ESCUELA DE SISTEMAS Y TECNOLOGÍAS

Transparencias de **ANALISTA DE SISTEMAS**

Edición 2020 Materia: **Java Web**

TEMA: MySQL

Agenda

- Introducción a MySQL
- Base de Datos en MySQL
- Datos en MySQL
- Consultas en MySQL
- Procedimientos Almacenados en MySQL

Introducción MySQL (1)

- Para poder comenzar a trabajar se debe ejecutar en la shell el siguiente comando:

  mysql –h host –u usuario –p

- En el caso de las máquinas en el Instituto, como conectaremos con el servidor de BD local de cada máquina, no será necesario el parámetro host.

  mysql –u root –p

- Recuerde que los nombres de las tablas en algunos sistemas (como por ej. Linux), son case sensitive

Introducción MySQL (2)

- Los comandos SQL se pueden dividir en 3 categorías:
  1. Comandos para la definición de la BD, esquema y otros objetos (DDL – Data Definition Language).

     CREATE, ALTER, DROP, etc.

  2. Comandos para la manipulación de los datos que se encuentran en la BD (DML – Data Manipulation Language).

     INSERT, UPDATE, DELETE, SELECT, etc.

  3. Comandos para controlar el acceso a los diferentes objetos de la BD (DCL – Data Control Language).

     GRANT, REVOKE, etc

Base de Datos en MySQL (1)

- El primer comando para la definición de la BD se refiere a cómo crearla (el usuario debe tener permiso de root para ejecutarlo):

  CREATE DATABASE mibase;

- Para poner en uso una base de datos:

  USE mibase;

- Luego se agregarán tablas mediante el siguiente comando:

  CREATE TABLE mitabla (campo tipo restricciones,
  campo tipo restricciones, ...);

- Para ver las bases de datos existentes:

  SHOW DATABASES;

Base de Datos en MySQL (2)

- Algunos tipos de datos que pueden ser asignados a columnas de una tabla:

CHAR	TINYINT
VARCHAR	SMALLINT
BLOB	MEDIUMINT
TEXT	INTEGER
ENUM	BIGINT
SET	DATE
FLOAT	TIME
REAL	DATETIME
DOUBLE PRECISION	TIMESTAMP
NUMERIC	BIT
DECIMAL	

Base de Datos en MySQL (3)

- Luego de haber creado la BD según el diseño relacional deseado, se puede observar el resultado:

  SHOW TABLES;

- Para ver la descripción de una de las tablas:

  DESCRIBE nombreTabla;

- Para hacer un backup de una BD (desde consola):

  mysqldump –u root –p mibase > miScript.txt

- Para eliminar toda la BD (desde mysql):

  DROP DATABASE mibase;

Base de Datos en MySQL (4)

- Para restaurar una BD:

  - Antes de importar el script, debe crearse la BD vacía (desde mysql):

    CREATE DATABASE mibase;

  - Desde consola:

    mysql –u root –p --database=mibase < miScript.txt

Base de Datos en MySQL (5)

- Para modificar la estructura de una BD ya creada se utiliza el comando ALTER:

```
ALTER TABLE mitabla ADD columnaAgregar tipo;
ALTER TABLE mitabla CHANGE viejaCol newCol tipo;
ALTER TABLE mitabla MODIFY nombreCol tipoNuevo;
ALTER TABLE mitabla DROP columnaAQuitar;
```

- Por más información:
  - https://dev.mysql.com/doc/refman/5.7/en/alter-table.html
  - https://dev.mysql.com/doc/refman/5.7/en/alter-database.html

Datos en MySQL (1)

- **Cargar datos de archivo**: implica tener los datos guardados en un formato específico. Para utilizar el comando *LOAD* se necesita que el archivo se encuentre con los valores de cada columna de la tupla separados por tabuladores y para cada tupla un fin de línea que la separe de la siguiente:

```
LOAD DATA INFILE 'archivo.txt'
    INTO TABLE mitabla;
```

Datos en MySQL (2)

- **Cargar datos con comando**: el comando que permite insertar manualmente valores en la BD es INSERT:

  INSERT INTO tabla VALUES (valorCol1, valorCol2, ...);

- Los valores de tipo cadena de caracteres o fechas se deben ingresar entre comillas simples.
- Los valores de tipo numérico irán sin comillas.
- Para insertar un campo nulo basta con poner NULL.
- Si una columna es autogenerada debe colocarse NULL como valor.
- También pueden especificarse las columnas de destino luego de la palabra INTO, en cuyo caso sólo se proveerán valores para dichas columnas.

Datos en MySQL (3)

- Luego de tener los datos ya cargados, éstos pueden ser modificados mediante el comando UPDATE:

```
UPDATE mitabla
SET columna1 = expresion1,
columna2 = expresion2,
...
WHERE condición;
```

Datos en MySQL (4)

- Para eliminar filas de una tabla puede utilizarse el comando DELETE:

```
DELETE
FROM mitabla
WHERE condición;
```

Consultas en MySQL

- Para consultar datos puede utilizarse el comando SELECT:

```
SELECT columnas
FROM tablas
WHERE condición;
```

- En donde columnas, son las columnas de las tablas elegidas (separadas por coma) o un asterisco (*) y condición es lo que deben cumplir las tuplas de las tablas elegidas para formar parte del resultado de la consulta.

SP en MySQL (1)

- SP de 1 sentencia sin parámetros:

```
CREATE PROCEDURE ListarTodos()
    SELECT *
    FROM Empleados;
```

- SP de 1 sentencia con parámetro (de entrada):

```
CREATE PROCEDURE MayoresDe(IN pEdad INT)
    SELECT *
    FROM Empleados
    WHERE edad > pEdad;
```

SP en MySQL (2)

- Mostrar todos los SPs:

  SHOW PROCEDURE STATUS;
- Mostrar un SP:

  SHOW CREATE PROCEDURE MayoresDe;
- Invocar un SP sin parámetros:

  CALL ListarTodos();
- Invocar un SP con parámetros:

  CALL MayoresDe(18);
- Eliminar un SP:

  DROP PROCEDURE MayoresDe;
- Cambiar el carácter delimitador:

  DELIMITER //

SP en MySQL (3)

- Crear un SP de varias sentencias:

```
CREATE PROCEDURE EliminarEmpleado(IN pCedula BIGINT)
BEGIN
    # Eliminar primero los teléfonos:
    DELETE FROM Telefonos WHERE empleado=pCedula;
    # Eliminar luego el empleado:
    DELETE FROM Empleados WHERE cedula=pCedula;
END//
```

SP en MySQL (4)

- Crear un SP con parámetro de salida:

```
CREATE PROCEDURE ObtenerMayorSueldo(OUT mayor DOUBLE)
        SELECT MAX(sueldo)
        FROM Empleados
        INTO mayor;
```

- Definir una variable (en el script):

  SET @varMayor = 0;

- Invocar al SP:

  CALL ObtenerMayorSueldo(@varMayor);

- Mostrar el nuevo valor de la variable:

  SELECT @varMayor;

SP en MySQL (5)

- Crear un SP con parámetro de entrada/salida:

```
CREATE PROCEDURE PruebaIO(INOUT i INT)
    SET i=i+1;
```

- Definir una variable

  SET @variable = 10;

- Invocar al SP:

  CALL PruebaIO(@variable);

- Mostrar el nuevo valor de la variable:

  SELECT @variable;

SP en MySQL (6)

- Crear una función:

```
CREATE FUNCTION buscar(pCI BIGINT) RETURNS VARCHAR(50)
BEGIN
    DECLARE temp VARCHAR(50);
    SELECT nombre
    FROM Empleados
    WHERE cedula=pCI
    INTO temp;

    RETURN temp;
END//
```

- Mostrar todas las funciones

```
SHOW FUNCTION STATUS;
```

- Invocar a la función y mostrar resultado:

```
SET @variable = buscar(2);
SELECT @variable
```

SP en MySQL (7)

- Crear un SP con transacciones (¡***Cuidado***! No se pueden definir transacciones dentro de las Stored Functions, sólo en los Stored Procedures)

```
CREATE PROCEDURE transferir(pMonto DOUBLE,
                            pOrigen INT,
                            pDestino INT)
BEGIN
    # Ante una excepción hacer el rollback:
    DECLARE EXIT HANDLER FOR SQLEXCEPTION ROLLBACK;
    START TRANSACTION;
    UPDATE Cuentas SET saldo=saldo-pMonto
       WHERE idCuenta=pOrigen;
    UPDATE Cuentas SET saldo=saldo+pMonto
       WHERE idCuenta=pDestino;
    COMMIT;
END//
```