# Educador Miguel Octaviano Jaramillo Puentes

# NORMALIZACIÓN

Los datos se almacenan en registros que a su vez forman tablas. Para determinar cómo están conformados los registros de cada tabla es necesario hacer un estudio de los datos y cómo podemos agruparlos en diferentes tablas para que su almacenamiento sea óptimo, este procedimiento se denomina NORMALIZACIÓN.

La normalización se puede definir como el examen a los almacenamientos de datos y a su reagrupación de tal forma que permitan de una manera eficiente el diseño y administración del sistema de información. Con este proceso se obtiene menor redundancia en los datos (evitar que un mismo dato se almacena varias veces innecesariamente) y una mayor integridad de los mismos (Que un dato en particular no contenga diferentes valores para un mismo objeto). Para lograr una estructura normalizada (en tercera forma normal) se debe cumplir el siguiente proceso:

Para normalizar una estructura de almacenamiento, lo primero que se debe hacer es definir cuál es el objeto a almacenar (recibo, factura, remisión, cita médica, historia clínica, etc) y definir su nombre, luego identificar que datos (campos) se deben almacenar de este objeto.

Por ejemplo, si se quisiera almacenar facturas de venta, estaríamos definiendo inicialmente el objeto factura de la siguiente forma:

FACTURA = { Número de la factura + Fecha + Código del cliente + Descripción del cliente + Dirección del cliente + Forma de pago + Fecha de vencimiento + {Código del producto + Descripción + Precio unitario + Unidades vendidas + Total por producto} + Bruto a pagar + Descuento + IVA + Neto a pagar}

Luego se procede a aplicar los siguientes pasos:

## PRIMERA FORMA NORMAL

Se identifica la clave primaria y se define en forma única. Esta clave primaria consta de uno o más datos que identifican unívocamente la ocurrencia, esta clave se identifica subrayándola. Esto quiere decir que no debe existir ninguna otra ocurrencia (registro, tupla o relación) con la misma clave o identificación. Como se puede observar en el ejemplo, el número de la factura siempre y cuando se lleven los respectivos controles, será la clave primaria por ser único.

Adicionalmente se remueven todos los grupos repetitivos de campos (o sea: todos los campos que puedan tomar varios valores para el mismo objeto de almacenamiento) y probablemente se conforman con ellos otras entidades o almacenamientos. En el ejemplo, vemos que los campos: {Código del producto + Descripción + Precio unitario + Unidades + Total por producto} para una factura pueden tomar varios datos diferentes

Por lo tanto, el almacenamiento original quedaría de la siguiente forma:

FACTURA = { Número de la factura + Fecha + Código del cliente + Descripción del cliente + Dirección del cliente + Forma de pago + Fecha de vencimiento + Bruto a pagar + Descuento + IVA + Neto a pagar}

LINEA FACTURA = { Número de la factura + Código del producto + Descripción + Precio unitario + Unidades + Total por producto}

***NOTA****: Obsérvese que los nuevos almacenamientos que se generen al aplicar la primera forma normal, también deben quedar en primera forma normal.*

***Cuando se crean nuevas tablas al aplicar la primera forma normal, a estas nuevas tablas se les debe adicional el campo clave de la tabla origen, y por lo regular (no siempre) en la nueva tabla, también será clave junto con otro u otros campos de la nueva tabla. En este ejemplo es el número de la factura.***

## SEGUNDA FORMA NORMAL

Luego de haber obtenido la primera forma normal se logra la segunda forma normal cuando todos los campos dependen funcionalmente (son determinados) de la clave primaria, esto es que todos los campos sean una propiedad directa del objeto de almacenamiento, los que no cumplan esta condición se deben remover y probablemente con ellos formar nuevas tablas.

En el ejemplo, la descripción del cliente y su dirección no dependen de la factura sino del cliente de la misma, mientras el código del cliente puede ser importante en este almacenamiento para reconocer el responsable de la factura, con estos campos se crea un nuevo almacenamiento. Se puede afirmar algo similar de la descripción del producto y su precio cuando es único. Las estructuras quedarían así:

FACTURA = { Número de la factura + Fecha + Código del cliente +Forma de pago + Fecha de vencimiento + Bruto a pagar + Descuento + IVA + Neto a pagar}

LINEA FACTURA = { Número de la factura + Código del producto + Unidades +Total por producto}

CLIENTE = {Código del cliente +Descripción del cliente + Dirección del cliente }

PRODUCTO = { Código del producto + Descripción del producto + Precio unitario}

***Nótese que en el almacenamiento original se deja el campo que será clave en el nuevo almacenamiento (código cliente en el almacenamiento FACTURA y código del producto en el almacenamiento LINEA FACTURA)***

## TERCERA FORMA NORMAL

Una vez lograda la segunda forma normal se obtiene la tercera eliminando toda dependencia transitiva de los campos respecto de la clave primaria. Esto es, todo campo que pueda ser calculado a partir de otros datos almacenados en la base de datos se debe eliminar para evitar problemas de integridad. En el ejemplo, el Bruto a pagar(es la suma de los totales por producto), el IVA (suponiendo que todos los productos tienen el mismo porcentaje) y el neto a pagar del almacenamiento FACTURAS, así como el total por producto (Precio Unitario X Unidades) de LINEA FACTURA, pueden ser calculados y por lo tanto removidos del almacenamiento así:

FACTURA = { Número de la factura + Fecha + Código del cliente +Forma de pago + Fecha de vencimiento + Descuento }

LINEA FACTURA = { Número de la factura + Código del producto + Unidades}

CLIENTE = {Código del cliente +Descripción del cliente + Dirección del cliente }

PRODUCTO = { Código del producto + Descripción del producto + Precio unitario}

Es importante anotar que un buen funcionamiento de un sistema depende fundamentalmente de la estructura de datos, más que de cualquier otro factor.

# MODELO RELACIONAL

Una base de datos relacional permite la utilización simultánea de datos procedentes de más de una tabla.

Al hacer uso de las relaciones, se evita la duplicidad de datos, ahorrando memoria y espacio en el disco, aumentando la velocidad de ejecución y facilitando al usuario el trabajo con tablas.

Para conseguir una correcta base de datos relacional es imprescindible realizar un estudio previo del diseño de la base de datos.

Para poder relacionar tablas entre sí, se deberá especificar un campo en común que contenga el mismo valor en las dos tablas y dicho campo será clave principal en una de ellas. Las tablas se relacionan de dos a dos, donde una de ellas será la tabla principal de la que parte la relación y la otra será la tabla secundaria o destino de la relación.

## Tipos de Relaciones

Existen varios tipos de relaciones entre las tablas, la dos más comunes son:

### Relación Uno a Uno

Cuando un registro de una tabla sólo puede estar relacionado con un único registro de la otra tabla y viceversa.

**A**

**B**

**1**

**1**

Por cada registro en la tabla A existe uno y solo uno en la tabla B

**Por ejemplo:** Tenemos dos tablas una con los datos de Ciudades y otra con una lista de Alcaldes, una Ciudad sólo puede tener un alcalde, y un alcalde lo será únicamente de una Ciudad.

### Relación Uno a Varios

Cuando un registro de una tabla (tabla principal) puede tener más de un registro relacionado en otra tabla (tabla secundaria).

**A**

**B**

**1**

**∞**

Por cada registro en la tabla A puede existir uno o varios en la tabla B