# Parte 1B – Normalización (Formato DDL)

A continuación se presentan tres tablas en formato DDL, cada una correspondiente a un nivel específico de normalización y su justificación. La cuarta tabla, como se explica más abajo, no puede existir según la teoría de normalización.

## Tabla 1 – No está en ninguna FN

DDL:

CREATE TABLE LibroConAutorCompuesto (  
 NRO\_LIBRO INT PRIMARY KEY,  
 TITULO VARCHAR(100),  
 AUTOR VARCHAR(255)  
);

Justificación:

No cumple 1FN porque el campo AUTOR puede contener múltiples valores (multivaluados), lo cual viola la atomicidad requerida por la primera forma normal.

## Tabla 2 – En 1FN pero no en 2FN

DDL:

CREATE TABLE Prestamo1FN (  
 NRO\_LECTOR INT,  
 NRO\_LIBRO INT,  
 NOMBRE\_LECTOR VARCHAR(100),  
 PRIMARY KEY (NRO\_LECTOR, NRO\_LIBRO)  
);

Justificación:

Cumple 1FN porque todos los campos contienen valores atómicos. Sin embargo, no cumple 2FN porque el atributo NOMBRE\_LECTOR depende únicamente de NRO\_LECTOR, que es solo parte de la clave primaria compuesta (NRO\_LECTOR, NRO\_LIBRO). Esto representa una dependencia parcial, lo que viola la segunda forma normal.

## Tabla 3 – En 2FN pero no en 3FN

DDL:

CREATE TABLE Lector2FN (  
 NRO\_LECTOR INT PRIMARY KEY,  
 NOMBRE VARCHAR(100),  
 TRABAJO VARCHAR(50),  
 SALARIO\_ESTIMADO MONEY  
);

Justificación:

Cumple 2FN ya que todos los atributos no clave dependen completamente de la clave primaria. Sin embargo, no cumple 3FN debido a la dependencia transitiva: TRABAJO → SALARIO\_ESTIMADO.

## Tabla 4 – En 3FN pero no en 2FN

Justificación:

Una tabla que esté en 3FN pero no en 2FN es teóricamente imposible. La normalización es un proceso jerárquico, por lo que para que una tabla esté en 3FN debe necesariamente haber pasado por 1FN y 2FN. Por lo tanto, no se puede construir una tabla válida que cumpla esta condición sin violar la lógica fundamental de la teoría de normalización.