

Normalizar una Base de Datos

..y ¿desnormalizarla?

Empiezo dando por sentadas las nociones de **Primary Key (PK)**, **Foreign Key (FK)**, **relaciones** entre tablas y **cardinalidad**. <dejo link al final>

Antes de ver CÓMO normalizar una BD, primero, saber **PARA QUÉ.** 🤔

Una BD normalizada de manera correcta brinda:

- Prevención de redundancia (data repetida)
- Ahorro de tiempo y \$\$: OPTIMIZACIÓN en espacio de almacenamiento, recursos de procesamiento y costos de mantención.
- Estructura, orden y lógica clara
- **Consistencia** de la información almacenada (facilidad <y seguridad> para realizar 'updateos'/actualización de información)
- Optimización de estructuras para sencillez y eficiencia en el guardado de datos (ver *OLTP/OLAP-Warehouse*)
- Facilidad de acceso a los datos
- Simpleza para detectar relaciones entre entidades u objetos del 'mundo real' que estamos almacenando
- Omnicanalidad: evitamos problemas de sincronización al acceder desde varios canales <usuarios> distintos
- Posibilidad de agregar nuevas columnas sin dañar el esquema actual ni las relaciones entre entidades
- Y por último, **CONFIANZA**, que tiene mucho que ver con el 4to punto, al poder ser modificada y actualizada sin generar datos desactualizados/erróneos.

Partimos de un **.xlsx**

TABLA_ORIGINAL

<u>Factura</u>	<u>Fecha</u>	<u>Cliente</u>	<u>Nombre Cliente</u>	<u>Ciudad</u>	<u>Email</u>	<u>Código Artículo</u>	<u>Nombre Artículo</u>	<u>Cantidad</u>	<u>Precio Unitario</u>
1703	2022-09-24	117	Felipe Darras	Esperanza	fd@gm.com	005	Alambre	3	70
1703	2022-09-24	117	Felipe Darras	Esperanza	fd@gm.com	009	Teflón	2	5
1703	2022-09-24	117	Felipe Darras	Esperanza	fd@gm.com	011	Tornillo	25	2
1712	2022-09-27	120	Virola Perro	Esperanza	vir@gm.com	005	Alambre	2	70
1734	2022-09-27	122	Homero Manzi	Santa Fe	hh@gm.com	009	Teflón	4	5

1734	2022-09-27	122	Homero Manzi	Santa Fe	hh@gm.com	002	Clavo	15	2
------	------------	-----	--------------	----------	-----------	-----	-------	----	---

Resalté con distintos colores para que sea más evidente la redundancia/repetición de datos.

Acá se hace mucho más fácil de ver no solo el ahorro de tiempo (**tiempo = \$\$**) para la mantención de la BD, sino que la facilidad para actualizar <updatear> algún dato:

pensemos que Homero cambia de mail 😞 deberíamos cambiar en cada registro, uno por uno, el mail de Homero. IMPOSIBLE.

También se hace muy evidente, al ser así, la posible falta de consistencia de nuestros datos: si actualizamos el mail de Homero uno por uno, y por error no actualizamos uno de los

registros, Homero pasaría a **¿tener dos mails?** DATA INCONSISTENTE Y NO CONFIABLE!

NORMALIZAMOS 💪

Para normalizar, seguimos las **formas normales** que se proponen. Son varias, por lo menos 5 (cinco), pero por lo general se busca llegar a la tercer forma normal (3FN).

Primer Forma Normal (1FN) : Se tienen que eliminar los datos/registros repetidos colocándolos en tablas separadas (revisar concepto de **ENTIDAD**<representación de un objeto o concepto del mundo real>), se debe definir una llave primaria<PK>, no se pueden duplicar columnas y se debe respetar la atomicidad de los datos (atomicidad = átomo = irreducible > debe contener el elemento de datos más pequeño posible que permita la clasificación y búsqueda fácil. Por ejemplo, la columna de fecha se puede reducir en día, mes y año).

En este caso, la atomicidad solo la apliqué al nombre (para hacer evidente el concepto), separando Nombre y Apellido en campos distintos.

Los campos Nombre, Ciudad, Email, se repetían. Empiezo por separar los datos de las FACTURAS, con los datos de los Clientes redundantes, en una única tabla, separando los detalles de las facturas en otra. Se hace evidente como se dejan de repetir los Emails, Ciudades.....

FACTURAS

Factura (PK)	Fecha	ClienteID	NombreCliente	ApellidoCliente	Ciudad	Email
1703	2022-09-24	117	Felipe	Darras	Esperanza	fd@gm.com
1712	2022-09-27	120	Vírola	Perro	Esperanza	vir@gm.com
1734	2022-09-27	122	Homero	Manzi	Santa Fe	hh@gm.com

DETALLE_FACTURAS

FacturaID (PK)	Factura (FK)	CódigoArtículo	NombreArticulo	Cantidad	PrecioUnitario
1	1703	005	Alambre	3	70
2	1703	009	Teflón	2	5
3	1703	011	Tornillo	25	2
4	1712	005	Alambre	2	70
5	1734	009	Teflón	4	5
6	1734	002	Clavo	15	2

Segunda Forma Normal (2FN): Se tienen que eliminar todas las **dependencias parciales** y separar dentro de sus propias tablas. Hay dependencia parcial cuando hay registros que NO dependen de la llave primaria. Entonces: la 2FN asegura que todas las columnas <campos> que no sean llave, sean completamente DEPENDIENTES de la **PK**, la cual nos deja identificar registros de manera única y unívoca.

Viendo la 2da Forma Normal, vemos rápido que en el estadio anterior (primer paso de normalización), hay dependencias parciales que tenemos que evitar para cumplirla: en mi tabla **DETALLE_FACTURAS** para poner un ejemplo, vemos que el Artículo TEFLÓN, no depende completamente de la llave primaria. ¿Depende del CódigoArtículo? ¿o de mi código de factura?

¡Mi identificador único <PK>, **NO** me está identificando de manera única a los artículos!!



¿Qué pasaría si decido cambiar el nombre del artículo? ¿Lo tengo que actualizar las cientos o miles de veces que está registrado? 🤔🤔

Concretar la 2FN:

FACTURAS

Factura (PK)	Fecha	ClienteID	NombreCliente	ApellidoCliente	Ciudad	Email
1703	2022-09-24	117	Felipe	Darras	Esperanza	fd@gm.com
1712	2022-09-27	120	Virola	Perro	Esperanza	vir@gm.com
1734	2022-09-27	122	Homero	Manzi	Santa Fe	hh@gm.com

DETALLE_FACTURAS

FacturaID (PK)	Factura (FK)	Código Artículo (FK)	Cantidad	Precio Unitario
1	1703	005	3	70
2	1703	009	2	5
3	1703	011	25	2
4	1712	005	2	70
5	1734	009	4	5
6	1734	002	15	2

ARTÍCULOS

ArtículoID (PK)	Nombre Artículo	Precio Unitario
002	Clavo	2
005	Alambre	70
009	Teflón	5
011	Tornillo	2

Parece ir tomando forma, el artículo Teflón ya depende completamente de su identificador, y si quisiera cambiar el nombre de alguno, lo haría solo una vez.

La integridad se conserva, puedo recuperar el nombre del artículo vendido uniendo <JOIN> las tablas donde **ArtículoID** = **CódigoArtículo**

Otro tema a tener en cuenta es el **Precio**. ¿Estoy repitiendo los datos? 🤔 **NO**.

Si en algún momento queremos saber el total facturado en algún mes del año anterior, podríamos calcular la **SUM(Cantidad * Precio)** y distintos son los precios de lista actuales, con los precios 'históricos' <los precios de los artículos al momento de ser vendidos ese mes, de ese año anterior>.



Otro ejemplo sencillo: si tuviese el campo **Descuento**, que llegaría a la 2FN en la tabla **DETALLE_FACTURAS**, deberíamos separarlo para que haga parte de la tabla/entidad **ARTICULOS**, al ser un **atributo** de ellos, y cumplir con la Segunda Forma Normal.

Tercer Forma Normal (3FN): Logramos la 3er Forma Normal cuando todas las columnas que NO son llave son **funcionalmente dependientes** de la **PK** y ya no quedan dependencias transitivas.

¿Dependencias transitivas? 🤖 Existe dependencia transitiva cuando hay campos que NO son llave que dependen de otras columnas que TAMPOCO son llave. Así, se previenen

errores de lógica cuando se insertan o borran registros y tenemos un esquema 'mantenible', limpio y **fácil de trabajar y expandir!**

Se llega al punto donde cada registro, de cada columna, está **identificado** por la **PK** de manera **única**.

Para ver claro el tema de las dependencias, pregunto: En mi tabla **FACTURAS**, el Email ¿depende de la **PK**? Claramente ahí hay una dependencia transitiva que NO tiene que existir para lograr 3FN,  **el Email dependiendo del Cliente**, que NO es **PK**. 

Concretar la 3FN:

CLIENTES

ClienteID	NombreCliente	ApellidoCliente	Ciudad	Email
117	Felipe	Darras	Esperanza	fd@gm.com
120	Virola	Perro	Esperanza	vir@gm.com
122	Homero	Manzi	Santa Fe	hh@gm.com

FACTURAS

Factura (PK)	Fecha	ClienteID (FK)
1712	2022-09-27	120
1703	2022-09-24	117
1734	2022-09-27	122

DETALLE_FACTURAS

FacturalID (PK)	Factura (FK)	Código Artículo (FK)	Cantidad	Precio Unitario
1	1703	005	3	70
2	1703	009	2	5
3	1703	011	25	2
4	1712	005	2	70
5	1734	009	4	5
6	1734	002	15	2

ARTÍCULOS

ArticuloID (PK)	Nombre Articulo	Precio Unitario
002	Clavo	2
005	Alambre	70
009	Teflón	5
011	Tornillo	2

Si quisiera, en mi BD <base de datos>, tener información detallada sobre la Ciudad de los clientes, podría separar la información en una nueva **entidad** "CIUDADES", dejando una FK en mi tabla CLIENTES que la referencie, donde tener normalizada información sobre el departamento al cual pertenezca esa ciudad, la provincia, los habitantes, etc.

La normalización viene a cuento cuando necesitamos definir **estructuras y sistemas** que están pensados para **funcionar** de determinadas maneras, con **propósitos** determinados: una BD pensada para registrar las ventas realizadas en distintas sucursales de una empresa de electrodomésticos, necesita de todos los beneficios de la normalización (entre otras cosas, eficiencia y rapidez para insertar nuevos registros) → **ESTRUCTURA OLTP**, OnLine **Transaction** Processing, pensada como queda claro en el nombre, para TRANSACCIONES. Hablando en lenguaje SQL, en esta estructura se priorizan los **INSERT's**, **UPDATE's** y **DELETE's**.

En cambio, estructura OLAP <Analytical Processing> prioriza la eficiencia de consultas (**SELECT's**). Es acá donde entra el concepto de **desnormalización**. Para evitar, a la hora de consultar una BD, ejecutar muchos JOIN's, se busca segregar los datos en la menor cantidad de tablas posibles. Por ejemplo, el género, sabiendo que son campos que van a recibir solo **F o M** como valores, y es una tabla que no va a crecer en atributos, no tiene sentido "normalizar" y generar una nueva tabla con el género de clientes o empleados.

Felipe Darras

Concepto **cardinalidad**: [cardinalidad modelo entidad relación - Informático Sin Límites \(informaticosinlimites.com\)](http://informaticosinlimites.com)