Bases de datos

U.T.5

Tratamiento de los datos con SQL

Tratamiento de los datos

- Herramienas gráficas
- Sentencia INSERT
- Sentencia INSERT y SELECT
- Sentencia UPDATE
- Sentencia DELETE
- Sentencia UPDATE y DELETE con Subconsultas
- Transacciones
- Vistas, usuarios y privilegios

El lenguaje DML

- Las sentencias DML del lenguaje SQL son consultas (query) o peticiones a la base de datos del tipo:
  - ***SELECT***

    Extrae información de la base de datos
  - ***INSERT***

    Insertar uno o varios registros en alguna tabla.
  - ***DELETE***

    Borra registros de una tabla
  - ***UPDATE***

    Modifica registros de una tabla.

La sentencia INSERT

- Permite insertar una fila en una tabla, es decir, añadir un registro de información a una tabla.

INSERT [INTO] nombre_tabla [(columna, ...)]
VALUES ({expr | DEFAULT}, ...)

- Si no se especifican las columnas, los valores se insertan según el orden de las columnas en la definición de la tabla.

La sentencia INSERT extendida

- Sólo existe para algunos gestores de bd como MySQL:

  INSERT [INTO] nombre_tabla [(columna, ...)]
  VALUES ({expr | DEFAULT}, ...), (...),...

- Los puntos suspensivos indican que se puede repetir varias veces la lista de valores.

```
insert into vehiculos (Matricula, Modelo, Marca)
values ('4123 BFH','Ibiza','Seat'),
       ('1314 FHD','Toledo','Seat'),
       ('3923 GJS','Leon','Seat')
```

INSERT y SELECT

- Utiliza un conjunto de datos obtenidos con SELECT para insertarlos en la tabla posteriormente.

```
INSERT [INTO] nombre_tabla [(columna, ...)]
SELECT ... FROM ...
```

- La sentencia SELECT debe devolver tantas columnas como columnas tenga la tabla donde se introduce la información.
- La sentencia SELECT puede ser todo lo compleja que sea necesario.

La sentencia UPDATE

- Permite modificar el contenido de cualquier columna y de cualquier fila de una tabla.

```
UPDATE  nombre_tabla
SET nombre_col1=expr1 [,nombre_col2=expr2]...
[WHERE filtro]
```

- La actualización se realiza dando a las columnas col1, col2, ... los valores expr1, expr2, ... respectivamente.

```
UPDATE jugadores SET Nombre_equipo='Knicks'
 WHERE Nombre = 'Pau Gasol'
```

La sentencia UPDATE

- Se actualizan todas las filas seleccionadas por el filtro
- Si se omite el filtro se modifican todos los registros de la tabla.
- Es posible cambiar más de una columna a la vez:

```
UPDATE jugadores SET Nombre_equipo='Knicks', Peso=210
 WHERE Nombre = 'Pau Gasol'
```

La sentencia DELETE

- Elimina filas de una tabla.

```
DELETE  FROM  nombre_tabla
[WHERE  filtro]
```

- Borrar los registros seleccionados por el filtro.

```
DELETE  FROM jugadores
 WHERE Nombre = 'Pau Gasol'
```

- Si se omite el filtro se borran todos los registros de la tabla.

UPDATE y DELETE con subconsultas

- Se puede actualizar o borrar registros de una tabla filtrando a través de una subconsulta.

```
DELETE  FROM Empleados
 WHERE CodigoEmpleado NOT IN (SELECT CodigoEmpleadoRepVentas
                                FROM Clientes)
AND Puesto='Representante Ventas'
```

- Hay gestores que no permiten realizar cambios en la tabla que se está leyendo en la subconsulta.

Actualización de registros con relaciones

- No siempre se pueden borrar o modificar los datos de una tabla cuando existen claves foráneas que relacionan dichas tablas.
- El comportamiento de las actualizaciones en estos casos dependen de la definición de la tabla

```
CREATE TABLE nombre-tabla
…..
definición de referencia:
        REFERENCES nombre_tabla (nombre-columna)
                        [ON DELETE opcion_referencia]
                        [ON UPDATE opcion_referencia]
 opcion_referencia:
        CASCADE | SET NULL | NO ACTION
```

Borrado y actualización de registros con relaciones

- ON DELETE/UPDATE NO ACTION
  - Si se intenta modificar o borrar un registro con otros registros relacionados, no se permitirá dicho cambio.

- ON DELETE/UPDATE CASCADE
  - Si se intenta modificar o borrar un registro con otros registros relacionados, se cambiarían o borrarían todos los registros relacionados con él en las otras tablas.

Borrado y actualización de registros con relaciones

- ON DELETE/UPDATE SET NULL
  - Si se intenta modificar o borrar un registro con otros registros relacionados, se cambiarían todos los registros relacionados con él en las otras tablas, insertando el valor ***null*** en todos los campos que aparece la clave eliminada o cambiada.

[Ejercicios páginas 195 y 196]

Transacciones

- Un SGBD actualiza múltiples datos a la vez en una transacción.
- Una transacción es un conjunto de sentencias SQL que se tratan como una sola instrucción (atómica).
  - Trabajar con transacciones es esencial para mantener la integridad de los datos en la BD.
  - Garantiza la atomicidad de la operación: ***O se hacen todas o no se hace ninguna.***

Transacciones

- Una transacción puede ser :
  - confirmada (***commit***): todas las operaciones individuales se ejecutaron correctamente.
  - o abortada (***rollback***): a mitad de la ejecución del grupo de instrucciones hubo algún problema que impide la ejecución completa del mismo.

Transacciones

- Modos de ejecución:
  - AUTOCOMMIT=ON, cada comando SQL ejecutado es considerado como una transacción (por defecto).
  - AUTOCOMMIT=OFF:
    - Activa las transacciones de múltiples sentencias
    - Todos los comandos SQL deben terminarse con COMMIT/ROLLBACK
- Formas de comenzar una transacción:

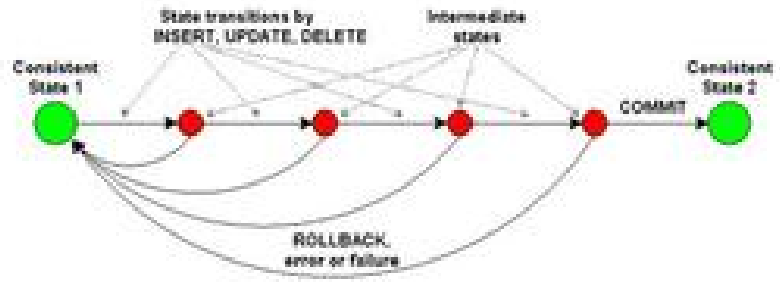
  SET AUTOCOMMIT=ON

  START TRANSACTION

  BEGIN WORK

Transacciones

Transacciones (MySQL)



Acceso concurrente a los datos

- Es el acceso al mismo dato desde dos transacciones distintas.
- Puede ocasionar algunos problemas:
  - Lectura sucia
    - Se leen datos no confirmados
  - Lectura no repetible
    - Diferentes lecturas del mismo dato
  - Lectura fantasma
    - Aparición de nuevos datos entre el inicio y el final de la lectura.

Acceso concurrente a los datos

- El SGBD puede bloquear conjuntos de datos a distintos niveles para evitar problemas.
- Los *niveles de aislamiento (isolation levels)* definen qué cambios hechos por una transacción son visibles a otras transacciones:
  - Read Uncommited
  - Read Commited
  - Repeateable
  - Serializable
- Algunas transacciones pueden quedar interbloqueadas dependiendo del nivel de aislamiento.

Acceso concurrente a los datos

El nivel de aislamiento por defecto *(repeateable read) se* puede cambiar mediante el comando

SET TRANSACTION ISOLATION LEVEL {

READ UNCOMMITED |

READ COMMITED |

REPEATABLE READ |

SERIALIZABLE }

El nivel de aislamiento con el que actualmente está trabajando nuestro servidor:

SHOW VARIABLES LIKE 'tx_isolation'.

Vistas

- Tabla sin contenido, totalmente virtual, que devuelve las filas resultado de una consulta SQL.
- Es una medida de seguridad ya que proporciona un acceso controlado a determinadas filas y columnas de las tablas de origen.
- Presenta menor tiempo de ejecución que el de una consulta ordinaria por su compilación única.
- En la tabla ***information_schema.key_column_usage*** podemos consultar las restricciones de clave foránea de una base de datos.

Vistas

- Sintaxis para crear una vista:

CREATE [OR REPLACE] VIEW [esquema.]nombre_vista
[(lista-columnas)] AS sentencia-select

```
CREATE VIEW VistaPedidos  (CodigoPedido,Cliente,Total)
AS SELECT CodigoPedido, NombreCliente, SUM(Cantidad*PrecioUnidad)
FROM Clientes
        INNER JOIN Pedidos ON Clientes.CodigoCliente=Pedidos.CodigoCliente
        INNER JOIN DetallePedidos ON
                        Clientes.CodigoCliente=DetallePedido.CodigoCliente
GROUP BY CodigoPedido;
```

- Eliminar una vista:

DROP VIEW [esquema.]nombre_vista

Usuarios

- El SGBD permite crear cuentas de usuario que permitan acceder a ciertos objetos con un nivel determinado de privilegios.

- La información de los usuarios se almacena en la tabla ***mysql.user***

Usuarios

- Sintaxis para crear una usuario:

  CREATE USER nombre_usuario IDENTIFIED BY 'password' [opciones]

```
CREATE USER antonio IDENTIFIED BY 'O99238KJKA';
```

- Eliminar usuarios:

  DROP USER nombre_usuario [CASCADE]

- No existe una sentencia ALTER USER en MySQL. Para modificar un usuario hay que modificar la tabla mysql.user

  [Ejemplo página 207]

Usuarios

- Renombrar un usuario

  RENAME USER nombre_antiguo TO nombre_nuevo

- Cambiar la contraseña de un usuario

  SET PASSWORD

```
RENAME USER antonio@localhost TO antonioSanchez@localhost;
SET PASSWORD for antonioSanchez@localhost=PASSWORD("nueva-passw");
```

Privilegios

- Un usuario puede obtener privilegios para manipular objetos de una base de datos con el comando GRANT.
- Igual se pueden denegar permisos con el comando REVOKE.
- La información de los privilegios se almacena en la tablas tablas ***mysql.user***, ***mysql.db***, ***mysql.host***, ***mysql.tables_priv*** y ***mysql.columns_priv***.

Privilegios

- Sintaxis del comando GRANT

GRANT tipo_privilegio [(columnas)] ON tablas TO usuario
IDENTIFIED BY [password] WITH opción

- tipo de privilegio

ALL PRIVILEGES, ALTER, SELECT, DELETE, UPDATE, EXECUTE, ...

- expresión de las tablas

nombre_tabla|* | *.* | bd.*| bd.tabla

- opción

GRANT OPTION, max_queries_per_hour, ...

Privilegios

- Se pueden crear usuarios a la vez que se conceden los privilegios a dicho usuario.

```
GRANT SELECT (NombreCliente, Telefono, Ciudad)
ON Clientes
TO antonio@localhost IDENTIFY BY contraseña
```

- La creación de usuarios mediante el comando GRANT ha quedado obsoleta.

http://mysqlserverteam.com/removal-and-deprecation-in-mysql-5-7/

- La sentencia REVOKE retira permisos a un usuario sobre un objeto.

REVOKE tipo_privilegio [(columnas)] ON ***tabla***
FROM usuario

Privilegios

- Crear un usuario con permiso sólo para conectarse a la BD

```
GRANT  USAGE ON Clientes  TO
antonio@localhost
```

- El comando que muestra los privilegios de un usuario:

  SHOW grants

Acceso remoto al servidor MySQL

- Crear un usuario en el servidor con los permisos necesarios:

```
CREATE USER antonio IDENTIFIED BY  '1234';
GRANT select ON jardinería.clientes  TO antonio
```

Abrir el puerto que utiliza MySQL en Windows (3306)

-> Firewall Windows –> Permitir programa -> Agregar puerto

- Instalar Workbench en el equipo cliente.
- Crea una conexión en workbench desde el equipo cliente con dicho usuario, añadiendo la IP del servidor al que conectamos.

```
antonio@DAW112
antonio@192.168.1.12
```

Backup de una base de datos MySQL

Backup de una base de datos MySQL

- Formas de llevar a cabo un backup de una base de datos MySQL:
  - **Workbench -> Data Export**
    - Seleccionar bases de datos y tablas a guardar.
    - Dump structure and data / Dump data only / Dump structure only
    - Export to Self-Contained File: Indicar la ruta del fichero destino del backup.
    - Include Create Schema

Backup de una base de datos MySQL

- Formas de llevar a cabo un Backup de una base de datos MySQL:
  - **phpMyAdmin -> Exportar**
    - Método de exportación: Personalizado / Rápido
    - Seleccionar la base de datos y sus tablas
    - Guardar salida a un fichero
    - Formato de salida: SQL, CSV, etc.
    - Opciones de formato y creación de datos

Backup de una base de datos MySQL

- Formas de llevar a cabo un Backup de una base de datos MySQL:
  - **Mediante el comando mysqldump:**
    - Permite crear copias de seguridad que pueden ser restaurados en distintos gestores de bases de datos.
    - Devuelve un fichero SQL con todas las sentencias necesarias para la restauración.

Backup de una base de datos MySQL

- **Ejemplos de uso del mysqldump:**
  - **Backup de una base de datos completa**

    mysqldump –h localhost –u user –p bbdd > backup.sql
  - **Backup de dos tablas de una base de datos**

mysqldump –h localhost –u user –ppassword bbdd tabla1 tabla2 > backup.sql

  - **Backup de dos bases de datos completas**

mysqldump –h localhost –u user –ppassword --databases bbdd1 bbdd2 > backup.sql

Backup de una base de datos MySQL

- **Ejemplos de uso del mysqldump:**
  - **Backup de todas las bases de datos**

mysqldump –h localhost –u user –p –all-databases > backup.sql

  - **Backup de una base de datos hospedada en un servidor externo con otro puerto**

mysqldump –h 197.125.47.89 –P 3310 –u user –ppassword bbdd > backup.sql

http://www.acens.com/wp-content/images/2013/02/white-paper-acens-backup-base-de-datos-mysql.pdf

RESTAURAR una copia de seguridad

- Formas de recuperar un backup de una base de datos MySQL:
  - **Workbench -> Data Import**
    - Seleccionar el fichero o carpeta de ficheros con el backup a restaurar.
      - Import from dump project folder
      - Import from Self-Contained File.

RESTAURAR una copia de seguridad

- Formas de llevar a cabo un Backup de una base de datos MySQL:
  - **phpMyAdmin -> Importar**
    - Elegir el fichero desde el que vamos a restaurar
    - Formato del fichero de backup: SQL, CSV, etc.

RESTAURAR una copia de seguridad

- Formas de recuperar un Backup de una base de datos MySQL:
  - **Mediante el comando mysql:**
    - Permite recuperar copias de seguridad para restaurar una base de datos guardada previamente con el comando ***mysqldump***.
    - Necesita un fichero SQL con un backup previo de la base de datos para la restauración.

mysql –user antonio -password nombreBD < copia_seguridad.sql