



**INSTITUTO POLITÉCNICO
NACIONAL**



ESCUELA SUPERIOR DE CÓMPUTO

BASE DE DATOS

PRÁCTICA 10

PROFESOR: HERNÁNDEZ CONTRERAS EULER

2CM10

PEREZ RAYA ALEJANDRO ADOLFO

Índice

1. Marco Teórico.....	3
2. Instrucciones.....	4
3. Desarrollo	5
4. Conclusión	11
5. Bibliografía	11

1. Marco Teórico

Un procedimiento almacenado es un conjunto de comandos SQL que pueden almacenarse en el servidor. Una vez que se hace, los clientes no necesitan relanzar los comandos individuales pero pueden en su lugar referirse al procedimiento almacenado.

Algunas situaciones en que los procedimientos almacenados pueden ser particularmente útiles:

- Cuando múltiples aplicaciones cliente se escriben en distintos lenguajes o funcionan en distintas plataformas, pero necesitan realizar la misma operación en la base de datos.
- Cuando la seguridad es muy importante. Los bancos, por ejemplo, usan procedimientos almacenados para todas las operaciones comunes. Esto proporciona un entorno seguro y consistente, y los procedimientos pueden asegurar que cada operación se logre apropiadamente. En tal entorno, las aplicaciones y los usuarios no obtendrían ningún acceso directo a las tablas de la base de datos, sólo pueden ejecutar algunos procedimientos almacenados.

Los procedimientos almacenados pueden mejorar el rendimiento ya que se necesita enviar menos información entre el servidor y el cliente. El intercambio que hay es que aumenta la carga del servidor de la base de datos ya que la mayoría del trabajo se realiza en la parte del servidor y no en el cliente. Considere esto si muchas máquinas cliente (como servidores Web) se sirven a sólo uno o pocos servidores de bases de datos.

Los procedimientos almacenados le permiten tener bibliotecas o funciones en el servidor de base de datos. Esta característica es compartida por los lenguajes de programación modernos que permiten este diseño interno, por ejemplo, usando clases. Usando estas características del lenguaje de programación cliente es beneficioso para el programador incluso fuera del entorno de la base de datos.

La sintaxis para crear un procedimiento que reciba parámetros es la siguiente:

```
create procedure nombreSP(in variable1 tipodedato(tamaño), in variable1 tipodedato(tamaño), ...  
, in variablen tipodedato(tamaño))  
begin  
    Sentencias sql.  
end
```

2. Instrucciones

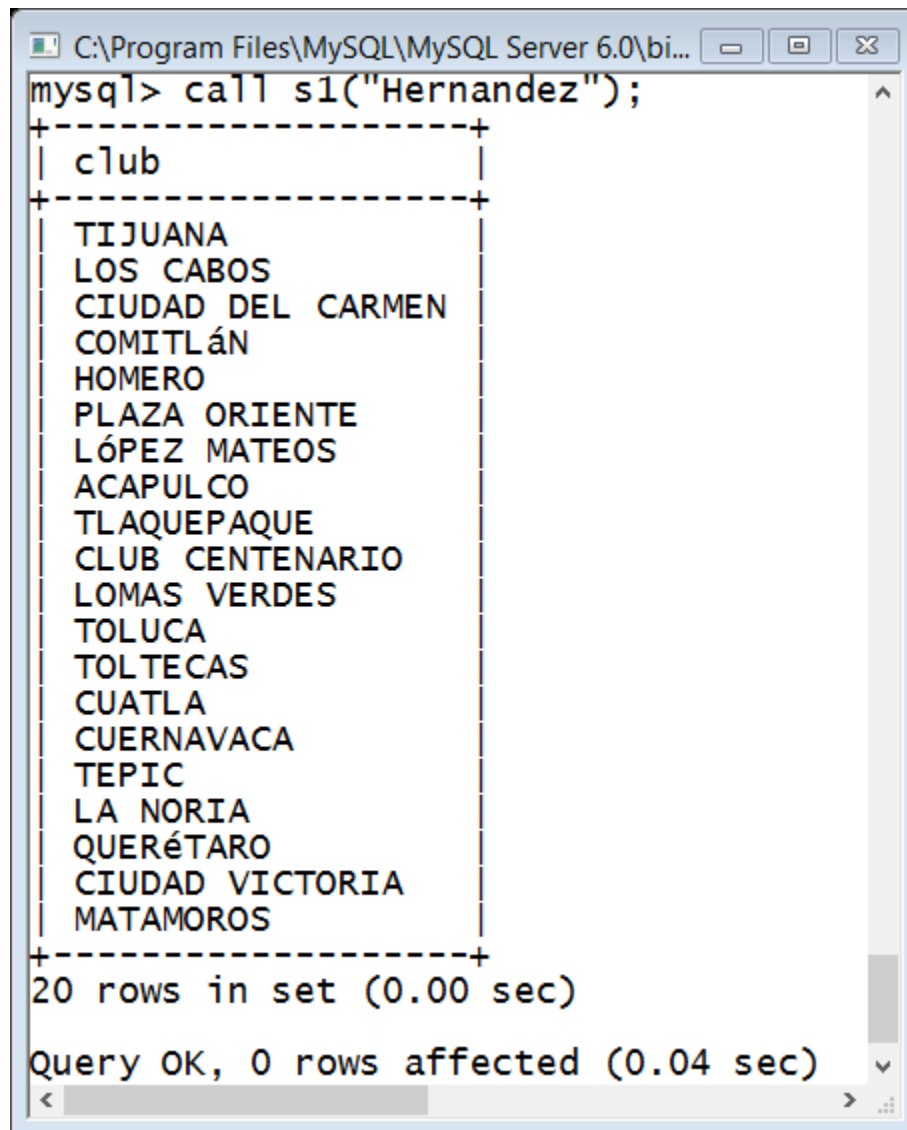
Crear los siguientes procedimientos almacenados

1. Cuales el nombre del club en donde se encuentra dado de alta un socio determinado.
2. Muestre el nombre del producto más su precio unitario de un proveedor determinado.
3. Indique el nombre del club y su ubicación (estado) de aquellos clubs que tienen un servicio determinado.
4. Muestre el id de Club y nombre de donde está un gerente determinado.
5. Que muestre los datos correspondientes de un socio a través de su id.
6. Que permita ingresar a un socio indicando el club.
7. Permita eliminar los productos de un proveedor determinado.

3. Desarrollo

1.

```
delimiter &&  
create procedure s1(in ns varchar(50))  
begin  
    select c.nombre as club from club c, socios s, sociosams x where c.idclub=x.idclub  
    and x.idsocio=s.idsocio and s.nombre like concat("%",ns,"%");  
end &&
```



The screenshot shows a MySQL command window titled "C:\Program Files\MySQL\MySQL Server 6.0\bi...". The command entered is `mysql> call s1("Hernandez");`. The output is a table with one column named "club" containing 20 rows of club names. Below the table, it says "20 rows in set (0.00 sec)". At the bottom, it says "Query OK, 0 rows affected (0.04 sec)".

club
TIJUANA
LOS CABOS
CIUDAD DEL CARMEN
COMITLÁN
HOMERO
PLAZA ORIENTE
LÓPEZ MATEOS
ACAPULCO
TLAQUEPAQUE
CLUB CENTENARIO
LOMAS VERDES
TOLUCA
TOLTECAS
CUATLA
CUERNAVACA
TEPIC
LA NORIA
QUERÉTARO
CIUDAD VICTORIA
MATAMOROS

20 rows in set (0.00 sec)

Query OK, 0 rows affected (0.04 sec)

2.

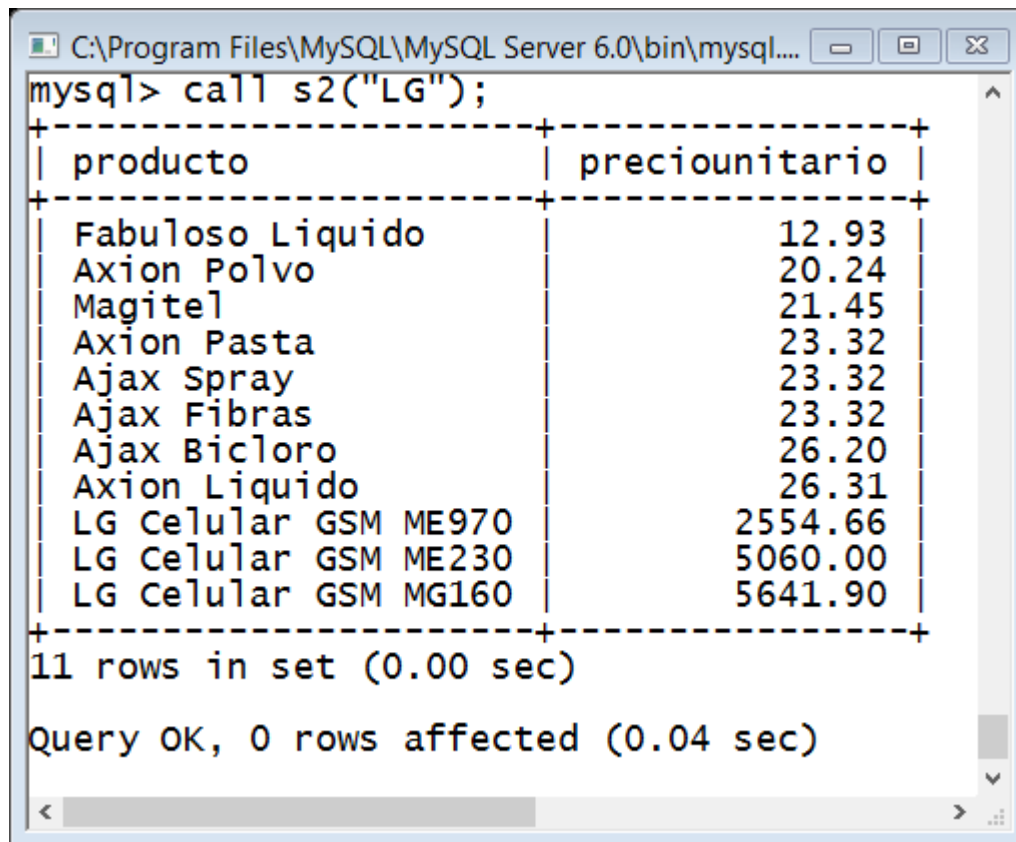
delimiter #&

create procedure s2(in x varchar(45))

begin

select p.producto, p.preciounitario from producto p, proveedor l where
p.idProveedor=l.idProveedor and l.nombre like concat("%",x,"%") order by
p.preciounitario;

end #&



The screenshot shows a MySQL command window with the title bar "C:\Program Files\MySQL\MySQL Server 6.0\bin\mysql...". The command prompt shows the execution of the stored procedure `call s2("LG");`. The results are displayed in a table with two columns: `producto` and `preciounitario`. The table contains 11 rows of data, including products like `Fabuloso Liquido`, `Axion Polvo`, `Magitel`, `Axion Pasta`, `Ajax Spray`, `Ajax Fibras`, `Ajax Bicloro`, `Axion Liquido`, and three LG mobile phones. Below the table, the status "11 rows in set (0.00 sec)" is shown, followed by "Query OK, 0 rows affected (0.04 sec)".

```
mysql> call s2("LG");
```

producto	preciounitario
Fabuloso Liquido	12.93
Axion Polvo	20.24
Magitel	21.45
Axion Pasta	23.32
Ajax Spray	23.32
Ajax Fibras	23.32
Ajax Bicloro	26.20
Axion Liquido	26.31
LG Celular GSM ME970	2554.66
LG Celular GSM ME230	5060.00
LG Celular GSM MG160	5641.90

11 rows in set (0.00 sec)

Query OK, 0 rows affected (0.04 sec)

3.

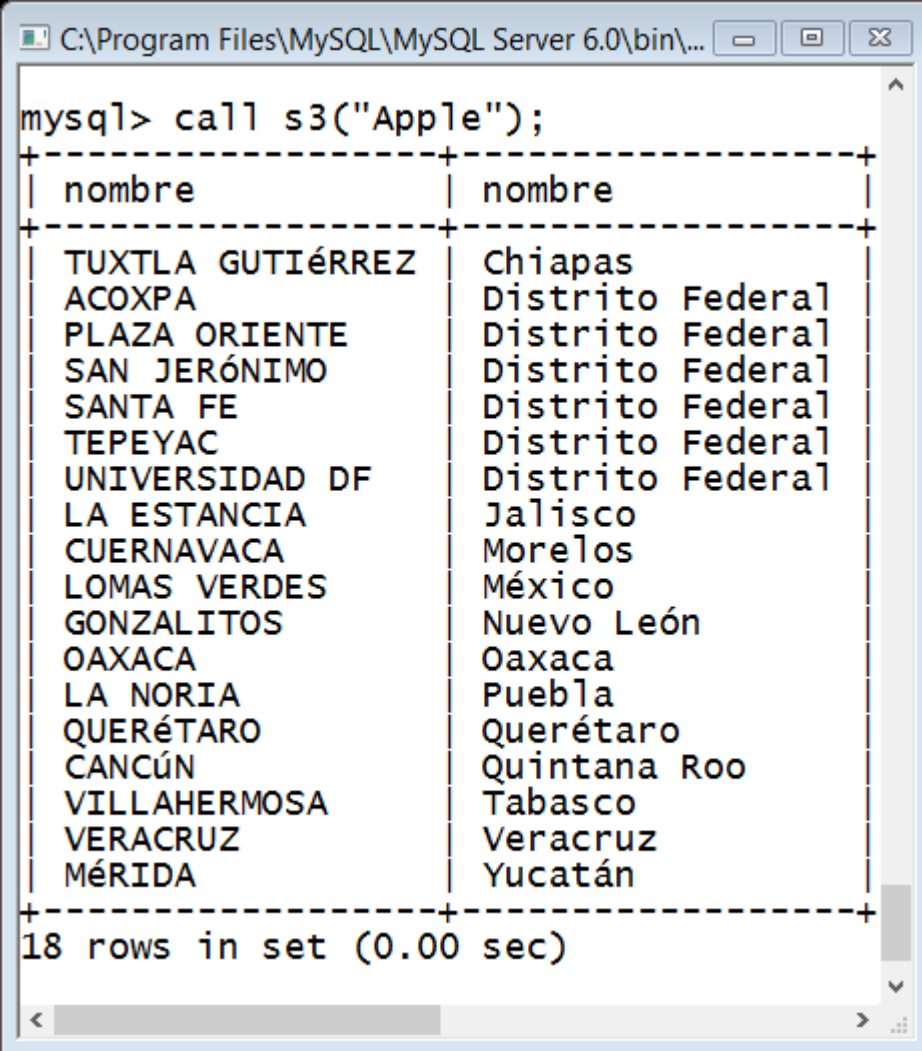
```
delimiter ##
```

```
create procedure s3(in ser varchar(30))
```

```
begin
```

```
    select c.nombre, e.nombre from club c,estado e, servicio s,servicioclub x where  
        c.idedo=e.idedo and c.idclub=x.idclub and x.idservicio=s.idservicio and s.nombre like  
        concat("%",ser,"%") order by e.nombre;
```

```
end ##
```



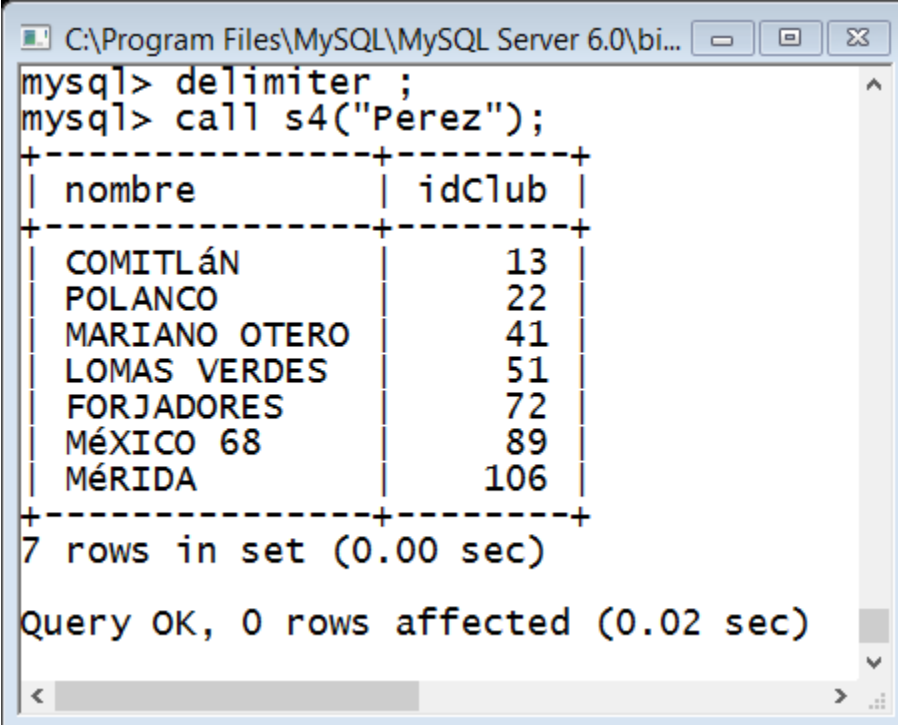
The screenshot shows a MySQL command window titled "C:\Program Files\MySQL\MySQL Server 6.0\bin\...". The command entered is `mysql> call s3("Apple");`. The output is a table with two columns, both labeled "nombre". The first column lists 18 club names, and the second column lists the corresponding state names. The results are as follows:

nombre	nombre
TUXTLA GUTIÉRREZ	Chiapas
ACOXPA	Distrito Federal
PLAZA ORIENTE	Distrito Federal
SAN JERÓNIMO	Distrito Federal
SANTA FE	Distrito Federal
TEPEYAC	Distrito Federal
UNIVERSIDAD DF	Distrito Federal
LA ESTANCIA	Jalisco
CUERNAVACA	Morelos
LOMAS VERDES	México
GONZALITOS	Nuevo León
OAXACA	Oaxaca
LA NORIA	Puebla
QUERÉTARO	Querétaro
CANCÚN	Quintana Roo
VILLAHERMOSA	Tabasco
VERACRUZ	Veracruz
MÉRIDA	Yucatán

Below the table, the text "18 rows in set (0.00 sec)" is displayed. The window has a scrollbar on the right and a status bar at the bottom.

4.

```
delimiter //  
create procedure s4(in ger varchar(50))  
begin  
    select c.nombre,c.idClub from club c,gerente g where c.idclub=g.idclub and  
        g.nombre like concat("%",ger,"%");  
end //
```



The screenshot shows a MySQL command window titled "C:\Program Files\MySQL\MySQL Server 6.0\bi...". The command prompt shows the following commands and output:

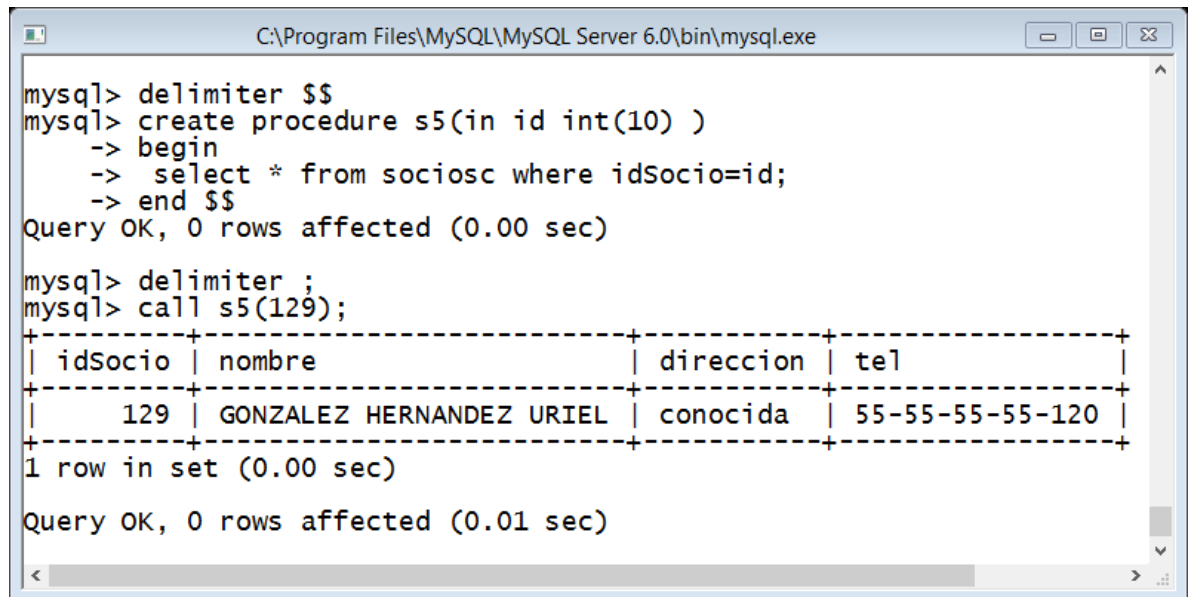
```
mysql> delimiter ;  
mysql> call s4("Perez");
```

nombre	idClub
COMITLÁN	13
POLANCO	22
MARIANO OTERO	41
LOMAS VERDES	51
FORJADORES	72
MÉXICO 68	89
MÉRIDA	106

```
7 rows in set (0.00 sec)  
  
Query OK, 0 rows affected (0.02 sec)
```


5.

```
delimiter $$  
create procedure s5(in id int(10) )  
begin  
    select * from sociosc where idSocio=id;  
end $$
```



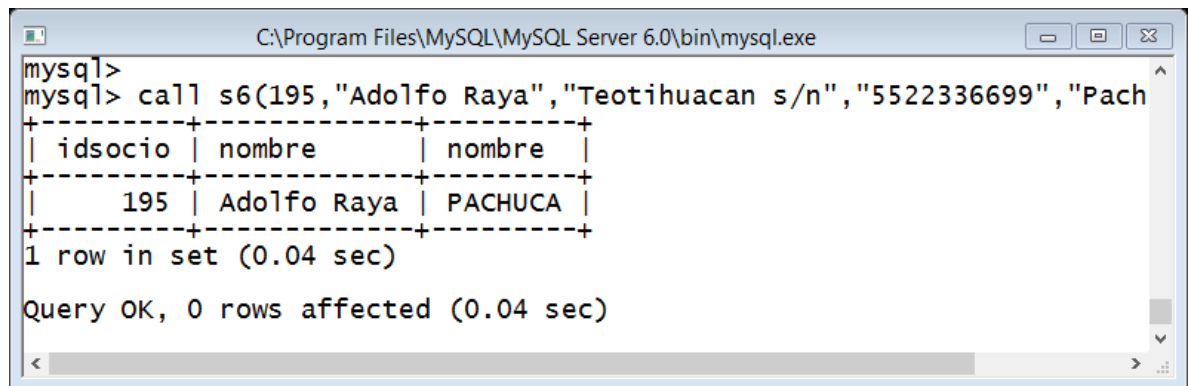
The screenshot shows a MySQL command prompt window with the following text:

```
C:\Program Files\MySQL\MySQL Server 6.0\bin\mysql.exe  
mysql> delimiter $$  
mysql> create procedure s5(in id int(10) )  
-> begin  
-> select * from sociosc where idSocio=id;  
-> end $$  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> delimiter ;  
mysql> call s5(129);  
+-----+-----+-----+-----+  
| idSocio | nombre | direccion | tel |  
+-----+-----+-----+-----+  
| 129 | GONZALEZ HERNANDEZ URIEL | conocida | 55-55-55-55-120 |  
+-----+-----+-----+-----+  
1 row in set (0.00 sec)  
  
Query OK, 0 rows affected (0.01 sec)
```

idSocio	nombre	direccion	tel
129	GONZALEZ HERNANDEZ URIEL	conocida	55-55-55-55-120

6.

```
delimiter ++
create procedure s6(in id int,in n varchar(50),in d varchar(100),in t varchar(15),in c
varchar(45))
begin
-- agregar socio
    insert into sociosc values(id,n,d,t);
-- agregar socioclub
    insert into sociosams values(id,(select idClub from club where nombre like concat(c)));
-- consulta
    select s.idsocio, s.nombre, c.nombre from sociosc s, club c, sociosams x where
s.idsocio=x.idsocio and x.idclub=c.idclub and s.idsocio=id;
end ++
```

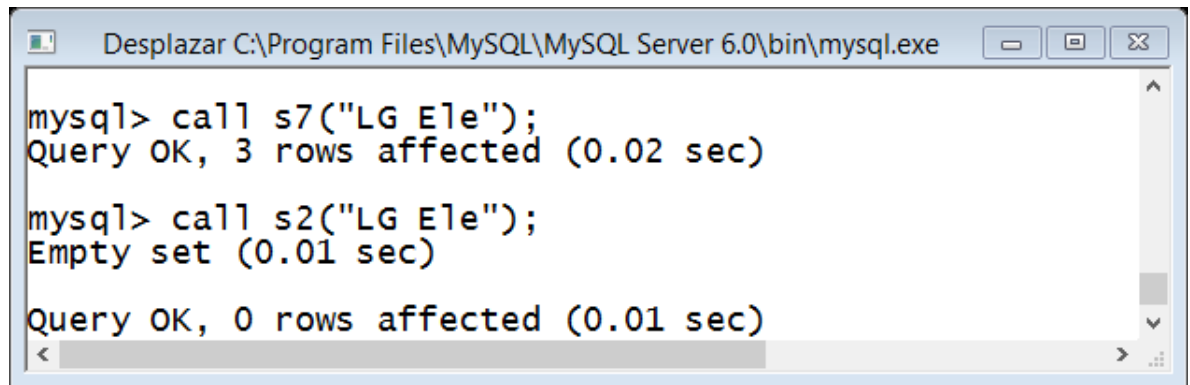


The screenshot shows a MySQL command window titled "C:\Program Files\MySQL\MySQL Server 6.0\bin\mysql.exe". The user has entered the command `mysql> call s6(195,"Adolfo Raya","Teotihuacan s/n","5522336699","Pach`. The output displays a table with three columns: `idsocio`, `nombre`, and `nombre`. The first row contains the values 195, Adolfo Raya, and PACHUCA. Below the table, it states "1 row in set (0.04 sec)" and "Query OK, 0 rows affected (0.04 sec)".

```
mysql>
mysql> call s6(195,"Adolfo Raya","Teotihuacan s/n","5522336699","Pach
+-----+-----+-----+
| idsocio | nombre      | nombre |
+-----+-----+-----+
|      195 | Adolfo Raya | PACHUCA |
+-----+-----+-----+
1 row in set (0.04 sec)
Query OK, 0 rows affected (0.04 sec)
```

7.

```
delimiter $#  
create procedure s7(in pro varchar(45))  
begin  
    delete from producto where idProveedor=(select idProveedor from proveedor where  
        nombre like concat(pro));  
end $#
```



The screenshot shows a Windows command prompt window titled "Desplazar C:\Program Files\MySQL\MySQL Server 6.0\bin\mysql.exe". The prompt is "mysql>". The first command entered is "call s7('LG Ele');", which returns "Query OK, 3 rows affected (0.02 sec)". The second command entered is "call s2('LG Ele');", which returns "Empty set (0.01 sec)". A third line shows "Query OK, 0 rows affected (0.01 sec)". The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
mysql> call s7("LG Ele");  
Query OK, 3 rows affected (0.02 sec)  
  
mysql> call s2("LG Ele");  
Empty set (0.01 sec)  
  
Query OK, 0 rows affected (0.01 sec)
```

4. Conclusión

En esta práctica le mandamos parámetros a los procedimientos lo cual hace aún más fácil el realizar consultas, actualizaciones e inserciones. Esto no es de gran ayuda por que basta con escribir la sentencia una sola vez y luego solo la mandamos a llamar cuantas veces necesitemos. Cuando programamos alguna aplicación que se conectará con una base de datos el uso de los procedimientos son una gran herramienta ya que agilizará las ejecuciones de instrucciones sql, y nuestra aplicación será más eficiente.

5. Bibliografía

1. *Capítulo 19. Procedimientos almacenados y funciones*. Recuperado 25 de noviembre de 2013.
[En línea]. Disponible en: <http://dev.mysql.com/doc/refman/5.0/es/stored-procedures.html>