INTEGRIDAD

ANALIZAR CONCEPTOS DE INTEGRIDAD PARA ESTABLECER REGLAS Y RESTRICCIONES PARA GUARDAR CON EXACTITUD LA INFORMACIÓN EN LOS SISTEMAS DE BASES DE DATOS.

INTEGRIDAD

La integridad de la base de datos, que se refiere a la consistencia y validez de los datos almacenados, se expresa mediante determinadas condiciones o restricciones que se deben cumplir. Pues bien, el sistema gestor de la base de datos se encarga de asegurar que estas condiciones se cumplan. Esto se debe a que en una base de datos no solo se almacenan los datos propiamente dichos, sino también la semántica de los mismos.

INTEGRIDAD

Una condición o limitación que se aplica a un conjunto particular de datos usualmente se denomina control de integridad o restricción.

Por ejemplo, los valores de las horas trabajadas en una semana podrían limitarse a 60 horas, o menos. Los controles de integridad se diseñan para minimizar las inconsistencias de los datos, ocasionadas cuando los usuario o los programas de aplicación cometen errores en la entrada de datos o cambiando los datos de la base de datos.

Restricciones de Integridad

Existen cinco diferentes tipos de restricciones: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY y CHECK.

En SQL, las restricciones UNIQUE y PRIMARY KEY se consideran restricciones únicas, y las restricciones de FOREIGN KEY se consideran como restricciones referenciales.

USO DE RESTRICCIONES

NOT NULL

En el momento que se requiere que un campo sea obligatorio, se debe utilizar dicha restricción.

Nulo significa que un valor no está definido o no se conoce. Esto no es lo mismo que cero, en blanco, una cadena vacía o un valor predeterminado. En lugar de ello, indica que un valor de dato está ausente.

NOT NULL

La restricción NOT NULL sólo puede utilizarse como una restricción de columna. La aplicación de una restricción NOT NULL es un proceso muy sencillo. Simplemente utilice la siguiente sintaxis cuando se crea una definición de columna:

<nombre de columna> { <tipo de datos>} NOT NULL

Ejemplo

nombreAlumno VARCHAR(50) NOT NULL

CHECK

Cuando se requiera verificar la información que va a contener un campo, se debe usar esta restricción ya que permite validar la información que va a contener ese campo según la restricción CHECK.

CHECK

La cláusula **check** de SQL permite restringir los dominios de maneras poderosas que no permiten la mayor parte de los sistemas de tipos de los lenguajes de programación. Concretamente, la cláusula **check** permite al diseñador del esquema especificar un predicado que debe satisfacer cualquier valor asignado a una variable cuyo tipo sea el dominio.

Una restricción CHECK permite especificar qué valores se pueden incluir en una columna. Se puede definir un rango de valores (por ejemplo, entre 10 y 100), una lista de valores (por ejemplo, blues, jazz, pop, country), o una serie de otras condiciones que restringen exactamente qué valores se permiten en una columna.

CHECK

```
CREATE TABLE departamentos
(id_departamento number(4),
id_usuario int, Ciudad varchar2(40),
CONSTRAINT chk_id CHECK (id_usuario BETWEEN 0 and 10000
));
```

UNIQUE

Se utiliza cuando se necesita que un campo contenga valores únicos, ósea valores que sean significativos que identifique a cada fila de todas las demás

UNIQUE

La restricción UNIQUE permite exigir que una columna o conjunto de columnas contengan valores únicos, valores significativos que sean diferentes de todas las demás filas en la misma tabla.

[CONSTRAINT <nombre de la restricción>]

UNIQUE (<nombre de columna> [{, <nombre de columna>}…]

Ejemplo

CONSTRAINT unq_noControl UNIQUE (noControl)

PRIMARY KEY

Solo se puede definir un primary key en una tabla, esta restricción al igual que la UNIQUE se utiliza cuando se requiere que un campo contenga valores únicos.

PRIMARY KEY

Una *clave de candidato* es un conjunto de una o más columnas que identifican de forma exclusiva a cada fila.

PRIMARY KEY tienen dos restricciones:

- Una columna que se define con una restricción PRIMARY KEY no puede contener valores nulos
- Sólo una restricción PRIMARY KEY puede definirse para cada tabla.

```
CREATE TABLE cd_artistas
( id_artista NUMBER,
nombre_artista VARCHAR2(60),
id_agencia INT,
CONSTRAINT PK_id_artista PRIMARY KEY ( id_artista));
```

Clave de candidato

ID_ARTISTA: INT	NOMBRE_ARTISTA: VARCHAR(60)	ID_AGENCIA: INT
10001	Jennifer Warnes	2305
10002	Joni Mitchell	2306
10003	William Ackerman	2306
10004	Kitaro	2345
10005	Bing Crosby	2367
10006	Patsy Cline	2049
10007	Jose Carreras	2876
10008	Placido Domingo	2305
10009	Luciano Pavarotti	2345

DEFAULT

La restricción Default es usada para insertar valores predeterminados en un atributo. Este es asignado a un nuevo registro, si no se ha especificado otro valor.

```
CREATE TABLE usuarios
(
id_usuario int NOT NULL,
nombre varchar(255) NOT NULL,
apellidos varchar(255), domicilio varchar(255),
ciudad varchar(55) DEFAULT 'La Paz'
);
```

Restricciones

Hasta este punto, los tipos de restricciones que se analizaron tienen que ver principalmente con la garantía de la integridad de los datos dentro de una tabla. La restricción NOT NULL evita el uso de valores nulos dentro de una columna, y las restricciones UNIQUE y PRIMARY KEY garantizan la singularidad de los valores dentro de una columna o conjunto de columnas. Sin embargo, la restricción FOREIGN KEY es diferente en el sentido de que se ocupa de cómo los datos en una tabla hacen referencia a los datos en otra tabla, que es la razón por la que se conoce como una *restricción referencial* (en relación con otra tabla).

FOREIGN KEY

Esta restricción se encarga de mantener la integridad relacional de los datos, en el sentido de que si el valor de un campo de una tabla hace referencia a otro campo de una segunda tabla que contiene ese mismo valor, no se podrá eliminar ese valor ya que esta referenciado y relacionado en otra tabla. Sin una restricción foreign key se pierde la integridad relacional de cierto conjunto de datos que se encuentren relacionados.

FOREIGN KEY

Cuando se crea una restricción FOREIGN KEY, se deben seguir varias directrices:

- Las columnas referenciadas se deben definir con una restricción UNIQUE o una PRIMARY KEY.

- Una restricción FOREIGN KEY se puede crear como una restricción de tabla o una restricción de columna. Si se crea la clave foránea como una restricción de columna, se puede incluir sólo una columna. Si se crea la clave foránea como una restricción de tabla, se pueden incluir una o más columnas.

FOREIGN KEY

- La clave foránea en la tabla de referencia debe incluir el mismo número de columnas que sean referenciadas, y las columnas de referencia deben cada una configurarse con los mismos tipos de datos que su contraparte de referencia. Sin embargo, las columnas de referencia no deben tener necesariamente los mismos nombres que las columnas referenciadas.

FOREIGN KEY

TITULOS_CD

ID_TITULO_CD: INT	TITULO_CD: VARCHAR(60)	ID_EDITOR: INT
11001	Famous Blue Raincoat	5422
11002	Blue	5402
11003	Past Light	5412
11004	Kojiki	5409
11005	That Christmas Feeling	5403
11006	Patsy Cline: 12 Greatest Hits	5403

EDITORES_CD

ID_EDITOR: INT	NOMBRE_COMPAÑIA: VARCHAR(60)
5403	MCA Records
5402	Reprise Records
5409	Geffen
5412	Windham Hill Records
5422	Private Music

FOREIGN KEY

[CONSTRAINT <nombre de la restricción>]
FOREIGN KEY (<columna referenciada> [{, <columna referenciada>}...])
REFERENCES <tabla referida> [(columnas de referencia>)]
[< acción referencial desencadenada >]

Ejemplo

CONSTRAINT fk_tituloCD_editor FOREIGN KEY (id_Editor)
REFERENCES Editores_cd (id_Editor)

El uso de restricciones tiene una relevancia muy importante en el diseño de una base de datos, por que no solo establecen seguridad, sino que también, garantiza la integridad de la información contenida en la misma.

ACCIÓN REFERENCIAL DESENCADENADA

La cláusula final en la sintaxis de la restricción FOREIGN KEY es la cláusula opcional <acción referencial desencadenada>. La cláusula permite definir qué tipos de acciones se deben tomar cuando se intenta actualizar o eliminar datos desde columnas referenciadas (si ese intento causa una violación de los datos en las columnas de referencia).

ACCION REFERENCIAL DESENCADENADA

Esto se puede aclarar echando un vistazo a la sintaxis para la cláusula de <acción referencial desencadenada>:

ON UPDATE <acción referencial> [ON DELETE <acción referencial>]
| ON DELETE <acción referencial> [ON UPDATE < acción referencial>]

<acción referencial>::=
CASCADE | SET NULL | SET DEFAULT | RESTRICT | NO ACTION

ACCION REFERENCIAL DESENCADENADA

Como se observa de la sintaxis, se puede definir una cláusula ON UPDATE, y una cláusula ON DELETE, o ambas, y se pueden definir en cualquier orden.

Para cada una de estas cláusulas se puede escoger una de cinco acciones referenciales:

- RESTRICT
- CASCADE
- SET NULL
- NO ACTION
- SET DEFAULT

RESTRICCIÓN DE CLAVE EXTERNA

El comportamiento por defecto de una restricción de clave externa es impedir un cambio en la base de datos como consecuencia de una sentencia DELETE o UPDATE, si esta trajese como consecuencia un fallo de la integridad referencial.

personas		
persona_id	**persona_nom**	**ciudad_id**
1	Pedro	1
2	Santiago	2
3	Juan	3
4	Andrés	1

ciudades	
ciudad_id	**ciudad_nom**
1	Galilea
2	Betsaida
3	Patmos
4	Jerusalén

* La columna **ciudad_id** de la tabla **personas** es llave foránea.

* La referencia es la columna **ciudad_id** de la tabla **ciudades**.

RESTRICT

Es el comportamiento por defecto, si se trata de actualizar o eliminar datos en las columnas referenciadas que causan una violación de clave foránea, se previene la ejecución de esta acción. Los datos en las columnas de referencia nunca pueden violar la restricción FOREIGN KEY, ni siquiera temporalmente.

RESTRICT

DELETE FROM ciudades WHERE ciudad_id=4;

personas		
persona_id	persona_nom	ciudad_id
1	Pedro	1
2	Santiago	2
3	Juan	3
4	Andrés	1

RESTRICT

ciudades	
ciudad_id	ciudad_nom
1	Galilea
2	Betsaida
3	Patmos
~~4~~	~~Jerusalén~~

* Si intentamos borrar una fila en **ciudades** que tenga valores en **personas** tendremos el error **foreign key constraint fails**

* Si no hay registro relacionado se borrará

RESTRICT

DELETE FROM ciudades WHERE ciudad_id=1;

> Cannot delete or update a parent row: a foreign key constraint fails (db.personas, CONSTRAINT personas_ibfk_1 FOREIGN KEY (ciudad_id) REFERENCES ciudades (ciudad_id))

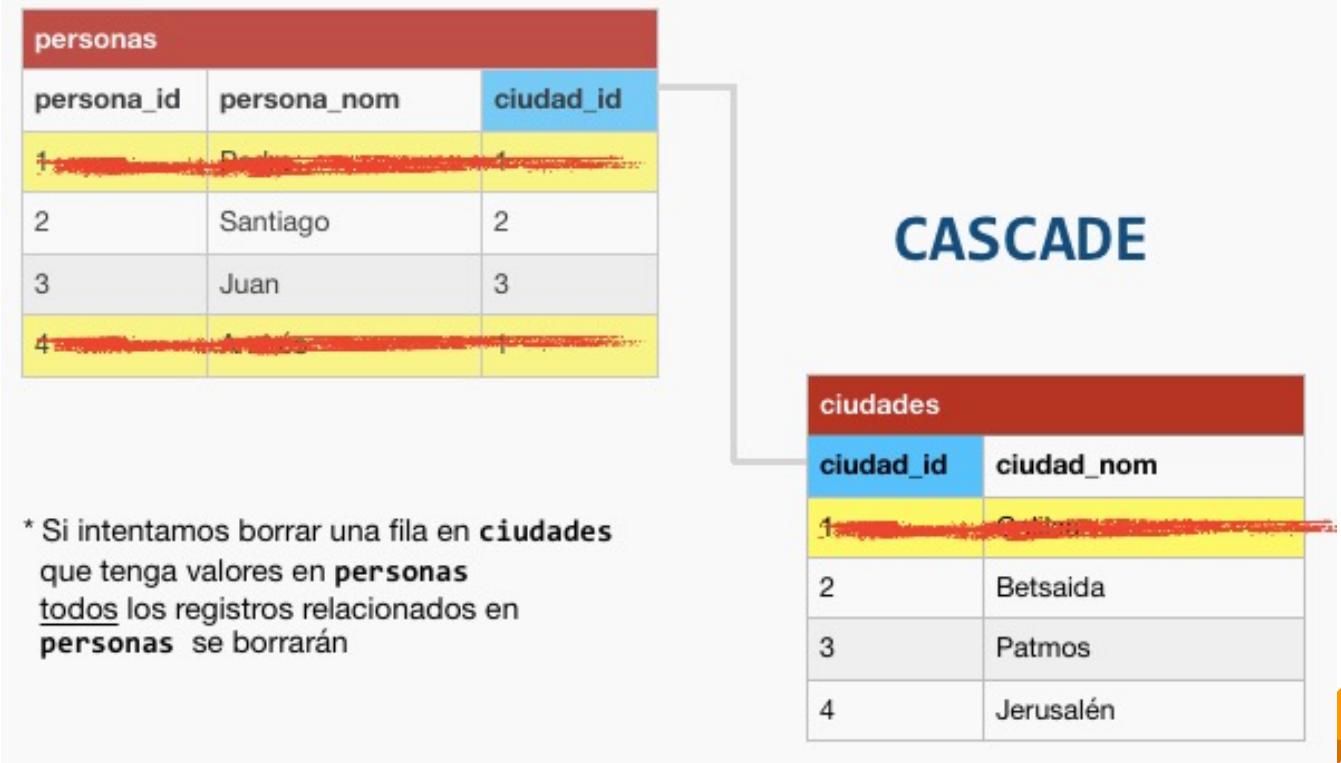
Porque el DELETE viola la restricción. Si la fila 1 de ciudades se borrase, las filas 1 y 4 de personas quedarían ***huérfanas***, o sea, sin relación en la tabla ciudades.

CASCADE

Borra los registros de la tabla dependiente cuando se borra el registro de la tabla principal (en una sentencia DELETE), o actualiza el valor de la clave secundaria cuando se actualiza el valor de la clave referenciada (en una sentencia UPDATE).

CASCADE

DELETE FROM ciudades WHERE ciudad_id=1;



SET NULL

Si se utiliza SET NULL y los datos se actualizan o eliminan en las columnas referenciadas, los valores en las columnas de referencia correspondientes se establecen como nulos. Los valores nulos tienen que respaldarse en las columnas de referencia para que esta opción funcione.

SET NULL

DELETE FROM ciudades WHERE ciudad_id=1;

```
CREATE TABLE personas (
    persona_id INT NOT NULL AUTO_INCREMENT
    CONSTRAINT pk_persona PRIMARY KEY (persona_id),
    persona_nom VARCHAR(70), ciudad_id INT NOT NULL,
    CONSTRAINT fk_persona_ciudad FOREIGN KEY
    fk_ciudad(ciudad_id) REFERENCES ciudades(ciudad_id) ON
    DELETE SET NULL )
```

Cannot add foreign key constraint

NO ACTION

Una palabra clave del SQL estándar. En MySQL, equivalente a RESTRICT. El servidor MySQL rechaza la operación de eliminación o actualización para la tabla primaria si hay un valor de clave externa relacionado en la tabla referenciada. Algunos sistemas de base de datos tienen verificaciones diferidass, y NO ACTION es un verificación diferida. En MySQL, las restricciones de clave externa se comprueban inmediatamente, por lo que **NO ACTION** es igual que **RESTRICT**.