MYSQL. Primera clase.

Grabación 22/04/2023

Considerar todos los requerimientos del cliente. ¿Qué necesitaría tener?

**Sobre la normalización:**

1. Primera forma

Cada columna de tu tabla solo puede tener 1 valor.

Ej.: No debes tener una columna de dirección en tu tabla que aliste la dirección, la ciudad, el estado y el código postal, todos separados por comas.

1. Segunda forma

Cada columna de tu tabla que no sea una clave (principal o foránea) debe tener valores únicos.

Ej.: Si tienes una tabla de películas con una columna de categorías, no deberías repetir una categoría.

1. Tercera forma

No puedes tener una columna sin clave que dependa de otra columna sin clave.

Ej.: Si tienes una tabla de libros con columnas nombre\_editorial y dirección\_editorial, dirección\_editorial y nombre\_editorial deben separarse en una tabla separada y vincularse a los libros con una clave foránea. dirección\_editorial depende de nombre\_editorial y ninguna columna es una columna clave

**Para crear tablas.**

1. Voy a modelos, que es el segundo ícono en el lado izquierdo de mysql workbench y clicko en el más que está a lado de models.
2. En physical schemas tengo los datos de mi tabla y haciendo click ahí, puedo denominar mi tabla, en este caso: base\_datos\_productos.
3. Hago click en diagramas y voy al dibujo que dice insertar tabla.
4. Hago doble click en la tabla y ahí puedo agregarle las características que necesito.

A screenshot of a computer

Description automatically generated

pk: llave primaria. MARCA

No null: que no se acepta nulos. MARCA

Uq: unique

Binary: que tendrá dos datos

Un: que si no tiene signo va de cero a algo. MARCA

Zero fill: el campo largo es de tres pero comienza con 0.

AI: autoincrementar, que si no doy el ID que incremente solo después de lo que yo le doy ultimo. SE MARCA

G: cálculos

A screenshot of a computer

Description automatically generated el con transparencia es boolean pero deja como tinyint por que 0 es falso y 1 es verdadero.

1. Cuando ya tengo todas mis tablas cargadas, le indico que tipo de relación hay entre cada una, y para eso hay una columna con los tipos de relaciones a la izquierda.
2. Para relación uno a uno: selecciono el que es el padre (el producto) y le marco el primer cuadrito. Se crea automáticamente la llave foránea en el padre y ahí debo cambiar el nombre, en vez de imágenes, debe ser imagen.
3. Para relación uno a muchos: uno a n. que es el segundo cuadrito. Se crea automáticamente la llave foránea en el padre y ahí debo cambiar el nombre, en vez de categorías, debe ser categoría.
4. Para relación mucho a muchos: n a n, que es el siguiente cuadrito. Esto crea una tercera tabla en donde debemos desactivar el uk para los dos primeros y crear un id que sea la llave primaria.

Se puede exportar el archivo como png y esa es

**CONVENCIONES**

* que el nombre de la tabla sea plural y TODO en minúsculas: hazlo en plural (por ejemplo, usuarios, clientes potenciales, sitios, clientes, capítulos, cursos, módulos)
* utiliza "id" como clave primaria: asígnale el nombre id (también hazlo de incremento automático).
* nombra a las claves foráneas con nombre\_tabla\_singular\_id: cuando hagas referencia a una clave primaria en otra tabla asígnale el nombre [nombre singular de la tabla a la que haces referencia]\_id (por ejemplo, user\_id, lead\_id, sitio\_id, cliente\_id, capítulo\_id, curso\_id, módulo\_id).
* utilizas created\_at y updated\_at como columnas para la marca de tiempo en CADA tabla que crees.

**TIPOS DE DATOS**

* Varchar (número de caracteres): Se utiliza para almacenar valores no numéricos que pueden tener hasta 255 caracteres. Se llama VARCHAR porque puede almacenar un número variable de caracteres y solo usará el espacio requerido para cada registro que se almacena en la base de datos. VARCHAR debe usarse para valores con diferentes longitudes de caracteres como un correo electrónico, first\_name o last\_name.
* Char (número de caracteres)
* También se utiliza para almacenar valores no numéricos, sin embargo, utilizará todo el espacio para el número establecido de caracteres independientemente del valor que se agregue. Por ejemplo, si establezco CHAR(15) e intento almacenar el valor "Codificación", utilizará el equivalente a 15 caracteres aunque "Codificación" tenga solo 12 caracteres. Char es bueno para usar en cosas que siempre tendrán un número determinado de caracteres. Char funcionaría bien para algo como abreviación\_regiones.
* INT: Se usa para almacenar números enteros; Las columnas que encontrarás principalmente utilizando INT son cosas como un identificador único para cada tabla. La mayoría de filas de una tabla no superará los 2.100 millones de registros. INT es bueno para la mayoría de los valores numéricos normales como un número\_de\_teléfono o un código\_postal; unsigned (solo números positivos): puede almacenar valores numéricos desde 0 hasta 4294967295; signed (números positivos y negativos): puede almacenar valores numéricos desde -2147483648 hasta 2147483647
* BIGINT: se usaría para columnas que necesitan almacenar grandes números. En la mayoría de los casos, no necesitarías BIGINT, pero si quisieras almacenar algo como una identificación de Facebook al usar la API de Facebook, dado que tienen más de mil millones de usuarios, la identificación deberá ser un tipo de datos de BIGINT; unsigned (de nuevo, solo números positivos): puede almacenar valores numéricos desde 0 hasta 18446744073709551615; signed (números positivos y negativos): puede almacenar valores numéricos desde 9223372036854775807 hasta -9223372036854775808.
* TINYINT: es bueno para usar con números que serán relativamente pequeños. Un buen ejemplo de algo que usaría un TINYINT es el identificador de nivel de usuario (0 - usuario inactivo, 1 - usuario activo, 9 - administrador); unsigned: puede almacenar valores numéricos desde 0 hasta 255; signed: puede almacenar valores numéricos desde -128 hasta 127
* FLOAT: Se utiliza para almacenar números de punto flotante (números que deben tener lugares decimales). Una columna de ejemplo para esto sería como item\_cost.
* TEXT: Se utiliza para almacenar una gran cantidad de texto, como una descripción, un mensaje o un comentario. Usa esto para cualquier texto que VARCHAR() sea demasiado pequeño para manejar.
* DATETIME: Se utiliza para almacenar una fecha y hora en el formato AAAA-MM-DD hh:mm:ss

Aber jaz, yo creo que podes entender como una caja que contiene lo importante, cual se relaciona con cual y que tipo de relación es la que tienen, dependiente? Independiente? Look for that para hacer la relación.

CLASE GRABADA DEL: 22 DE JULIO  
A screenshot of a computer

Description automatically generatedEsto hará que en el caso de que se borre un dato que es esencial para otro dato, en cascada se borre el segundo para no ocupar memoria alpepe.

**SQL Consultas**

En el delfin del workbrench hago clicl en +, le pongo el nombre que quiero y le doy ok.

La opción drop schema hace que se elimine el esquema que seleccioné.

Si quiero exportar un ERD a mi Laragon debo considerar ciertos aspectos:

1. El nombre de mi base de datos, en este caso: ejemplo\_primera clase
2. Voy a database en workbench y clicko en forward engineer y ahí pide stored connection, le doy siguiente dos veces y luego debo tener seleccionadas las tablas como mínimo. Se le puede dar continuar o se puede guardar el archivo sql para usarlo después. Hay dos maneras de ejecutar, una copiando el contenido o sino haciendo click en file y en open sql script (esto es en el coso nuevo que abri que es de laragon).
3. Se pone el cursor al principio sin seleccionar nada y se aprieta el rayo (lo que hace que se ejecuten todas las consultas)

Si en la columna de la izquierda no veo ningún archivo nuevo es porque tengo que actualizar, es un simbolito chiquito.

En la tarea esta el de twitter, cuando descargo, debo ir a la parte en donde estoy ejecutando laragon y ahí en file selecciono la opción de open sql script, después me aparece habilitado el rayo.

En la pestaña que dice query1 escribo la indicación que se me apetece y le doy rayito para obtener el dato.

***EXAMPLE***

Select \* users where id = 1 (Esto se traduce en: seleccionar a todos los usuarios donde el id sea igual a 1)

Si quiero seleccionar solo algunos campos y no todos (\*) se escribe como sigue:

Select id, first\_name from users

El punto y coma después de cada consulta permite que no me dé error.

El % significa cualquier cosa, cualquier cosa antes, cualquier cosa después. Para usar una base de datos, la misma debe estar en negrita y se hace seleccionando manualmente o escribiendo USE y el nombre en la especie de consola.

**CONCEPTOS BÁSICOS DE**

**INSERT**

Se puede hacer de dos maneras, editando directo desde la consola y le doy al botón de aplicar y me dará aviso de lo que se quiere añadir. El numero de columnas y el numero de valores debe coincidir para que esto funcione.

La segunda manera es desde la consolita esa usando INSERT

INSERT INTO nombre\_de\_la\_tabla (nombre\_columna1, nombre\_columna2)

VALUES('valor\_columna1', 'valor\_columna2');

Si me da paja escribir todo hago clic derecho en la tabla de mi interés y voy a hola copito keyword y luego insert statement, hago control V en mi consola y ahí me aparecen todos los campos en donde puedo cargar datos.

Cuando nosotros ponemos el ID como auto incremental no es necesario darle un valor en la consulta SQL porque automáticamente tendrá una numeración de acuerdo al índice previo

***Se comenta -- así. Los números pueden ir sin comillas pero no los string.***

Update

También hay dos formas de hacerse, directo en la tabla y apply o directo en la consola.

UPDATE nombre\_de\_la\_tabla SET nombre\_columna1 = 'un\_valor', nombre\_columna2 = 'otro\_valor', ... WHERE condición;

***IMPORTANTE: si la condición WHERE no se agrega a la instrucción UPDATE, los cambios se aplicarán a todos los registros de la tabla.***

UPDATE users SET first\_name = 'MARIA', last\_name = 'IVERSON' WHERE id = 3; También haciendo click derecho puede aparecer las opciones para el update, debo recordar escribir bien y después indicar el comando para que me muestre todos los datos pero actualizados.

DELETE

También hay dos formas de hacerse, directo en la tabla y apply o directo en la consola.

Si recibes un error con respecto a las ACTUALIZACIONES DE SEGURIDAD SQL, ejecuta el siguiente comando para que MySQL Workbench sepa que sabes lo que estás haciendo y quieres BORRAR cosas de la base de datos. SET SQL\_SAFE\_UPDATES = 0; escribir y apretar el rayito

El patrón de comando SQL para eliminar/remover registros es el siguiente:

DELETE FROM nombre\_de\_la\_tabla WHERE condición;

***IMPORTANTE: si la condición WHERE no se agrega a la instrucción DELETE, se borrarán todos los registros de la tabla.***

UNIR TABLAS

UNIMOS dos tablas en los identificadores que coincidan entre sí. Esto significa que no podemos UNIR tablas que no tengan una relación entre sí (por ejemplo, una a una, una a muchas, muchas a muchas). Una clave foránea en una tabla coincide directamente con una identificación en otra tabla.

Exportar la base de datos

Apretamos la opción de SERVER y después DATA EXPORT, asegurarse que sea de autocontenido; está determinado que vaya a una carpeta llamada dumps, por lo que debo cambiar ahí donde quiero que vaya y ahí le cambio tb el nombre del archivo si así lo deseo. ASEGURARSE clickear include créate schema. Seleccionar el esquema que queremos exportar y click en comenzar exportación