CFP Programador full-stack

Base de Datos

CFP Programador full-stack

Lenguaje de DDL - Creación de Esquemas

Introducción a SQL

- Para acceder y manipular los datos dentro de la base de datos se necesita un lenguaje declarativo
- **SQL** significa Lenguaje de consulta estructurado (Structured Query Language)
- **SQL** es un lenguaje de bases de datos global: cuenta con sentencias para definir datos, consultas y actualizaciones
- Sintaxis para:
  - Lenguaje de definición de datos: **DDL** (Data Definition Language)
  - Lenguaje de manipulación de datos: **DML** (Data Manipulation Language)

Lenguaje de definición de datos - DDL

- Permite crear y definir nuevas bases de datos, campos e índices:
  - **CREATE:** Crea nuevas tablas, campos e índices
  - **DROP:** Elimina tablas e índices
  - **ALTER:** Modifica las tablas agregando campos o cambiando la definición de los campos

Sintaxis para sentencias SQL

- ***{Alternativas}***, entre llaves se colocarán las palabras que tienen opciones o alternativas en la sentencia a la que pertenecen

[Opcional], entre corchetes se colocarán las palabras que son opcionales en la sentencia, es decir que pueden colocarse o pueden obviarse

Creación de Base de Datos

- CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] ***nombre_base_datos***
  - Esta sentencia sirve para crear una base de datos con un nombre específico
  - Para poder crear una base de datos, el usuario que la crea debe tener privilegios de creación asignados
- **IF NOT EXISTS** significa: SI NO EXISTE, por lo tanto, esto es útil para validar que la base de datos sea creada en caso de que no exista, si la base de datos existe y se ejecuta esta sentencia, se genera error
- CREATE SCHEMA o CREATE DATABASE son sinónimos

CREATE DATABASE IF NOT EXISTS complejo_de_cine

Creación de Tablas

- CREATE [TEMPORARY] TABLE [IF NOT EXISTS] nombre_de_tabla
- nombre_de_columna **tipo_de_dato** [NOT NULL | NULL] [DEFAULT valor_por_defecto][AUTO_INCREMENT] [UNIQUE | [PRIMARY] KEY
- [CONSTRAINT [nombre_relación] FOREIGN KEY (nombre_columna) REFERENCES nombre_de_tabla (nombre_columna)]
- [ON DELETE **opciones_de_referencia**]
- [ON UPDATE **opciones_de_referencia**]

Tipo de Datos Más Comunes

- BIT[(longitud)] → indica un valor booleano
- INT[(longitud)] → indica un número entero
- BIGINT[(longitud)]→ indica un número entero grande
- DOUBLE[(longitud,decimales)] → indica un número en punto flotante con precisión
- DATE → almacenar solo el día, mes y año
- TIME → almacena una hora con minutos y segundos
- DATETIME→ almacenar día, mes, año, hora, minuto y segundo
- CHAR[(longitud)] → indica un caracter
- VARCHAR(longitud) → indica una cadena de caracteres (string)

Creación de Tablas

Ejemplo de Película

```
CREATE TABLE IF NOT EXISTS `PELICULA` (
 `id_pelicula` INT NOT NULL,
 `anio_estreno` BIGINT,
 `disponible` BIT,
 `duracion` VARCHAR(45),
 `fecha_ingreso` DATETIME,
 PRIMARY KEY (`id_pelicula`))
```

Opciones de Referencias

- Las opciones de referencia sirven para establecer que se hará en casos de que se elimine o se actualice una fila de la tabla primaria que está siendo referenciada por una fila de la tabla secundaria
- **CASCADE**: Eliminar o actualizar la fila de la tabla primaria, y automáticamente eliminar o actualizar las filas coincidentes en la tabla secundaria
- **SET NULL**: Eliminar o actualizar la fila de la tabla primaria, y establecer la columna de clave externa (Foreign key) de la tabla secundaria a NULL. Si se especifica una SET NULL, hay que asegurarse que no se haya declarado la columna de la tabla secundaria como NOT NULL
- **RESTRICT:** Rechaza la operación de eliminación o actualización en la tabla primaria
- **SET DEFAULT:** Para una operación de eliminación o actualización en la tabla primaria se establecerá un valor por defecto para la tabla secundaria

Definición de Opcionalidades

- Las opciones **NOT NULL | NULL**, sirven para especificar si dicha columna puede aceptar valores nulos: NULL, o si no puede guardar valores nulos: NOT NULL
- También se puede de manera opcional, indicar un valor por defecto **DEFAULT** para una columna
- **AUTO_INCREMENT:** se refiere a un valor AUTO INCREMENTAL; sirve para aquellas columnas con valores que numéricos enteros donde se necesita que dicho valor se incremente en uno por cada fila insertada en la tabla. Se utiliza muy frecuentemente en las claves primarias
- **UNIQUE:** sirve para indicar que una columna en la tabla no puede tener valores repetidos, debe ser UNICA. No pueden existir dos filas en la tabla que tengan el mismo valor para un atributo definido como UNIQUE

Definición de Opcionalidades

- **PRIMARY KEY:** sirve para especificar que una columna es clave primaria de una tabla. Si una columna es clave primaria implica que sus valores no deben repetirse (UNIQUE)
- **CONSTRAINT:** significa RESTRICCIÓN y se le asigna un nombre para identificarla; dicho nombre funciona como clave que identifica unívocamente a cada CONSTRAINT que exista en la base de datos, por lo que toda CONSTRAINT debe tener un nombre no repetido; luego se indica con la palabra FOREIGN KEY, cuál es la columna de la tabla que funciona como clave foránea, indicando a continuación a cuál tabla y a cuál columna de dicha tabla hace referencia la clave foránea con la palabra REFERENCES

Alteración de Tabla

- Las acciones que se pueden hacer sobre una **Tabla** mediante la alteración son:
- **ADD:** COLUMN, INDEX, KEY, CONSTRAINT, FULLTEXT
- **CHANGE:** COLUMN
- **MODIFY:** COLUMN
- **DROP:** COLUMN. PRIMARY KEY, INDEX, KEY, FOREING KEY
- **DISABLE:** KEYS
- **ENABLE:** KEYS
- **RENAME:** INDEX, KEY, TO, AS
- **ORDER BY**

Alteraciones de Tabla

Ejemplo

Para cambiar el nombre de una tabla de t1 a t2

ALTER TABLE t1 RENAME t2;

Alteración de Tabla

- Para cambiar los tipos de datos de sus atributos, suponiendo que tenemos dos atributos **a** es INTEGER y **b** es CHAR con longitud 10: CHAR (10), cambiaremos a como TINYINT sin aceptar nulos y cambiaremos b por el nombre **c** y con longitud 20:

```
ALTER TABLE t2 MODIFY a TINYINT NOT NULL, CHANGE
b c CHAR(20);
```

Alteración de Tabla

- Para agregar una nueva columna d, de tipo DATETIME:

```
ALTER TABLE t2 ADD d DATETIME;
```

- Para agregar un nuevo índice en una columna d

```
ALTER TABLE t2 ADD INDEX (d);
```

Alteración de Tabla

- Para eliminar la columna c:

```
ALTER TABLE t2 DROP COLUMN c;
```

- Para agregar una nueva columna auto incremental y clave primaria, de tipo entera llamada e:

```
ALTER TABLE t2 ADD e INT NOT NULL
AUTO_INCREMENT, ADD PRIMARY KEY (e);
```

Borrar Tabla

- **DROP TABLE** remueve una o más tablas. Para poder eliminar tablas, el usuario que ejecuta la sentencia debe tener privilegios de **DROP** para cada tabla que quiera eliminar
- Con este comando se elimina todos los datos de la tabla, tanto la definición como las filas de datos de la misma. Si no existe la tabla que se desea borrar MySQL devuelve error, por eso es útil utilizar la opción IF EXISTS en caso de que exista la tabla la elimina

DROP [TEMPORARY] TABLE [IF EXISTS]

nombre_tabla [,nombre_tabla] ...

Creación de Índices

Un índice es una estructura de disco asociada con una tabla

o una vista que acelera la recuperación de filas de la tabla

Un índice contiene claves generadas a partir de una o varias

columnas de la tabla. Dichas claves están almacenadas en una estructura que permite que SQL Server busque de forma rápida y eficiente la fila o filas asociadas a los valores de cada clave

CREATE INDEX *nombre_indice*

ON *nombre_tabla* (*col_name* [(*length*)] [ASC | DESC])

[*opcion_algoritmo* | *opcion_bloqueo*] ...

Tipos de Índices

- Una tabla puede contener los siguientes tipos de índices:
- Agrupado
  - Los índices agrupados ordenan y almacenan las filas de los datos de la tabla de acuerdo con los valores de la clave del índice. Son columnas incluidas en la definición del índice
  - La única ocasión en la que las filas de datos de una tabla están ordenadas es cuando la tabla contiene un índice clúster. Cuando una tabla tiene un índice clúster, la tabla se denomina tabla agrupada. Si una tabla no tiene un índice clúster, sus filas de datos están almacenadas en una estructura sin ordenar denominada montón

Tipos de Índice

- No agrupado
  - Tienen una estructura separada de las filas de datos. Contienes los valores de clave de índice no agrupado y cada entrada de valor de clave tiene un puntero a la fila de datos que contiene el valor clave
  - El puntero de una fila hacia una fila de datos se denomina localizador de fila. La estructura del localizador de filas depende de si las páginas de datos están almacenadas en un montón o en una tabla agrupada
  - Se puede agregar columnas sin clave al nivel hoja de un índice no agrupado con el fin de eludir los límites existentes para las claves de índice, 900 bytes y columnas de 16 claves, así como para ejecutar consultas indizadas y totalmente cubiertas

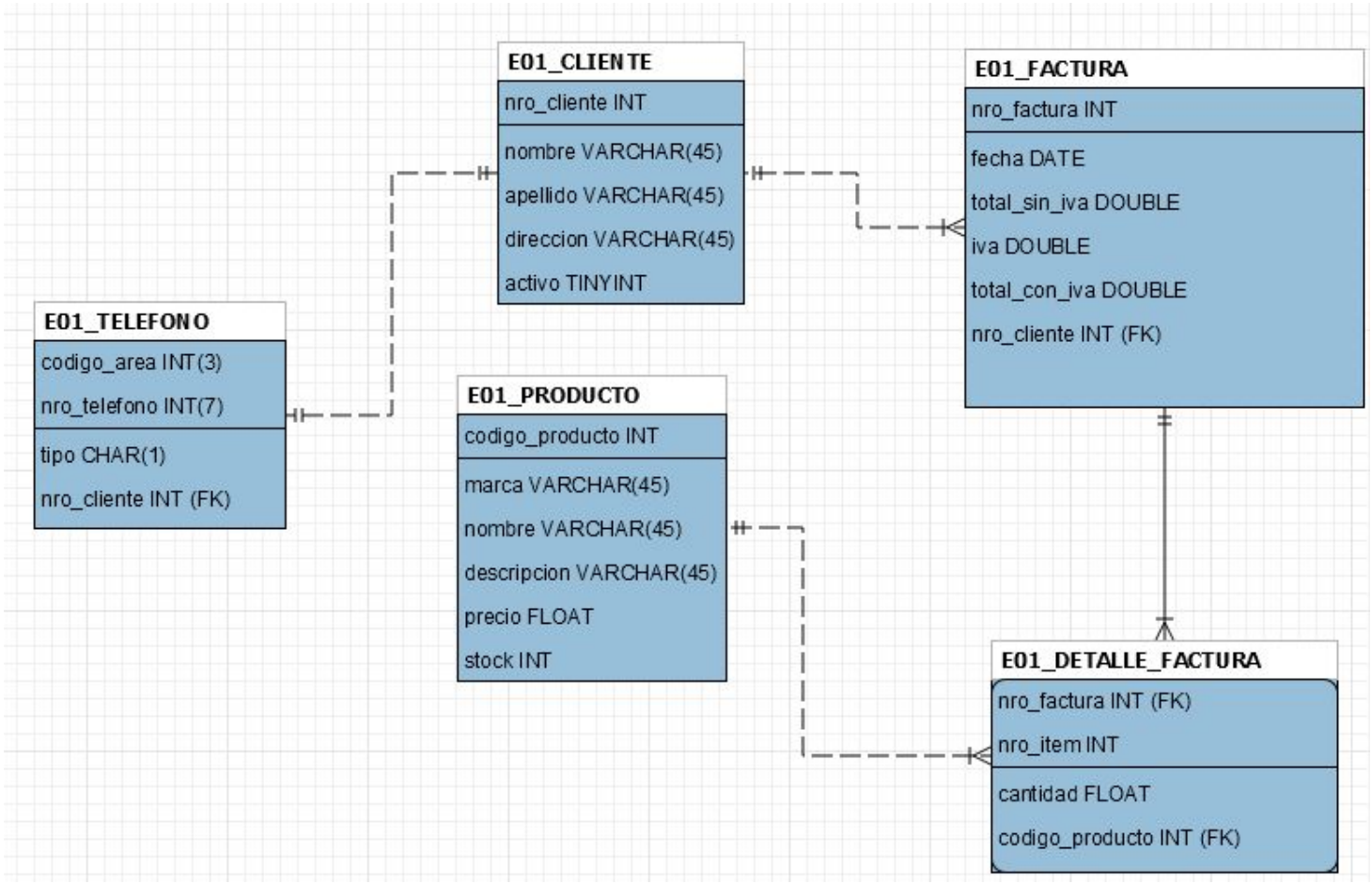
Base de Datos

CFP Programador full-stack

Lenguaje de DDL (Ejercicios)

Creación de Esquemas

Ejemplo Facturación



Creación de Esquemas

Ejemplo Facturación

- Dado el Modelo de Entidad-Relación de la figura anterior, crear las siguientes tablas con DDL:
  - E01_TELEFONO
  - E01_CLIENTE
  - E01_PRODUCTO
  - E01_FACTURA
  - E01_DETALLE_FACTURA

Base de Datos

CFP Programador full-stack

Lenguaje de DLL (Repaso)

Lenguaje de Consulta – Creación de Esquemas

Repaso

- La definición de los datos se realiza a través de las sentencias de DDL (Data Definition Language) del SQL
- Sus comandos permiten definir la semántica del esquema relacional: qué tablas o relaciones se establecen, sus dominios, asociaciones, restricciones, etc
- Las tablas son indentificadas unívocamente por sus nombres y contienen filas y columnas
- Una tabla en una base de datos relacional es similar a una tabla en papel, posee filas y columnas

Lenguaje SQL

Algunas funciones del estándar SQL son:

- DDL
  - **Definición de datos:**
    - **Creación de tablas (CREATE)**
    - **Modificación de tablas (ALTER)**
    - **Eliminación de tablas (DROP)**
- DML
  - Consulta de datos
    - Selección (SELECT)
  - Actualización de los datos
    - Inserción (INSERT)
    - Actualización (UPDATE)
    - Eliminación (DELETE)