**UNIDAD 1: Profundizando en Base de Datos Relacionales**

1. [UNIDAD 1: Introducción](https://learn.nextu.com/mod/page/view.php?id=10246&pid=P_WEB_DATABASE)

**Introducción**

**Profundizando en Base de Datos Relacionales**

Cuando la complejidad de las soluciones de base de datos se incrementa, se hace necesario manejar otros conceptos relacionados a las mismas, que permitan ampliar la forma en que operamos sobre ellas y realizar un manejo más eficiente de nuestros datos y su almacenamiento. En esta unidad comenzamos definiendo ciertos conceptos avanzados de las bases de datos relacionales, como lo son las restricciones e índices, los procedimientos y triggers, para luego explicar los procesos de normalización de la base de datos y así terminar con la especificación de operaciones más complejas que podemos realizar sobre nuestras tablas. Para lograr los objetivos planteados seguimos un enfoque teórico-práctico, que busca que el participante pueda practicar los conceptos explicados en videos y lecturas a una diversa cantidad de escenarios y contextos.

**Objetivos de aprendizaje**

1. Aplicar los procesos de normalización a una base de datos de mediana complejidad.
2. Crear claves primarias y foráneas, y sus restricciones como parte de una solución de base de datos.
3. Usar las sentencias SQL GROUP BY, HAVING y JOIN con la finalidad de combinar el contenido de las tablas en diversas maneras.
4. Lección 1: Conceptos Avanzados en Base de Datos Relacionales
   1. [Restricciones e Índices](https://learn.nextu.com/mod/lesson/view.php?id=10247&pid=P_WEB_DATABASE)

Cuando el volumen crece.

Restricciones cuando declaramos una Primary Key:

* - Unicidad.
* - Not Null
* - Clave primaria debe estar en todas las tablas.
* - Clave primaria compuesta esta conformada por más de un campo
* - usualmente las claves compuestas se utilizan en relaciones N a M.
* - Otras situaciones donde se pueden utilizar claves primarias compuestas:

 id del estado y id de la ciudad quienes representan una ciudad.

Una clave foránea apunta a una clave primaria de otra tabla, convirtiéndose en una referencia. En la tabla se debe indicar que es una clave foránea.

Reglas de integridad:

- Que se garantice que las referencias sean válidas.

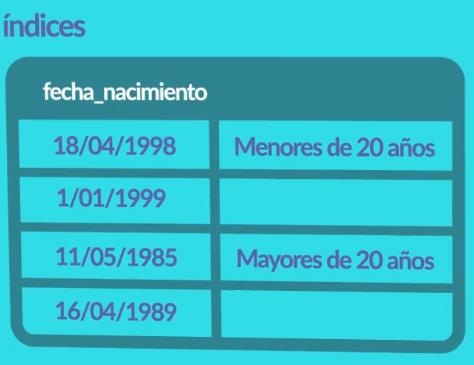
- Not null

Índices, cuando las tablas se vuelven muy grandes, listas enlazadas o árboles de búsqueda para encontrar los registros más fácilmente.

Por ejemplo:



Se podría crear un índice para las fechas de nacimiento.



El índice puede tener uno o más campos. Algunos índices se pueden definir con las restricciones

Los índices permiten un acceso más rápido a las consultas, sin embargo son más lentos para la inserción o eliminación.

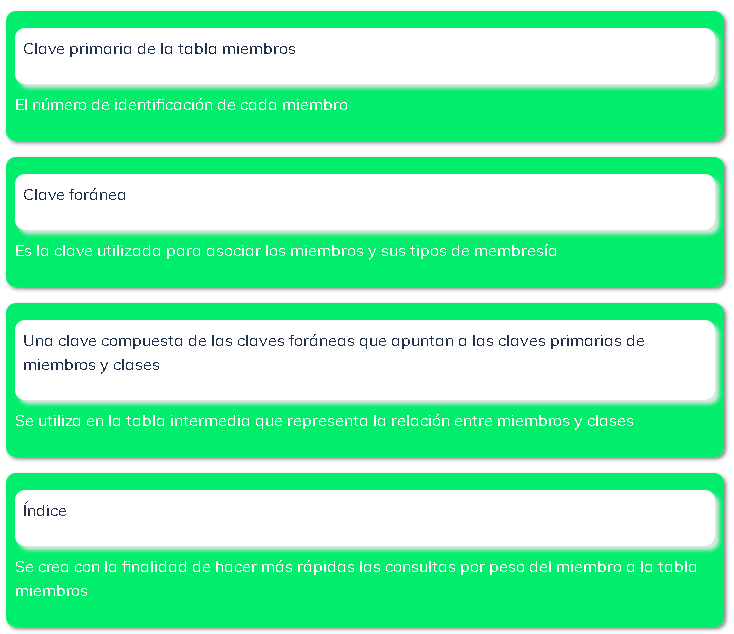
Debido a que cuando se inserta un registro hay que indexar el nuevo valor, y cuando se elimina hay que reindexar, de forma tal que ya no aparezca más en las restricciones definidas.

Se agregan índices en los campos de búsqueda más comunes.

* 1. Actividad Interactiva 1

**Analiza y relaciona**

Supongamos que estamos desarrollando un sistema para el control de los miembros de un gimnasio. Hasta ahora hemos identificado las siguientes tablas: miembros, entrenadores, clases, tipos de membresía (platino, oro, plata, bronce). Además sabemos que la relación entre la tabla miembros y clases es de N a M o de muchos a muchos, ya que muchos miembros pueden asistir a muchas clases y las clases pueden tener muchos miembros. El líder del proyecto requiere ayuda para organizar las ideas sobre los elementos básicos de estas tablas. Para ello es necesario que relaciones las siguientes definiciones.



* 1. Triggers y Procedures

Un trigger es un procedimiento que se ejecuta cuando se da una condición, cuando se declara el trigger también se debe definir la condición que conlleva a que se ejecute el trigger.

Los triggers se usan para realizar auditorías, reportes, tablas, tablas resúmenes, administración de los datos, entre otros.

Ventajas:

Los triggers son usados internamente. Proporcionando así un acceso rápido y hacen el trabajo de forma automática.

Desventajas:

Están escritos en un lenguaje que suele ser distinto al del programador. Además de que son difíciles de mantener y actualizar en ambientes de producción.

Procedures o Stored Procedures – son programas almacenados que pueden ser ejecutados en cualquier momento siempre que se necesiten



* 1. Actividad Interactiva 2

**Analiza y relaciona**

A continuación, se encuentran las definiciones asociadas con Triggers y Procedures, relaciona el término de la columna de la izquierda con su correspondiente de la columna derecha:



* 1. Ejercicio Práctico

**Desafío práctico**

Una compañía desea desarrollar una aplicación Web con una Base de Datos Relacional que tiene las siguientes tablas actores, películas y géneros.

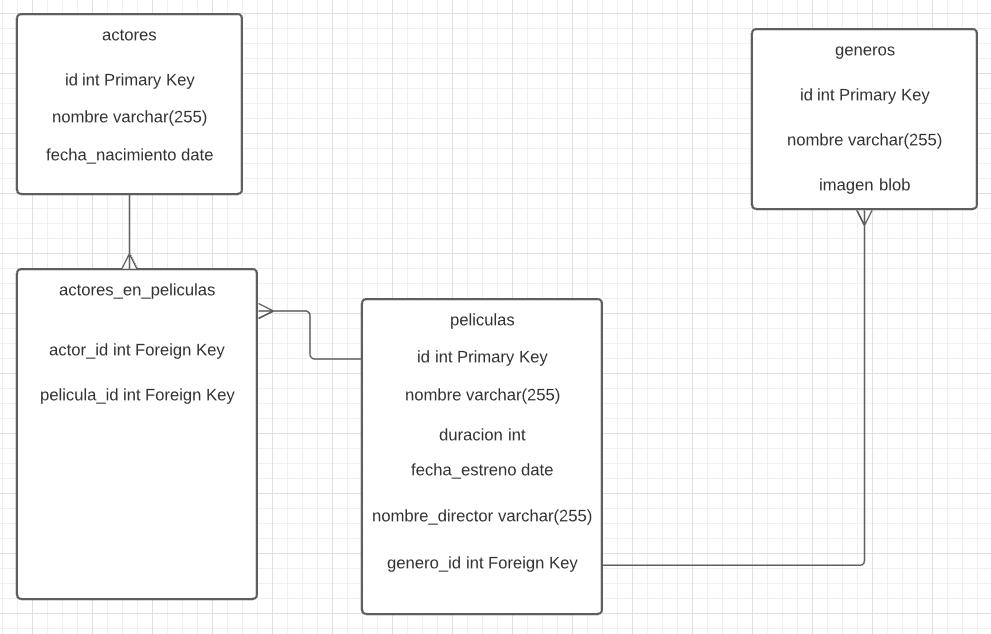
* Del actor se guarda el nombre y la fecha de nacimiento.
* De la película se guarda el nombre, la duración, la fecha de estreno y el nombre del director (representado por un tipo string). Además cada película pertenece a un género.
* De cada género se guarda el nombre y una imagen representativa del mismo.

Adicionalmente, se tendrá un buscador que permite:

* Buscar películas por año, por nombre del director y por duración, y usando dos de estos criterios al mismo tiempo.
* Buscar actores por edad.

**Te encomendamos el reto de diseñar esta base de datos, para lo cual queremos que:**

1. Crees el diagrama entidad/relación



1. Listes las claves primarias

Tabla actores: id Primary Key

Tabla películas: id Primary Key

Tabla géneros: id Primary Key

1. Listes las claves foráneas

Tabla peliculas: genero\_id Foreign Key

Tabla intermedia actores\_en\_peliculas: actor\_id Foreign Key y película\_id Foreign Key

1. Listes todos los índices que se deben crear usando las buenas prácticas descritas

Tabla películas:

* [fecha\_estreno] index,
* [director] index,
* [duración] index,
* [fecha\_estreno, director] index,
* [fecha\_estreno, duracion] index,
* [director, duracion] index

Tabla actores:

- [fecha\_nacimiento] index

2.6. ¿Sabías que?

**Cápsula de conocimiento**

**Pasos para definir una base de datos consistente**

La construcción de una base de datos se puede ver como un proceso sistemático que está compuesto de una serie de pasos bien definidos, los cuales vamos a describir a continuación.



Definir entidades y sus atributos: Luego de identificar los datos en nuestro problema debemos definir las entidades que agrupan esos datos, de forma tal que tengan sentido en nuestro problema. Ejemplo de entidades son estudiantes en un sistema de control de estudios y miembros en un sistema para el control de miembros de un gimnasio. Cada una de estas entidades tienen atributos los cuales se generan a partir de la identificación de los datos de nuestro problema. Por ejemplo, un atributo del estudiante es su nombre.



Definir tablas y campos: Existe una correspondencia entre las tablas y entidades, en este paso debemos representar las entidades como tablas en la base de datos y los atributos como campos en dichas tablas. Por ejemplo, la entidad estudiante se puede convertir en la tabla estudiantes y su atributo nombre sería un campo en dicha tabla.



Definir campos: Cada campo en la tabla debe tener un tipo, tales como el tipo entero. Existen un conjunto de tipos básicos, pero cada manejador de base de datos puede incluir algunos otros.



Definir relaciones y cardinalidad: Una vez definidas las tablas en nuestra base de datos debemos definir las relaciones entre ellas. Existen cuatro tipos de relaciones: de N a M o de muchos a muchos, de 1 a N o de uno a muchos, de N a 1 o de muchos a uno, y de 1 a 1 o de uno a uno.



Definir los campos nulos (null): Los campos nulos son aquellos que permiten que un campo no tenga ningún valor. En algunos casos esto no es deseable, por ejemplo, una clave primaria nunca debería tener un valor nulo. Es necesario definir que campos en nuestras tablas permiten valores nulos siendo no obligatorios y cuáles no siendo campos obligatorios.



Definir las claves primarias: Consiste en determinar cuál campo en cada una de nuestras tablas será la clave primaria, representando unívocamente cada fila de la misma. Podríamos también tener la necesidad de tener claves primarias compuestas por varios campos. Un ejemplo de clave primaria es el ID de un estudiante en la tabla estudiantes. Las claves primarias podrían ser auto numéricas en el caso de MySQL o podrían tener alguna regla especial para la definición del ID.



Definir las claves foráneas: Se definen basándose en las relaciones entre tablas. Cuando se define una clave foránea a parte de crear el campo hay que crear la referencia, es decir, decirle al manejador de bases de datos esa referencia. Además, se deben definir las restricciones para las claves foráneas. Por ejemplo, cuando se elimina un registro en la tabla referenciada, se debe establecer si se va a eliminar el registro en la tabla referenciante o se debe actualizar el registro en la tabla referenciante.



Agregar un índice por cada clave foránea: Esta es una buena práctica ya que se suele filtrar por ese campo.



Agregar un índice en los campos de búsqueda comunes: Los índices son ventajosos si se usan para acelerar los tiempos de ejecución de las consultas a una tabla, en particular, cuando la tabla es de gran tamaño. Sin embargo, pueden no ser beneficiosos cuando se insertan o eliminan registros de la misma porque hay que realizar un proceso de re-indexación, el cual toma tiempo. Por esto, es adecuado determinar dónde vamos a agregar nuestros índices. Naturalmente, ellos deberían ser agregados en aquellos campos que usualmente participan en una búsqueda, por ejemplo, si en la tabla estudiantes solemos hacer búsquedas por apellidos, entonces es natural crear un índice por el campo apellido. En aquellos casos donde la búsqueda involucra más de un campo, estos campos deben ser parte del índice. Entre otros, si se estila buscar a los estudiantes por su apellido y ciudad dónde viven, entonces estos dos campos deben ser parte del índice.

Esto es solo un resumen de los pasos a seguir cuando estás implementando las tablas en tu base de datos, esperamos que te sea muy útil. Y recuerda: ¡Siempre piensa en mejorar tus procesos!

1. Lección 2: Normalización
   1. [Conceptos Generales](https://learn.nextu.com/mod/lesson/view.php?id=10248&pid=P_WEB_DATABASE)

Forma eficiente. La normalización busca la opción que ocupe menos espacio y que no tenga redundancia de los datos.

Es una solución con muchas tablas pequeñas con sus respectivas relaciones

Formas normales: 1FN, 2FN, 3FN, 4FN y 5FN

Primera Forma Normal:



Ejemplo en caso de que se tenga una tabla con personas y estas posean más de un número telefónico:



Ventajas de la normalización:

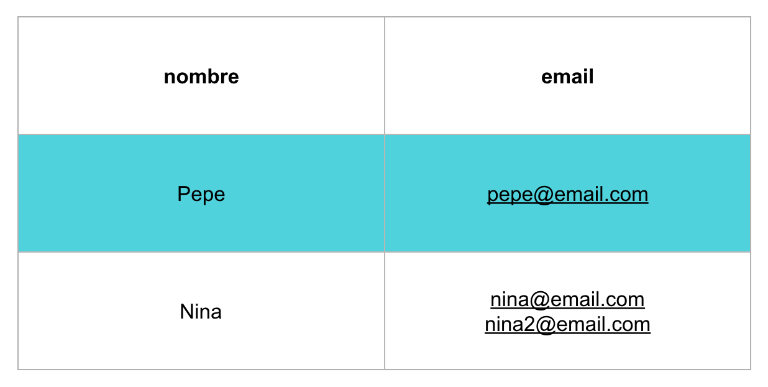
* Existe un lugar para cada dato. (ningún dato es redundante y los datos se pueden localizar fácilmente)
* Los datos ocupan menos espacio.

Desventajas de la normalización:

* Puede resultar ineficiente (debemos relacionar muchas tablas cuando necesitamos mezclar muchos datos de muchas fuenteso relaciones) y puede afectar el rendimiento en la consulta de los datos
* Puede resultar complejo.
  1. [Ejercicio Práctico 1](https://learn.nextu.com/mod/lesson/view.php?id=10248&pageid=46571&pid=P_WEB_DATABASE)

Desafío práctico

La normalización es un proceso importante en el diseño de nuestra base de datos. A continuación vamos a normalizar una tabla usando la primera forma normal. Supongamos que en una empresa ABC se tiene una tabla de usuarios, donde se guarda el nombre y el email (correo electrónico) de todos sus trabajadores. Dentro de la empresa existe un usuario que posee dos correos electrónicos, y ambos se almacenan dentro de la base de datos, tal como lo muestra la imagen a continuación:

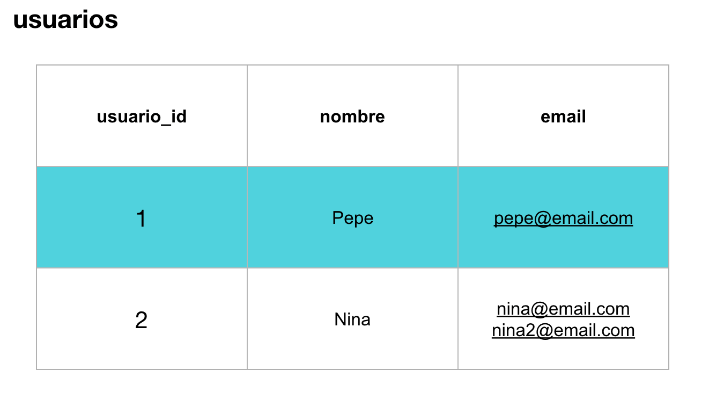


Tu rol en este desarrollo es aplicar la normalización de acuerdo a la primera forma normal. Uno de los miembros del equipo desarrollador ha comenzado ha trabajar en este proyecto y tiene algunos diseños preliminares de la tabla. Queremos que nos indiques cuál de estas opciones es la correcta de acuerdo con lo establecido por la primera formal normal (1FN).

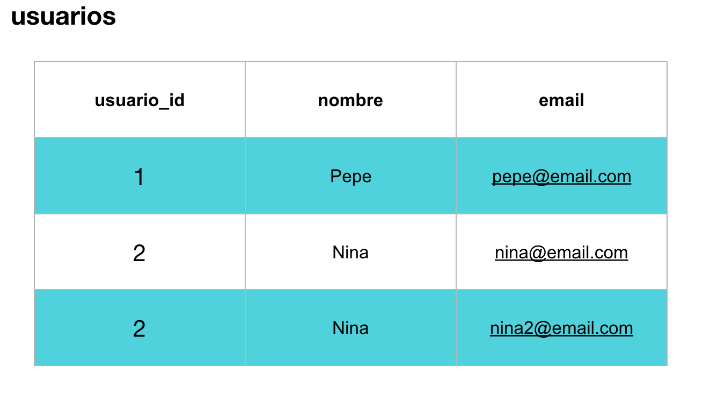
Opción 1



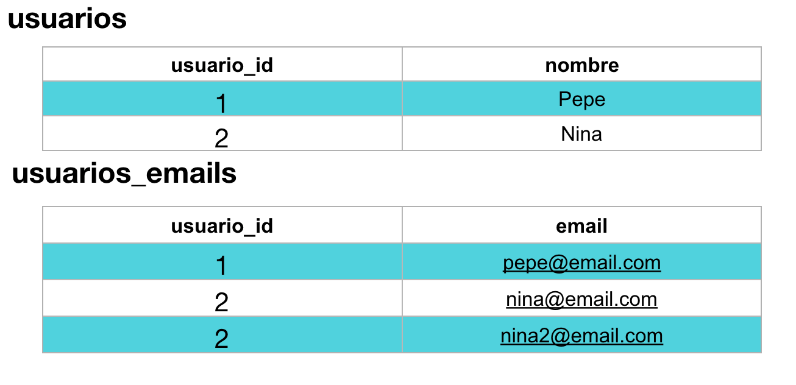
Opción 2



Opción 3



Opción 4



**La opción correcta es la Opción 4.**

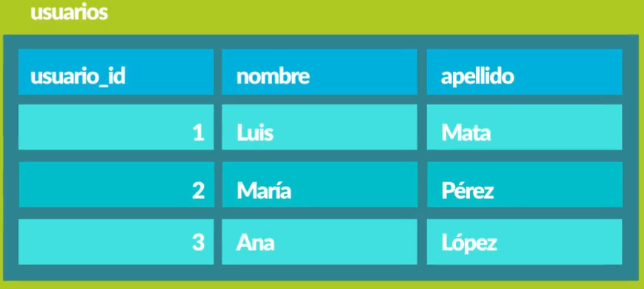
* 1. Formas Normales

Ejemplo de normalización:

2 FN: separar los datos por subgrupos



3FN – crear todas las tablas necesarias para asociar los usuarios a las presentaciones.





* 1. Ejercicio Práctico 2

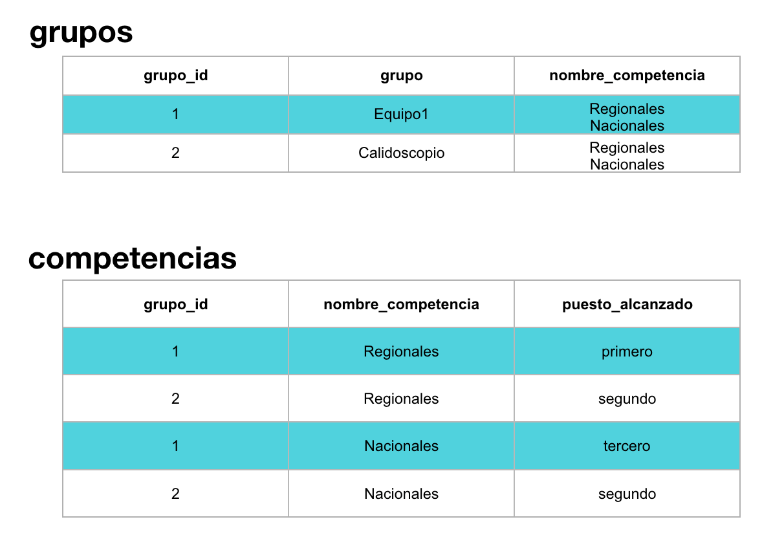
**Desafío práctico**

La normalización es un proceso importante en el diseño de nuestra base de datos. A continuación, vamos a normalizar una tabla usando la segunda forma normal. Se está desarrollando la competencia Nacional de Hip Hop en la ciudad, y ya se tienen los resultados parciales de la misma y los de la competencia clasificatoria llamada Regionales. Estos resultados son importantes para la nueva base de datos que se está implementando para la Aplicación de Danza Hip Hop.

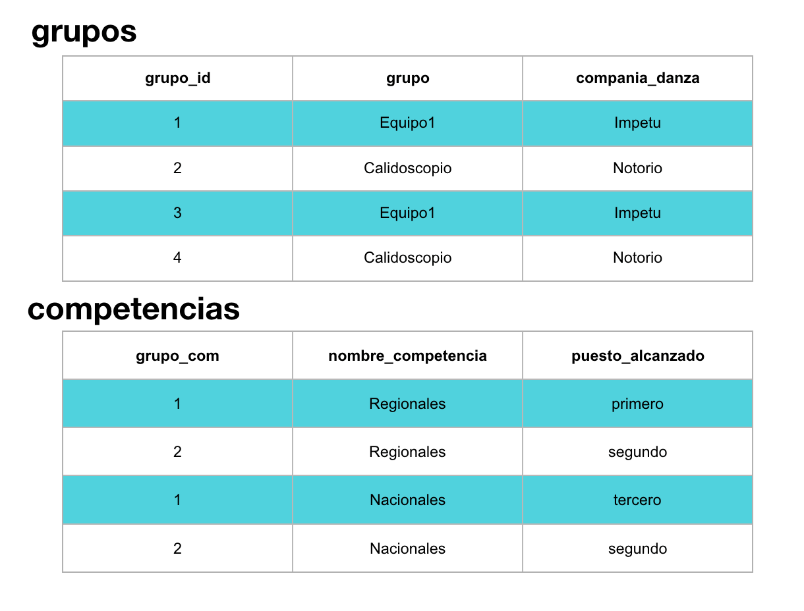


Tu rol en este desarrollo es aplicar la normalización de acuerdo con la segunda forma normal. Uno de los miembros del equipo desarrollador ha comenzado a trabajar en este proyecto y tiene algunos diseños preliminares de las tablas. Queremos que nos indiques cuál de estas opciones es la correcta.

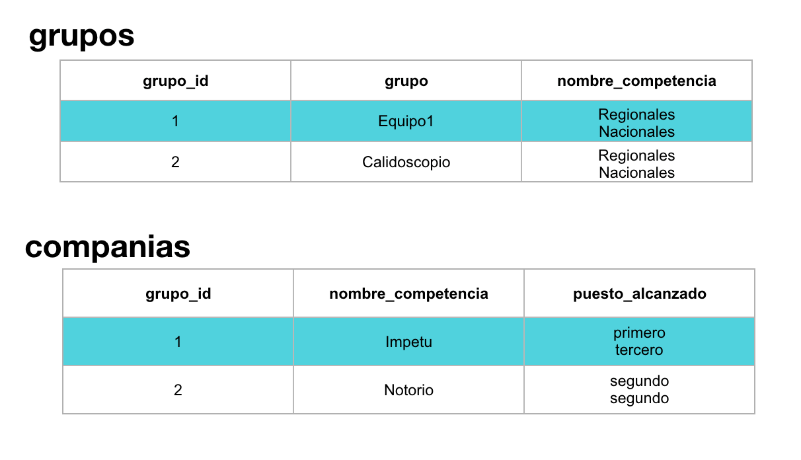
Opción 1



Opción 2



Opción 3



Opción 4

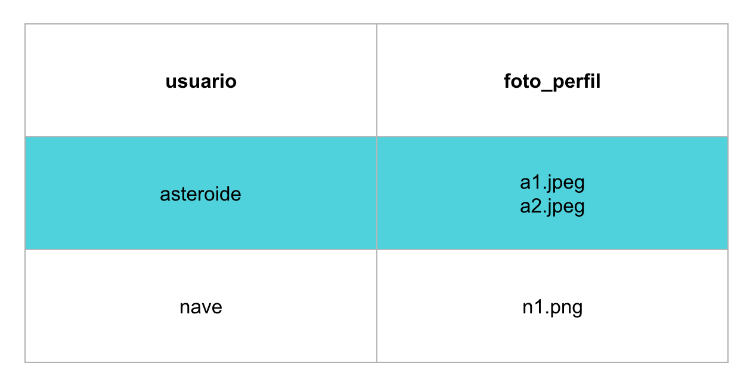
La tabla está en su segunda formal normal así que no hay que realizar ningún cambio.

* 1. Ejercicio Práctico 3

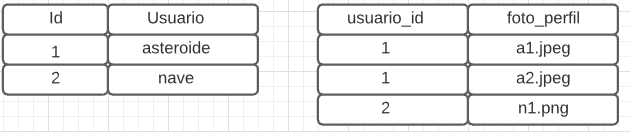
Desafío práctico

Hola, con este ejercicio te invitamos a practicar los conceptos de normalización en dos situaciones diferentes. Recuerda que el objetivo de este proceso es evitar la redundancia.

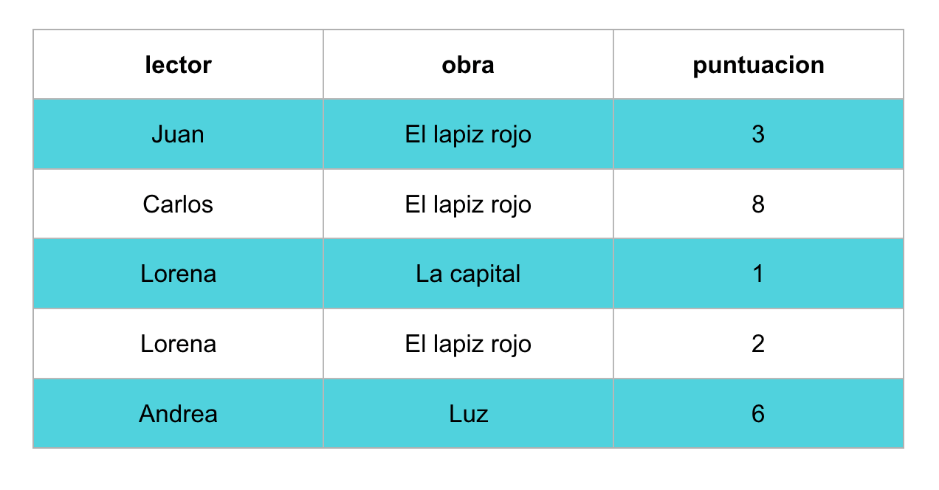
1) José está desarrollando su primer videojuego y generó la siguiente tabla con las fotos de perfil de los jugadores. ¿Está la tabla normalizada en su primera forma normal? Si no lo está, se desea que ayudes a José a convertir esta tabla en su primera forma normal.



No está normalizada. Se normaliza de la siguiente forma:



2) En una librería digital se colocan las obras y lectores con una puntuación del 1 al 10. Se ha generado la siguiente tabla. ¿Está la tabla normalizada en su segunda forma normal? Si no lo está, se desea que normalices la tabla en su segunda forma normal.



Solución:

|  |  |
| --- | --- |
| Tabla lector | |
| Id | lector |
| 1 | Juan |
| 2 | Carlos |
| 3 | Lorena |
| 4 | Andrea |

|  |  |
| --- | --- |
| Tabla obras | |
| Id | obra |
| 1 | El lapiz rojo |
| 2 | La capital |
| 3 | Luz |

|  |  |  |
| --- | --- | --- |
| Tabla obras\_puntuacion | | |
| lector\_id | obra\_id | Puntuacion |
| 1 | 1 | 3 |
| 2 | 1 | 8 |
| 3 | 2 | 1 |
| 3 | 1 | 2 |
| 4 | 3 | 6 |

* 1. ¿Sabías qué?

**Cápsula de conocimiento**

**Resumen de normalización**

La normalización es una forma sistemática de colocar los datos de manera no redundante y ocupando el menor espacio. La estrategia es dividir nuestros datos en muchas tablas para evitar datos repetidos y usar las relaciones como elemento para realizar las consultas complejas en la base de datos.

La normalización se lleva a cabo a través de reglas en secuencia: 1era, 2da y 3era forma normal.

Existen más formas normales, pero solo hablaremos de las primeras tres que son las importantes para el objetivo de nuestro curso. A continuación explicamos los planteamientos de estas formas normales:



* Todos los campos deben ser atómicos
* Evitar campos con más de un dato
* Evitar tablas con campos del mismo elemento. Ejemplo: telefono\_1, telefono\_2, telefono\_3
* Para llevar una base de datos a su primera forma normal debemos colocar los campos que se repiten o no están atómicos en otra tabla y relacionarlos a través del ID (o identificador).



* Si hay datos de distintas entidades en una misma tabla, se observan las filas que tengan contenido repetido.
* Para llevar una base de datos a su segunda forma normal debemos separar las tablas que contienen datos referentes a distintas entidades en dos o más tablas enlazadas por claves foráneas (cada clave foránea debe apuntar a la clave primaria de la tabla correspondiente).



* En las tablas de una misma entidad, aún después de llegar a la segunda forma normal, se repiten valores de columnas.
* Para llevar una base de datos a su tercera forma normal debemos separar los datos repetidos y enlazarlos por el id.

Esperamos que a través de este resumen puedas comprender mejor los conceptos de normalización y en cuáles situaciones es oportuno utilizar cada una de ellas.

1. Lección 3: Progresando en SQL
   1. [Sentencias con GROUP BY y HAVING](https://learn.nextu.com/mod/lesson/view.php?id=10249&pid=P_WEB_DATABASE)

GROUP BY

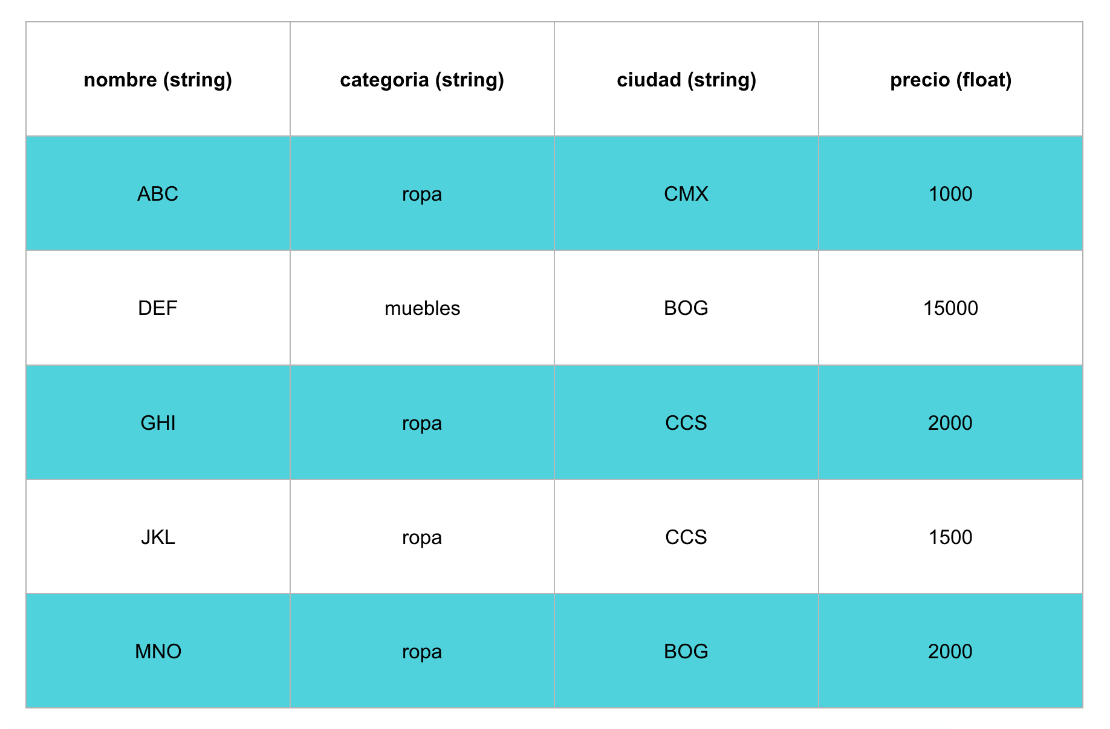
HAVING se aplica a la condición de un grupo

ORDER BY

* 1. Actividad Interactiva 1

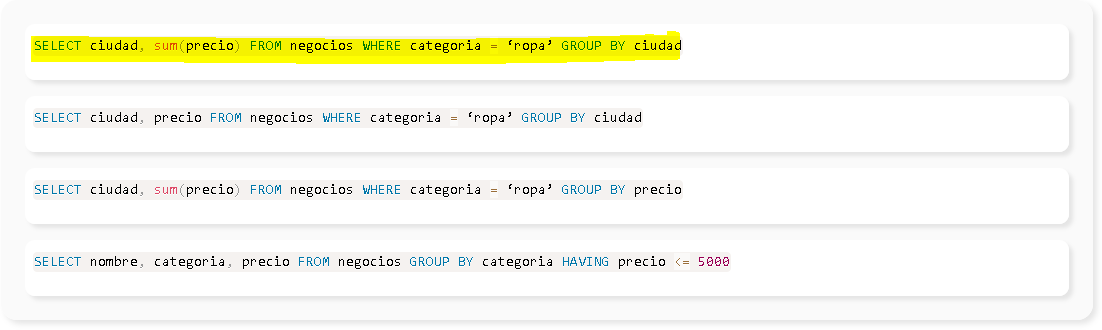
**Prueba tus conocimientos**

MiNuevoNegocio es una tienda que ofrece a la venta negocios en todo el país. Ellos tienen un sistema para el control de las ventas, el cual tiene una base de datos que incluye la tabla negocios con todos los negocios disponibles. Los campos y tipos de la tabla se muestran a continuación:



Un cliente está interesado en comprar un negocio cuyo precio categoría sea “ropa”. Él desea ver los resultados agrupados por ciudad. El resultado debe mostrar la ciudad, ganancia total.

**¿Qué sentencia SQL deberías usar para cumplir con este requerimiento?**



* 1. Uniendo tablas con JOIN y Condiciones

JOIN tabla ON igualdad

* 1. Actividad Interactiva 2
  2. ¿Sabías que?
  3. Taller: Tienda de Zapatos

5. UNIDAD 1: Prueba