**UNIDAD 2: MySQL**

1. [UNIDAD 2: Introducción](https://learn.nextu.com/mod/page/view.php?id=10253&pid=P_WEB_DATABASE)

**MySQL**

Con un enfoque que busca que puedas aplicar los conceptos de bases de datos aprendidos hasta el momento, en esta unidad usaremos MySQL, con la ayuda de la línea de comando y la herramienta gráfica MySQL Workbench, para que te vayas familiarizando con la utilización de los conceptos de bases de datos relacionales y su representación en MySQL. Continuando con nuestro enfoque teórico-práctico, cada lección proporciona un número de talleres, donde tendrás el tiempo y espacio de poder explorar y practicar los comandos y sentencias explicadas en video y lecturas.

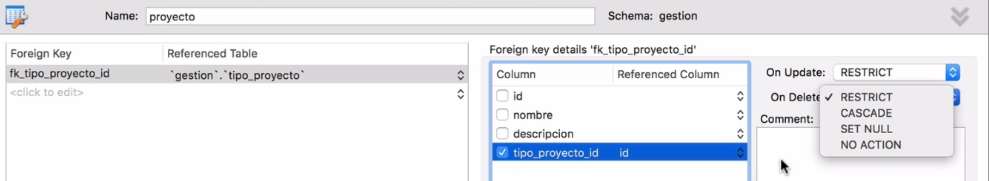
**Objetivos de aprendizaje**

1. Crear **Triggers** y **Procedures** en MySQL para operar nuestra base de datos de forma automática.
2. Crear claves primarias y foráneas y sus restricciones a una solución de base de datos en MySQL.
3. Usar las sentencias SQL **GROUP BY**, **HAVING** y **JOIN** en MySQL para mezclar tablas de diversas maneras.
4. Diferenciar los distintos tipos de funciones predefinidas en MySQL con la finalidad de poder realizar operaciones más avanzadas sobre nuestra base de datos.
5. Crear queries en MySQL con la finalidad de realizar consultas a bases de datos de mediana complejidad.
6. Lección 1: Merge
   1. [Creando Tablas con Índices y Claves Foráneas](https://learn.nextu.com/mod/lesson/view.php?id=10254&pid=P_WEB_DATABASE)

Recomendado utf8 – Ya que permite diversidad de caracteres. (default collation)

Create Scheme o create database, en MySQL funciona

Restricciones de llaves foráneas en actualización o en eliminación:

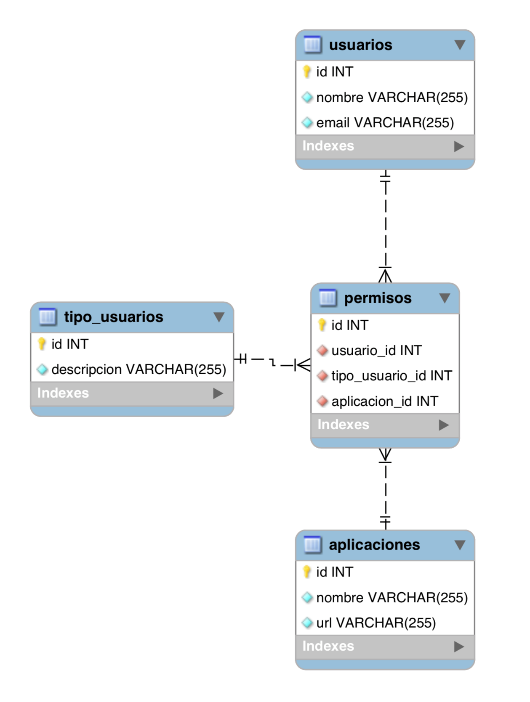


* 1. Taller 1

**Aprende haciendo**

**Taller de Creación de Índices: Usuarios de la Organización**

¡Bienvenido! La idea de este taller es que puedas practicar con índices y claves foráneas.  
  
Dadas las siguientes tablas:



Donde:

* **usuarios:** Representa a los usuarios de una empresa.
* **aplicaciones:** Representa las aplicaciones que manejan internamente.
* **tipo\_usuarios:** Es una tabla con los posibles tipos de usuarios, por ejemplo, Administrador, Analista, etc.
* **permisos:** Asociación que indica el tipo de usuario en una aplicación particular.

Se desea que ejecutes los siguientes pasos usando la herramienta MySQL Workbench:

1. Crear las tablas según el diagrama.
2. Crear un índice de unicidad para el campo email en la tabla usuarios.
3. Crear las claves foráneas de la tabla permisos definiendo un constrain CASCADE para ON UPDATE y ON DELETE
   * fk\_usuario\_id
   * fk\_tipo\_usuario\_id
   * fk\_aplicacion\_id
4. Crear los índices para cada una de las claves foráneas.
   * ix\_usuario\_id
   * ix\_tipo\_usuario\_id
   * ix\_aplicacion\_id

**Puntos de verificación**

Para confirmar que tu experiencia ha sido llevada a cabo con éxito, revisa los siguientes puntos:

1. Verifica las tablas: puedes usar la consulta SHOW TABLES para ver las tablas creadas usando el terminal
2. Verifica el índice de unicidad: Intenta crear dos registros con el mismo email y MySQL no debe dejar guardar el segundo registro.
3. Verifica clave foránea: Cambia los ids de la tabla tipo\_usuarios y verifica que cambiaron en la tabla usuario
4. Índices creados: Verifica que tengas índices, puedes hacerlo con la consulta DESCRIBE permisos

Es muy importante definir las claves foráneas y los índices para mantener la consistencia de la base de datos y tener buen rendimiento del sistema.

* 1. JOINS en MySQL

Tienes 2 tipos adicionales de JOIN:

LEFT JOIN – Ejemplo:

SELECT \* FROM lista\_a LEFT JOIN lista\_b ON lista\_a.email = lista\_b.email

En este ejemplo deja a lista\_a como una columna fija

RIGHT JOIN – deja fija la lista de lista\_b

* 1. ¿Sabías qué? -1

**Cápsula de conocimiento**

**Tipos de Datos en MySQL**

¡Bienvenido! En esta lectura te proporcionamos un resumen de los tipos de datos más utilizados en MySQL, categorizados en tres grupos:



**Strings**

| **Tipo** | **Descripción** |
| --- | --- |
| **CHAR(largo)** | Secuencia de caracteres (string) de tamaño fijo hasta 255 |
| **VARCHAR(largo)** | Secuencia de caracteres (string) de tamaño variable hasta 255 |
| **TEXT** | String hasta 65535 caracteres |
| **MEDIUMTEXT** | String hasta 16777215 caracteres |
| **LONGTEXT** | String hasta 4294967295 caracteres |



**Numéricos**

| **Tipo** | **Descripción** |
| --- | --- |
| **TINYINT** | Entero desde -128 a 127 (usado sin signo para booleanos) |
| **INT** | Entero desde -2147483648 a 2147483647 |
| **BIGINT** | Entero desde -9223372036854775808 a 9223372036854775807 |
| **FLOAT** | Decimal Pequeño (4 bytes) |
| **DOUBLE** | Decimal Grande (8 bytes) |



**Otros**

| **Tipo** | **Descripción** |
| --- | --- |
| **DATE** | Fecha en formato YYYY-MM-DD |
| **DATETIME** | Fecha y hora en formato YYYY-MM-DD HH:MM:SS |
| **TIME** | Hora en formato HH:MM:SS |
| **ENUM** | Enumerado |

Esperamos que esta guía te sea útil a la hora de escoger tus datos en MySQL.

* 1. ¿Sabías qué? – 2

**Cápsula de conocimiento**

**Resumen de Comandos**

En esta lectura encontrarás un resumen de los comandos de MySQL para crear tablas con sus claves e índices, así como también de la sentencia Join para unir y comparar tablas basados en el uso de ciertos campos de nuestras tablas.

* **Índices:**

CREATE TABLE [tabla] ([col 1] [tipo 1],....[col 2] [tipo 2],PRIMARY KEY ([campo\_pk])KEY [key\_name1] ([campos2]),...KEY [key\_name2] ([campos2]),)

* **Claves Foráneas:**

CREATE TABLE [tabla] ([col 1] [tipo 1],....[col 2] [tipo 2],PRIMARY KEY ([campo\_pk])KEY [key\_name1] ([campos2]),...CONSTRAIN [nombre\_fk1] FOREIGN KEY ([campo\_origen1]) REFERENCES [tabla destino1]([campo\_destino1]),CONSTRAIN [nombre\_fk2] FOREIGN KEY ([campo\_origen2]) REFERENCES [tabla destino2]([campo\_destino2]))

* **Left Join:**

SELECT \* FROM [tabla\_izquierda] LEFT JOIN [tabla\_derecha] ON [condición que une las tablas]

* **Right Join:**

SELECT \* FROM [tabla\_izquierda] RIGHT JOIN [tabla\_derecha] ON [condición que une las tablas]

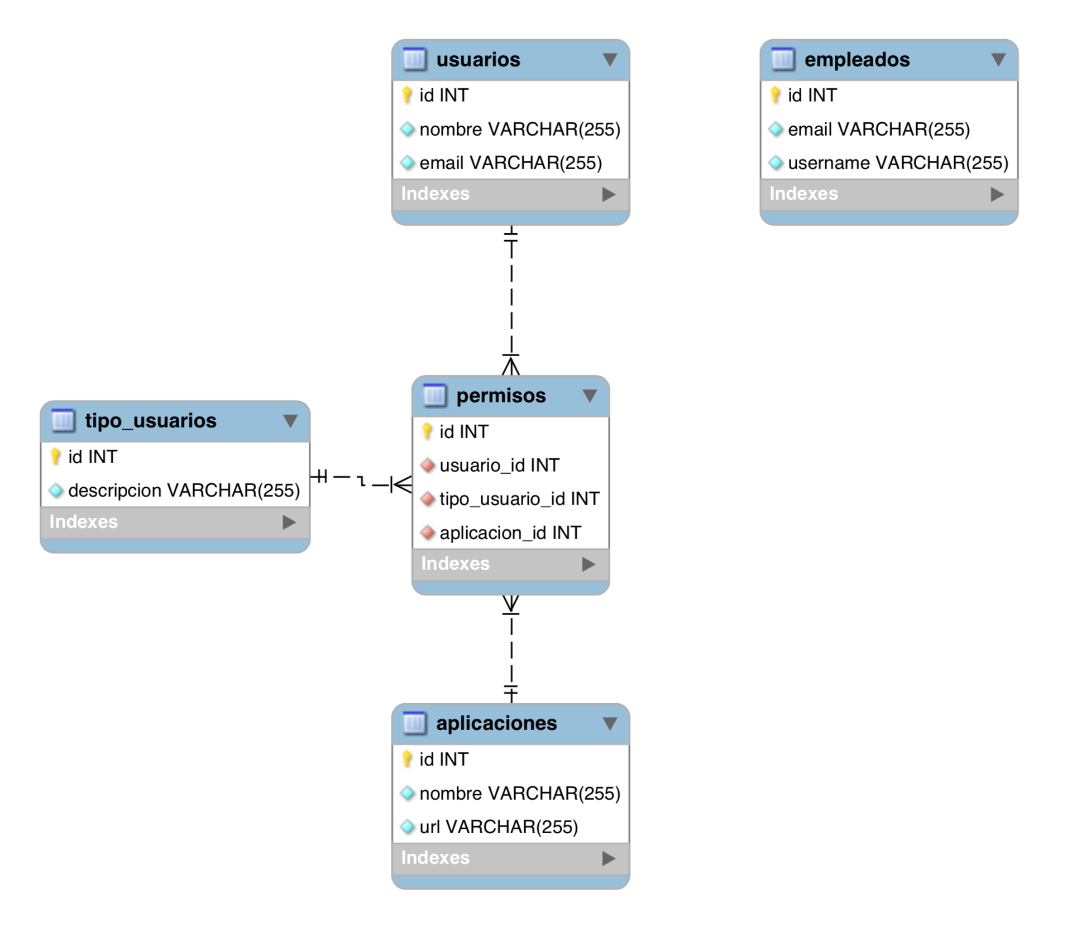
Recuerda que hay variantes de estas instrucciones, que hemos omitido por simplicidad pero que puedes encontrar en el sitio oficial de MySQL.   
Esperamos que esta guía te sea útil en la construcción y manejo de tus tablas.

* 1. Taller 2

**Aprende haciendo**

**Taller de LEFT JOIN y RIGHT JOIN: Usuarios de la Organización**

¡Bienvenido! Hemos diseñado este taller para que puedas practicar con la sentencia Join, usando la siguiente base de datos:



Donde:

* **usuarios:** Representa a los usuarios de una empresa.
* **aplicaciones:** Representa las aplicaciones que manejan internamente.
* **tipo\_usuarios:** Es una tabla con los posibles tipos de usuarios, por ejemplo, Administrador, Analista, etc.
* **permisos:** Asociación que indica el tipo de usuario en una aplicación particular.
* **empleados:** representa una tabla del sistema central antiguo.

Se desea que ejecutes los siguientes pasos usando la herramienta MySQL Workbench:

1. Crear las tablas según el diagrama.
2. Llenar las tablas con valor de ejemplo.
   * Crear 3 usuarios en la tabla usuarios (luis, maría, pedro)
   * Crear 2 aplicaciones (app1, app2)
   * Crear 2 tipo\_usuarios (admin, analista)
   * Crear permisos
   * Crear 4 empleados (luis, maría, ana, camila)
3. Hacer una consulta con LEFT JOIN para ver cuáles usuarios están en la tabla empleados y no están en la tabla usuarios a través del correo electrónico.
4. Hacer una consulta con RIGHT JOIN para ver cuáles usuarios están en la tabla empleados y no están en la tabla usuarios a través del correo electrónico.
5. Hacer una consulta con JOIN para ver cuáles aplicaciones están permitidas para los usuarios.

**Puntos de verificación**

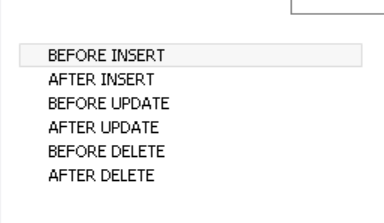
Para confirmar que tu experiencia ha sido llevada a cabo con éxito, revisa los siguientes puntos:

1. Verifica las tablas: puedes usar la consulta SHOW TABLES para ver las tablas creadas usando el terminal
2. Verificar datos: Puedes hacer un SELECT de cada tabla para verificar el contenido
3. LEFT JOIN: El resultado debe mostrar: ana y camila
4. RIGHT JOIN: El resultado debe mostrar: pedro
5. JOIN: Verifica la información de tu tabla permisos con tu consulta JOIN

La sentencia **JOIN** es comúnmente utilizada en las operaciones realizadas sobre una base de datos, de allí la importancia de que te familiarices con ella.

1. Lección 2: Triggers y Procedures
   1. [Triggers](https://learn.nextu.com/mod/lesson/view.php?id=10255&pid=P_WEB_DATABASE)

Triggers



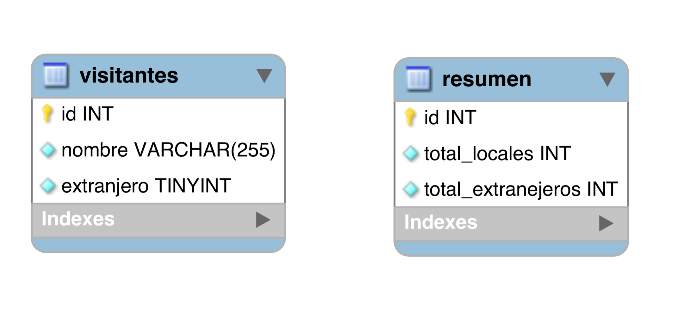
NEW o OLD, para insertar el nuevo o el registro anterior. Esto podría generar un historial

* 1. [Taller 1](https://learn.nextu.com/mod/lesson/view.php?id=10255&pageid=46621&pid=P_WEB_DATABASE)

**Aprende haciendo**

**Taller de Triggers: Visitantes de una atracción turística**

El sistema ABC mantiene un registro de las personas que visitan una atracción turística muy conocida en la ciudad de un país maravilloso. Las tablas que usa el sistema son:



Donde:

* **visitantes:** guarda el nombre de cada persona y si es extranjero o no. Se guarda como TINYINT donde 0 indica no es extranjero y 1 indica que sí lo es.
* **resumen:** tiene un solo registro con los valores de la cantidad de visitas totales para cada caso. Esta tabla fue inicializada en 0. Representa las aplicaciones que manejan internamente.

**Meta**

Crea un trigger para actualizar la tabla resumen cada vez que se ingrese un registro en la tabla visitantes. Para esto debes crear las tablas e ir llenándolas con valores.

**Tips**

Puedes usar un

**SQL UPDATE**

de la tabla resumen haciendo cada total = total + 1 pero debes colocar un IF para saber cuál total debes afectar. Dentro del cuerpo del trigger puedes poner condicionales, tales como IF, THEN, ELSE, END, IF.

**Puntos de verificación**

Para confirmar que tu experiencia ha sido llevada a cabo con éxito, revisa los siguientes puntos:

1. Verifica las tablas: puedes usar la consulta SHOW TABLES para ver las tablas creadas usando el terminal
2. Crear registros: Agrega varios visitantes
3. Verifica el resumen: Cuenta los visitantes y compara con la tabla resumen

En este taller has podido comprobar la fortaleza de los triggers en la definición de ciertas acciones que se disparan cuando ocurre una condición determinada en el sistema.

Solución:

CREATE DEFINER=`root`@`localhost` TRIGGER `abc`.`visitantes\_AFTER\_INSERT` AFTER INSERT ON `visitantes` FOR EACH ROW

BEGIN

DECLARE locales,extranjeros,totales\_locales,totales\_extranjeros INT;

IF NEW.extranjero = 0 THEN

SET locales = (SELECT total\_locales FROM abc.resumen LIMIT 1);

SET totales\_locales = locales + 1;

UPDATE resumen SET total\_locales = totales\_locales WHERE id=1;

END IF;

IF NEW.extranjero = 1 THEN

SET extranjeros = (SELECT total\_extranjeros FROM abc.resumen LIMIT 1);

SET totales\_extranjeros = extranjeros + 1;

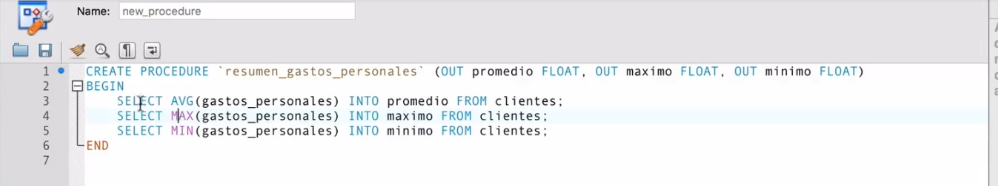
UPDATE resumen SET total\_extranjeros = totales\_extranjeros WHERE id=1;

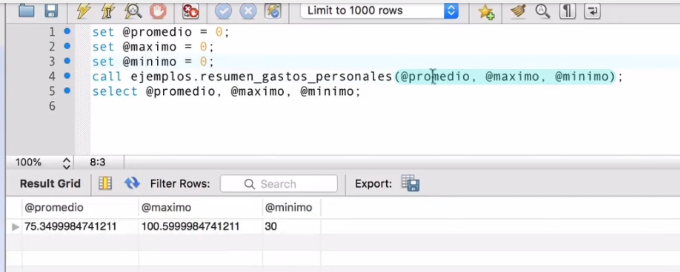
END IF;

END

* 1. [Stored Procedures](https://learn.nextu.com/mod/lesson/view.php?id=10255&pageid=46622&pid=P_WEB_DATABASE)

Ejemplo:



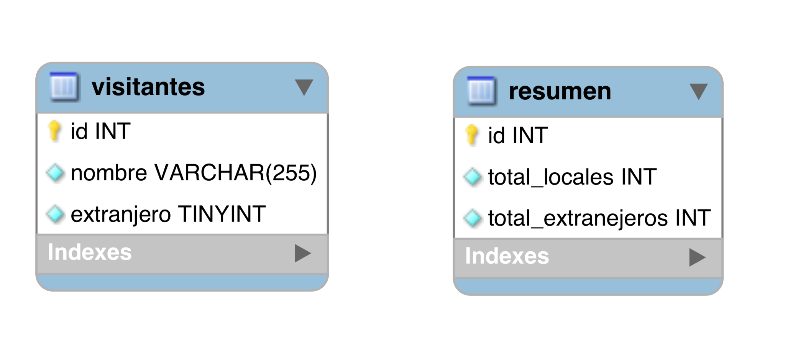


* 1. Taller 2

**Aprende haciendo**

**Taller de Stored Procedures: Visitantes de una atracción turística**

En este taller vamos a seguir utilizando nuestro ejemplo del sistema ABC que guarda el registro de las personas que visitan una atracción turística. Las tablas que usa el sistema son:



Supongamos que, por alguna razón desconocida, los totales se desincronizan y no reflejan el valor real de las visitas. Entonces podemos resolver esto creando un **Stored Procedure** que recalcule estos totales en cualquier momento.

**Meta**

Crea un **Stored Procedure** para actualizar la tabla resumen, dados los registros de la tabla visitantes. Para esto debes crear las tablas e ir llenándola con valores. El **Stored Procedure** debe calcular los totales haciendo un Group by de la tabla visitantes.

**Puntos de verificación**

Para confirmar que tu experiencia ha sido llevada a cabo con éxito, revisa los siguientes puntos:

1. Verifica las tablas: puedes usar la consulta SHOW TABLES para ver las tablas creadas usando el terminal
2. Crear registros: Agrega varios visitantes
3. Verifica el resumen: Coloca los totales en 0, ejecuta el stored procedure, cuenta los visitantes y compara con la tabla resumen.

En este taller has podido comprobar las ventajas de usar **Stored Procedure** en la definición de ciertas acciones que se pueden ejecutar en cualquier momento en el sistema.

**Solución:**

CREATE DEFINER=`root`@`localhost` PROCEDURE `recalcular\_totales`()

BEGIN

DECLARE extranjeros, locales INT;

SET extranjeros = (SELECT COUNT(\*) FROM abc.visitantes WHERE extranjero = 1);

SET locales = (SELECT COUNT(\*) FROM abc.visitantes WHERE extranjero = 0);

UPDATE resumen SET total\_extranjeros = extranjeros, total\_locales = locales WHERE id=1;

END

* 1. ¿Sabías qué?

**Cápsula de conocimiento**

**Resumen de Comandos**

En esta lectura encontrarás un resumen de la sintaxis usada para crear Triggers, que son los mecanismos de MySQL para realizar un conjunto de acciones, luego o antes de una inserción, actualización o borrado de un registro en alguna de nuestras tablas. También te proporcionamos un resumen de la sintaxis usada para crear los Stored Procedures, procedimientos, para la ejecución de ciertas acciones necesarias para automatizar ciertas tareas en nuestra base de datos.

* **Triggers:**

CREATE TRIGGER [nombre\_trigger] [AFTER|BEFORE] [UPDATE|INSERT|DELETE] ON [nombre\_tabla]FOR EACH ROW BEGIN …[cuerpo trigger]END

* **Procedures:**

CREATE PROCEDURE [nombre\_procedure] (parametros) BEGIN …[cuerpo procedure]END

Recuerda que los parámetros se clasifican en parámetros de entrada (IN) y parámetros de salida (OUT), y cada uno de ellos tiene un tipo asociado.

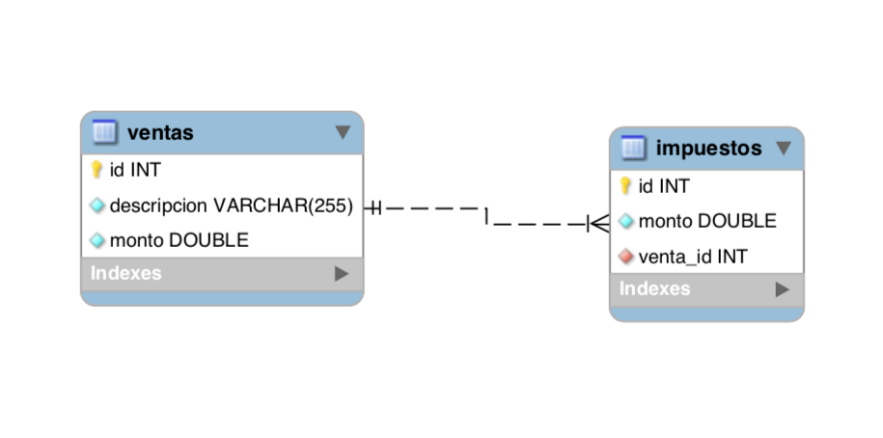
Esperamos que esta guía sea útil para reforzar tu aprendizaje de estos comandos.

* 1. Ejercicio Práctico

**Desafío práctico**

**Registrando Impuestos**

La idea de este ejercicio es que practiques con el uso de Triggers en MySQL. En esta oportunidad queremos que crees una forma automática de calcular y extraer los impuestos de una tabla de ventas al mayor. Para ello debes crear las tablas ventas e impuestos de la siguiente estructura:



Luego de crear la tabla debes definir un trigger, que agregue un registro en la tabla impuestos con el 10% del monto de cada venta nueva, y por último debes probar que tu trigger funciona.

**Solución:**



1. Lección 3: Funciones Predefinidas
   1. [Funciones Predefinidas de String](https://learn.nextu.com/mod/lesson/view.php?id=10256&pid=P_WEB_DATABASE)

CONCAT()

UCASE() 🡪 Upper Case

TRIM()

IF( Condicion, lo que se devuelve , lo que no es verdad)

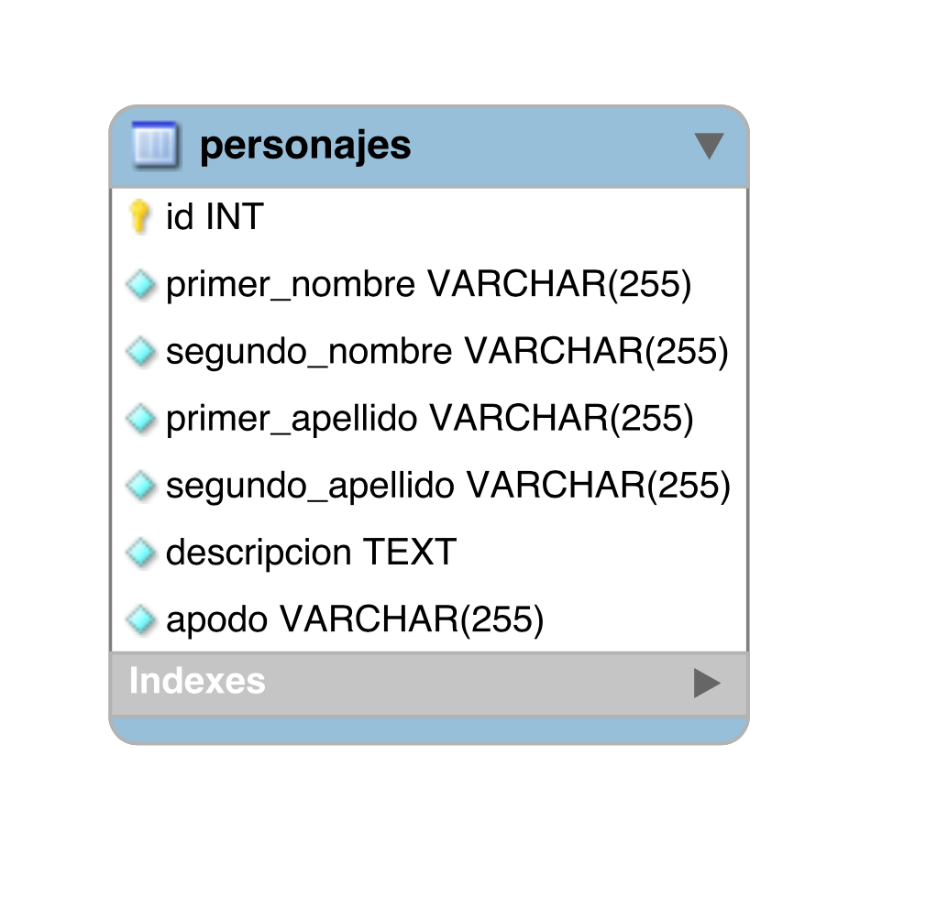
MD5() 🡪 Criptográficos

* 1. Taller 1

**Aprende haciendo**

**Taller: Funciones de Manipulación de String**

¡Bienvenido! En este taller seguimos practicando con las funciones predefinidas de MySQL. En este caso tenemos la tabla personajes del guión de una nueva película. A continuación te mostramos la tabla personajes:



**Meta**

Para practicar con las funciones de String debes crear una tabla con la estructura anterior e insertar varios registros para tu control. Luego realiza las siguientes consultas:

1. Consulta un campo nombre\_completo que contenga (primer\_nombre, segundo\_nombre, primer\_apellido y segundo apellido).
2. Consulta un campo llamado descripcion\_buena que sea la descripción, pero de su contenido reemplaza todas las ocurrencias de la palabra “Yes” por la palabra “Sí”.
3. Consulta los apodos, pero con las siguientes condiciones:
   * El apodo debe devolverse en minúscula.
   * Si el apodo es NULL devolver “N/A”. Nota que esto lo puedes hacer con un condicional IF.

**Puntos de verificación**

Para confirmar que tu experiencia ha sido llevada a cabo con éxito, revisa los siguientes puntos:

1. Verifica la tabla: puedes usar la consulta SHOW TABLES para ver la tabla personajes, usando el terminal
2. Ingresa datos: crea registros con valores diferentes para cada uno de los campos
3. Nombre completo: Verifica que tu consulta coincida con la concatenación de los nombres
4. Descripción buena: Asegúrate que las descripciones tienen la palabra “Yes” y verifica que cuando haces la consulta efectivamente se reemplazan.
5. Apodos: Crea apodos con algunas letras en mayúsculas y otros vacíos, luego al hacer la consulta revisa que se devuelvan en minúsculas o “N/A” cuando aplique.

Esperamos hayas disfrutado de la actividad, pero sobre todo que hayas aprendido a usar las correspondientes funciones predefinidas de MySQL.

Solución:

INSERT INTO taller.personajes VALUES (null,'Jose','Rodrigo','Sancho','Castro','Presente siempre en la cocina','Igo');

INSERT INTO taller.personajes VALUES (null,'Fabiola','María','Sanchez','Rodríguez','Es una sirvienta','Machi');

INSERT INTO taller.personajes VALUES (null,'Raul','Fabricio','Solórzano','Tarantino','Le gusta el Futbol','Ichio');

INSERT INTO taller.personajes VALUES (null,'María','Jose','Miranda','Salas','Es una persona intelectual','Mira');

SELECT CONCAT(primer\_nombre,' ',segundo\_nombre,' ',primer\_appelido,' ',segundo\_apellido) nombre\_completo FROM taller.personajes;

SELECT REPLACE(descripcion,'Yes','Sí') FROM taller.personajes;

SELECT IF(apodo IS NULL,'N/A',LCASE(apodo)) FROM taller.personajes;

* 1. Funciones Matemáticas y Fecha
  2. Taller 2
  3. ¿Sabías qué?

5. Lección 4: Subqueries y Alter table

6. UNIDAD 2: Prueba

zczxc