Universidad del Quindío – Ingeniería de Sistemas - Leonardo Hernández R.
leonardohernandez@telesat.com.co - www.geocities.com/leoher314
Sep. 9/2002, mar 6/2003

Tutorial de MySQL
Lección II

1. Scripts MySQL

Los scripts MySQL son archivos de texto comunes y corrientes, que contienen una o varias instrucciones SQL. Usándolos, se pueden ejecutar las instrucciones repetidamente sin tener que escribirlas de nuevo cada vez.

Los scripts se crean con cualquier editor de texto. Si utiliza el el bloc de notas se recomienda que desactive la opción de Windows “Ocultar extensiones para los tipos de archivos conocidos” (ver Lección I), ya que el bloc de notas agrega automáticamente .txt a los archivos y estos pueden quedar con nombres tan curiosos como: crear.sql.txt.

Supongamos que se ha creado un script con los siguientes renglones:

```
create table Asignatura (
      idAsignatura mediumint unsigned not null,
      nombre varchar(30) not null,
      primary key(idAsignatura));

insert into Asignatura values
      (100,"Bases de Datos"),
      (200,"Redes");
select * from Asignatura;
```

Supongamos también que el archivo que contiene el anterior script se llama c:\scripts\crear.sql. La extensión .sql no es obligatoria pero sí muy recomendable.

Para ejecutar el script, desde la tarea cliente de MySQL, use la siguiente instrucción, pero tenga en cuenta que debe usar el slash derecho (/) y no el contrario (\ ) como es usual en MS-DOS:

```
mysql> source c:/scripts/crear.sql;
```

También es posible ejecutar scripts desde la línea de comandos del MS-DOS:

```
C:\WINDOWS>       cd ..
C:\>              cd mysql
C:\mysql>         cd bin
C:\mysql\bin>     mysql < c:\scripts\crear.sql
```

2. Selección de registros

Para este apartado usaremos como ejemplo la tabla Empleado:

Empleado(idEmpleado, nombre, apellido, sexo, estadoCivil, sueldo)

El sexo puede ser M o F. Para el estado civil se adoptará la siguiente convención:

s	soltero
c	casado
u	unión libre
v	viudo
r	religioso

La tabla se supone ya creada y con varios registros.

Ya hemos visto que la instrucción:

```
mysql> select * from Empleado;
```

muestra todo el contenido de la tabla. El asterisco hace que se muestren todos los atributos de la tabla. El resultado de esta instrucción es una salida similar a la siguiente:

idEmpleado	nombre	apellido	sexo	estadoCivil	sueldo
123	Juan	Valdés	m	c	1000000
727	Luz	Pérez	f	s	3000000
223	Leonardo	Hernández	m	c	500000
527	Ana Karina	López	f	s	300000
229	Alejandro	Santa	m	s	400000
299	James	Kirk	m	u	400000
391	Ana Karina	Pérez	f	u	500000
295	Angélica	Arias	f	u	300000
400	Magda	Silva	f	c	1000000
...	...	...	...	...	...

No se preocupe por la sencillez de la presentación. Los lenguajes PHP y HTML nos permitirán mejorar este aspecto.

Es posible limitar la salida a los registros que uno necesite, es decir, eliminar renglones de la salida anterior. Por ejemplo, para obtener una salida con solo los registros de empleadas de nombre Ana Karina use:

```
mysql> select * from Empleado where nombre = "Ana Karina";
```

idEmpleado	nombre	apellido	sexo	estadoCivil	sueldo
527	Ana Karina	López	f	s	300000
391	Ana Karina	Pérez	f	u	500000

```
2 rows in set (0.01 sec)
```

La salida de una instrucción select es una tabla, en este caso una tabla con solo dos registros.

No interesa si se usan mayúsculas o minúsculas dentro de las cadenas de caracteres y no importa si se teclean algunos espacios al final. Se puede usar la comilla sencilla o la doble, con tal de que se abra y se cierre la cadena con el mismo tipo de comilla. Por ejemplo la siguiente instrucción es equivalente a la anterior:

```
mysql> select * from Empleado where nombre = 'ANA KARIna  ';
```

Los espacios al comienzo o en la parte intermedia si son significativos. Por ejemplo, las siguientes instrucciones no equivalen a la anterior:

```
mysql> select * from Empleado where nombre = "  Ana Karina";
mysql> select * from Empleado where nombre = "Ana    Karina";
```

En las condiciones se puede usar operadores relacionales (=, >, <, <>, <=, >=, !=) y lógicos (and, or, not, &&, ||, !). El operador relacional "diferente" se puede representar por <> o por !=. Los operadores lógicos &&, ||, y ! representan respectivamente and, or y not. Por compatibilidad con otros motores de bases de datos y por claridad, se recomienda usar and, or y not.

3. Más ejemplos de selección de registros

A continuación se dan varios ejemplos de selección de registros, para la tabla:

```
Empleado(   idEmpleado, nombre, apellido, sexo, estadoCivil,
            fechaNacimiento, dependencia, sueldo )
```

que es la misma tabla del apartado anterior, pero con los atributos fechaNacimiento y dependencia.

a) Lista de los empleados de sexo masculino

```
mysql> select * from Empleado where sexo = 'm';
```

b) Lista de los empleados que ganan más de $500.000.

```
mysql> select * from Empleado where sueldo > 500000;
```

c) Lista de las empleadas solteras

```
mysql> select * from Empleado where sexo='f' and estadoCivil='s';
```

d) Lista de los empleados que nacieron antes de diciembre de 1970

```
mysql> select * from Empleado where fechaNacimiento < '1970-12-01';
```

4. Atributos nulos y la selección de registros

En MySQL, al igual que en lenguaje C, un operador relacional da como resultado uno (1) para verdadero y cero (0) para falso. Lo anterior se puede comprobar con la siguiente instrucción:

```
mysql> select 5=5, 5=4;
+-----+-----+
| 5=5 | 5=4 |
+-----+-----+
|   1 |   0 |
+-----+-----+
```

Sorprendentemente la comparación con *null* no da 1 ni 0 como esperaríamos, sino que da como resultado *null.* Por ejemplo:

```
mysql> select 1>null, 1<null, 1!=null, 1=null;
```

```
+--------+--------+---------+--------+
| 1>null | 1<null | 1!=null | 1=null |
+--------+--------+---------+--------+
|   NULL |   NULL |    NULL |   NULL |
+--------+--------+---------+--------+
```

La condición “1>null” se interpreta como “1 es mayor a un valor desconocido” . Si se piensa bien, no se puede decir que esta frase sea verdadera o falsa. Sólo se puede decir que se desconoce si es verdadera o falsa, de ahí que MySQL , acertadamente, de como resultado null.

Aplicando esto a nuestro ejemplo, tendremos que para obtener una lista de los empleados con fecha de nacimiento registrada, NO sirve la instrucción:

```
mysql> select * from Empleado where fechaNacimiento <> null;
```

porque la condición nunca sería verdadera, siempre sería NULL. Sin embargo la lista puede ser obtenida, gracias al operador *is not*:

```
mysql> select * from Empleado where fechaNacimiento is not null;
```

Para una lista de los empleados a quienes NO se les ha registrado la fecha de nacimiento use:

```
mysql> select * from Empleado where fechaNacimiento is null;
```

5. Funciones para trabajar con fechas

MySQL dispone de muchas funciones para trabajar con fechas. A continuación se explicarán algunas de ellas.

La función year extrae el año de una fecha. Por ejemplo, los datos de los empleados que nacieron en el año 1980 se pueden obtener con:

```
mysql> select * from Empleado where year(fechaNacimiento) = 1980;
```

La función month extrae el mes de una fecha. Por ejemplo, los datos de los empleados que cumplen años en noviembre o en diciembre se pueden obtener con:

```
mysql> select * from Empleado where month(fechaNacimiento)>10;
```

La función dayofmonth retorna el día del mes de una fecha dada. Ejemplo: Los datos de los empleados que nacieron después del día 15 del mes se pueden obtener con:

```
mysql> select * from Empleado where dayofmonth(fechaNacimiento)>15;
```

La siguiente instrucción produce una lista de los empleados que cumplen años este mes:

```
mysql> select * from Empleado where month(fechaNacimiento) = month(now());
```

Para un lista de los empleados que cumplen años el siguiente mes, parece natural la instrucción:

```
mysql> select * from Empleado
    -> where month(fechaNacimiento) = month( now() ) + 1;
```

sin embargo, esto no funcionará si estamos en diciembre, ya que buscará los empleados nacidos en el mes trece. La instrucción correcta es la siguiente:

```
mysql> select * from Empleado
    -> where month(fechaNacimiento) = mod(month(now()),12)+1;
```

Veamos que por ejemplo que la expresión encuentra que el mes siguiente a octubre (10) es noviembre (11) y que el mes siguiente a diciembre (12) es enero (1):

mod(month('1980-10-11'), 12) + 1 = mod(**10**, 12) + 1 = **10** + 1 = 11.
mod(month('1980-12-11'), 12) + 1 = mod(**12**, 12) + 1 = **0** + 1 = 1.

Otra alternativa es utilizar el operador *interval* para hallar la fecha correspondiente a un mes a partir del día de hoy y luego con la función *month* extraerle el mes a dicha fecha.

```
mysql> select * from Empleado
    -> where month(fechaNacimiento) = month (now() + interval 1 month);
```

En el manual se puede consultar la descripción de otras interesantes funciones que trabajan con fechas y horas, como por ejemplo: DAYOFWEEK(date), WEEKDAY(date), DAYOFYEAR(date), DAYNAME(date), MONTHNAME(date), HOUR(time), MINUTE(time), SECOND(time).

6. Selección de Atributos (de columnas)

Es posible excluir del resultado de la consulta, columnas que en el momento no se necesiten. Por ejemplo, para ver solamente el nombre, el apellido y el sueldo de los empleados use:

```
mysql> select nombre, apellido, sueldo from Empleado;

+------------+-----------+---------+
| nombre     | apellido  | sueldo  |
+------------+-----------+---------+
| Juan       | Valdés    | 1000000 |
| Luz        | Pérez     | 3000000 |
| Leonardo   | Hernández |  500000 |
| Ana Karina | López     |  300000 |
| Alejandro  | Santa     |  400000 |
| James      | Kirk      |  400000 |
| Ana Karina | Pérez     |  500000 |
| Angélica   | Arias     |  300000 |
| Magda      | Silva     | 1000000 |
+------------+-----------+---------+
9 rows in set (0.00 sec)
```

Se puede mezclar la selección de columnas, con la selección de filas. Por ejemplo la siguiente instrucción mostrará el número de identificación, el nombre y el apellido de las mujeres solteras:

```
mysql> select idEmpleado, nombre, apellido from Empleado
    -> where sexo = 'f' and estadoCivil='s';
```

Incluso se puede reducir la salida a una sola columna:

```
mysql> select sueldo from Empleado;
```

La anterior consulta producirá con mucha probabilidad sueldos repetidos. Para evitar esto use:

```
mysql> select distinct sueldo from Empleado;
```

7. Definición de columnas mediante expresiones algebraicas

Se puede obtener salidas, donde una columna sea el resultado de un cálculo y no un atributo de la tabla. Por ejemplo podemos realizar una consulta donde para cada empleado se muestre el sueldo actual y el sueldo con un incremento del 10%:

```
mysql> select nombre, apellido, sueldo, sueldo*1.1 from Empleado;
+------------+-----------+---------+------------+
| nombre     | apellido  | sueldo  | sueldo*1.1 |
+------------+-----------+---------+------------+
| Juan       | Valdés    | 1000000 |  1100000.0 |
| ...        | ...       | ...     |  ...       |
+------------+-----------+---------+------------+
```

Se puede cambiar el encabezado “sueldo*1.1”, usando un “alias” del nombre del atributo:

```
mysql> select nombre, apellido, sueldo, sueldo*1.1 as sueldoNuevo
    -> from Empleado;
+------------+-----------+---------+-------------+
| nombre     | apellido  | sueldo  | sueldoNuevo |
+------------+-----------+---------+-------------+
| Juan       | Valdés    | 1000000 |   1100000.0 |
| ...        | ...       | ...     |   ...       |
+------------+-----------+---------+-------------+
```

Una lista de empleados con nombre, apellido, fecha de nacimiento y la fecha 15 días antes de la fecha de nacimiento se obtiene con:

```
mysql> select nombre, apellido,
    -> fechaNacimiento, fechaNacimiento - interval 15 day from Empleado;
```

8. Exportación de datos

Es posible migrar datos de tablas MySQL a otros programas como por ejemplo Excel. Esto se logra pasando los datos de las tablas MySQL a un archivo de texto y luego de aquí a la nueva aplicación.

Para pasar los datos de nuestra tabla Empleado a un archivo de texto se deben efectuar los siguientes pasos:

- Crear un script, por ejemplo c:\scripts\migrar.sql, con los siguientes renglones:

```
use nomina;
select * from Empleado;
```

- Ejecutar el script desde la línea de comandos del MS-DOS con:

```
C:\mysql\bin>  mysql < c:\scripts\migrar.sql > c:\datos\salida.txt
```

Se supuso que la carpeta *c:\datos* está previamente creada.

Los datos están ahora en el archivo de texto c:\datos\salida.txt y de aquí se podrán migrar a una gran variedad de aplicaciones.

9. Scripts

Se denomina script a un programa que no se compila, sino que es ejecutado línea por línea por un interpretador previamente instalado en el sistema. Si alguna línea tiene errores de sintaxis, el interpretador ejecutará hasta la línea inmediatamente anterior. En un lenguaje compilado, si hay un solo error de sintaxis, no se ejecuta ni una sóla línea del programa.

Los lenguajes de script suelen ser de más alto nivel (más orientados al ser humano que a la máquina) y más especializados que los programas compilados.

Los lenguajes compilados permiten escribir programas más eficientes en velocidad pero a la vez más difíciles de implementar. Los archivos ejecutables pueden ser de gran tamaño.

10. Ejercicios:

Elabore las consultas SQL que producen los siguientes resultados. Debe usar un solo select por cada numeral el cual debe producir exactamente los registros y las columnas indicadas:

1) El apellido y el sexo de los empleados que nacieron a partir del año 1970
2) Todos los datos de los hombres casados.
3) Todos los datos de las mujeres casadas o en unión libre.
4) Todos los datos de los hombres solteros y de las mujeres casadas
5) Todos los datos de los empleados que no nacieron en los años 80.
6) Todos los datos de los hombres solteros o viudos.
7) El nombre y el apellido de los hombres que nacieron en 1980 o en 1981.
8) Todos los datos de los empleados que NO nacieron en 1985 ni en 1986.
9) Los nombres y apellidos de los empleados con fecha de nacimiento desconocida y con estado civil conocido.
10) El nombre, apellido y el número de días vividos por cada empleado.
11) Los apellidos de los empleados que ganan entre $500.000 y $600.000 o entre $800.000 y $900.000.
12) Los nombres y los apellidos de los empleados que no nacieron en el segundo semestre de 1980.

11. Respuestas:

```
1)    select apellido, sexo from Empleado
      where year(fechaNacimiento) >= 1970;
3)    select * from Empleado
      where sexo = ‘f’ and ( estadoCivil = ‘c’ or estadoCivil = ‘u’);
5)    select * from Empleado
      where year(fechaNacimiento) < 1980 or year(fechaNacimiento)>1989;
7)    select nombre, apellido from Empleado
      where sexo = ‘m’ and ( year(fechaNacimiento) = 1980 or
      year(fechaNacimiento) = 1981);
9)    select nombre, apellido from Empleado
      where  fechaNacimiento is null and estadoCivil is not null;
11)   select apellido from Empleado
      where ( 500000 <= sueldo and sueldo <= 600000) or
      ( 800000 <= sueldo and sueldo <= 900000);
2)    select * from Empleado
      where sexo = ‘m’ and estadoCivil = ‘c’;
```

8

```
4)    select * from Empleado
      where ( sexo = “m” and estadoCivil = ‘s’ ) or ( sexo=’f’ and
      estadoCivil=’c’);
6)    select * from Empleado
      where sexo = ‘m’ and ( estadoCivil = ‘s’ or estadoCivil = ‘v’);
8)    select * from Empleado
      where  year(fechaNacimiento) <> 1985 and year(fechaNacimiento) <> 1986;
10)   select nombre, apellido,
      to_days( current_date() ) - to_days(fechaNacimiento)
      from Empleado;
12)   select nombre, apellido from Empleado
      where  fechaNacimiento<”1980-07-01” or year(fechaNacimiento)>1980;
```