Inicialmente, Detalles se pensaba como una tabla separada para almacenar información adicional de los estudiantes. Sin embargo, aplicando las reglas de normalización, se identificó que la información mínima de los estudiantes, como Nombre y Grado, debía ir directamente en la tabla Estudiante, eliminando la necesidad de una tabla adicional para estos datos. Estos cambios ayudaron a mejorar la optimización del código, ya que ahora los detalles están directamente en la tabla Estudiante, haciendo el proceso más sencillo y puntual. Esto ayuda el control de los datos y hace más sencillo el código.

- ## Explica brevemente por qué cada forma normal resuelve un problema (1FN, 2FN, 3FN).

Primera Forma Normal (1FN): La 1FN resuelve el problema de datos no estructurados. Asegura que todas las columnas en una tabla contengan valores atómicos, es decir, no se permiten listas ni conjuntos de valores en una misma celda. Esto elimina la duplicación de datos y asegura que cada columna tenga solo un tipo de valor, lo que facilita las consultas y la integridad de los datos.

Segunda Forma Normal (2FN): La 2FN resuelve el problema de dependencias parciales. Si una tabla tiene una clave primaria compuesta (más de una columna), algunos atributos pueden depender solo de una parte de la clave, lo que genera redundancias. La 2FN asegura que todos los atributos dependan completamente de la clave primaria, eliminando las dependencias parciales y optimizando la estructura de la base de datos.

Tercera Forma Normal (3FN): La 3FN resuelve el problema de dependencias transitivas. Si un atributo no clave depende de otro atributo no clave, la base de datos se vuelve menos eficiente y más propensa a errores. La 3FN asegura que todos los atributos no clave dependan solo de la clave primaria, eliminando dependencias indirectas y reduciendo la redundancia, lo que hace que la base de datos sea más consistente y fácil de mantener.