# Clase Normalización

La normalización es una técnica aplicada a bases de datos relacionales cuyo objetivo es bajar la redundancia del modelo relacional y lograr mayor consistencia.

Enfoque Práctico: En esta materia veremos este punto de vista de la normalización orientado a la solución de problemas de modelos relacionales.

Veamos un ejemplo:

FACTURA (nro, fecha, CUIT, nombre, tel, total)

PK= (nro)

¿Qué problemas tiene?

Es inconsistente, porque por ej: puede haber diferentes CUIT con el mismo nombre.

Es redundante, porque para distintas facturas se repiten los CUIT, nombres etc.

Veamos algunas filas:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| nro | fecha | CUIT | nombre | tel | total |
| 1 | 1/1/2018 | C1 | N1 | T1 | 1000 |
| 2 | 1/1/2018 | C2 | N2 | T2 | 500 |
| 3 | 2/2/2018 | C1 | N7 | T1 | 800 |

La inconsistencia está dada en C1 con N1 y C1 con N7.

La redundancia está dada porque tanto nombre, tel y CUIT se repiten en todas las filas.

Esta tabla no garantiza ni consistencia ni baja redundancia.

Conociendo el dominio del problema se pueden deducir las siguientes dependencias funcionales.

Dependencia funcional: Es la relación de dependencia entre campos. Se produce cuando conociendo un campo puedo deducir otros. Notación: A->B, C

Quiere decir que f(x)=y,

CUIT-> nombre, tel

En este caso, f(CUIT)=nombre y f(CUIT)=tel

Estas relaciones no se pueden inferir de los datos, sino del conocimiento del dominio del problema.

Así, se reformula el modelo relacional para que sea consistente, teniendo en cuenta el modelo relacional + dependencias funcionales. Esta reformulación nos da como resultado un nuevo modelo relacional que se encuentra en algún grado de normalización.

Luego, se pueden aplicar ciertos algoritmos para saber cuán normalizado está.

Estos algoritmos no se verán teóricamente sino su aplicación y verificación, volviendo al modelo conceptual o DER y evitando que se den estas situaciones de inconsistencia y redundancia.

Veamos el paso a paso:

1) Dado el modelo relacional MR y las dependencias funcionales. Se clarifican los mismos.

2) Para el ejemplo que teníamos:

FACTURA (nro, fecha, CUIT, nombre, tel, total)

CUIT-> nombre, tel

Se identifica que es necesaria una tabla CLIENTE (cuit, nombre, tel) y la tabla FACTURA nos queda:

FACTURA (nro, fecha, CUIT, total)

FK={CUIT->CLIENTE, cuit}

Notar que ya no hay inconsistencia para nombre y teléfono y que nos ahorramos el espacio que los datos de cliente ocupaban en la tabla factura.

3) Generalizando el ejemplo sería: extraer de la tabla los campos involucrados en las dependencias funcionales y colocarlos en una nueva tabla colocando su clave primaria como foránea en la tabla original.

4) Se analiza el DER original y se lo modifica para llegar a un DER que respete el nuevo MR que resultó del paso 3).

Existen distintos grados de normalización como 1FN, 2FN, 3FN, etc. Que determinan cuánta inconsistencia tiene el modelo relacional, tan es así que la 1FN es la menos restrictiva y por lo tanto es menos normalizado. Existen, asimismo, diferentes algoritmos para pasar de una forma normal a otra más estricta. Al ir normalizando se agrega un grado de indirección y por lo tanto más complejidad al modelo.

5) Se pregunta: ¿hay más dependencias inducidas por la clave primaria? Si es así, sigo analizando, sino paro.

Ahora veremos otros casos en donde también deberemos analizar y modificar el modelo para reducir la redundancia y minimizar las inconsistencias.

CASO 1: Redundancia en la clave primaria

Ej:

LOCALIDAD (nombre, cant\_habitantes)

PK= (nombre)

CLIENTE (CUIT, nombre, tel, nom\_loc)

PK= (CUIT)

FK= {nom\_loc ->LOCALIDAD, nombre}

¿Cuál es el problema?

Hay redundancia ya que al ser la clave de LOCALIDAD nombre alfanumérica, se ocupa espacio en CLIENTE que si se eligiera un código numérico. Para solucionar esto lo que se hace es elegir una clave numérica llamada clave subrogada.

LOCALIDAD (cod\_loc, nombre, cant\_habitantes)

PK= (cod\_loc)

CLIENTE (CUIT, nombre, tel, cod\_loc)

PK= (CUIT)

FK= {cod\_loc ->LOCALIDAD, cod\_loc}

Indexar algo alfanumérico ocupa más espacio y procesamiento que algo numérico.

CASO 2: Inconsistencia de nombres

Observando el modelo relacional resultante, se puede ver que podría suceder que para cod\_loc diferentes tengan el mismo nombre. En la tabla se soluciona con una restricción UNIQUE sobre nombre.

CASO 3: Claves compuestas

CLIENTE (tipo, num, nombre)

PK= (tipo, num)

La clave es compuesta, entonces cualquier tabla que la referencie, por ejemplo, FACTURA, tendrá una clave foránea compuesta. Esto es más restrictivo que si intervenimos el modelo relacional creando una clave subrogada id\_cliente y dejando tipo y num como campos comunes.

Veamos otro ejemplo:

Entidad PELICULA con un atributo género. El MR sería:

PELICULA (id, genero)

Vemos que para bajar la redundancia debemos crear otra tabla GENERO (cod, descrip), así que PELICULA (id, cod) donde cod es clave foránea a GENERO.

Haciendo ingeniería inversa vemos que el DER correspondiente podria ser:

Entidad PELICULA y entidad GENERO cuyos atributos con cod y descrip; y con la relación entre ellas "tiene" de N:1.