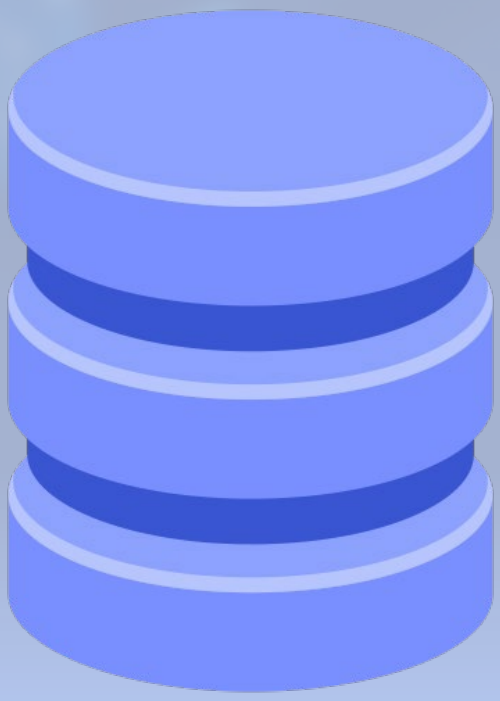
PROCEDIMIENTOS II: MYSQL



GESTIÓN DE BASES DE DATOS

JUAN CARLOS NAVIDAD GARCÍA

1. **Crear un procedimiento para obtener las líneas de un pedido que se le pasa como parámetro, junto con la descripción y el precio de los productos.**

```
create PROCEDURE lin_pedido(IN xidlinea int)
BEGIN
select descripcion, precio from lineaspedido,productos
where lineaspedido.idproducto=productos.idprod and
idlinea=xidlinea ;
end //
```

```
mysql> call lin_pedido(1) //
+------------------------+--------+
| descripcion            | precio |
+------------------------+--------+
| Budweiser              |   1.71 |
| Pizza Carbonara        |  12.30 |
| Hamburguesa Basica     |   2.37 |
| Hamburguesa con Queso  |   3.10 |
| Pizza Peperoni         |  11.35 |
| Fanta Naranja          |   1.81 |
| Coca Cola              |   1.66 |
| Hamburguesa Basica     |   2.37 |
| Paulaner               |   2.61 |
| Coca Cola Light        |   1.71 |
| Spaguetti Bolognesa    |   4.50 |
+------------------------+--------+
11 rows in set (0.02 sec)
```

2. **Crear una función para poder actualizar el saldo de una cuenta en función de los movimientos que tenga. Aplicar esa función a todas las cuentas del banco a través de un procedimiento.**

```
create function act_saldo1(xcodcuenta varchar(10)) returns decimal DETERMINISTIC
begin
declare total decimal;
select sum(cantidad) into total from transacciones WHERE codcuenta=xcodcuenta;
return total;
end //
```

```
create PROCEDURE act_saldo()
begin
update cuentas set saldo=act_saldo1(codcuenta);
end //
```

```
mysql> call act_saldo()//
Query OK, 5 rows affected (0.00 sec)
```

3. **Crear un procedimiento para obtener un listado de las habitaciones ocupadas junto con el nombre del cliente en una fecha determinada que se le pasa como parámetro.**

```
create PROCEDURE hab_libre(IN xfecha date)
begin
select Nombre, habitaciones.* from habitaciones,ocupaciones,clientes where
clientes.numcli=ocupaciones.numcli and habitaciones.numhab=ocupaciones.numhab and
xfecha between fentrada and fsalida group by numhab//
end //
```

2022-01-07

```
mysql> call hab_libre("2022-01-06")//
+-----------------+--------+-----------+-----------+---------+
| Nombre          | NumHab | preciodia | categoria | fumador |
+-----------------+--------+-----------+-----------+---------+
| Andres Martinez | 201    |    100.80 | Doble     | S       |
+-----------------+--------+-----------+-----------+---------+
1 row in set (0.00 sec)
```

4. **Crear un procedimiento que devuelva el saldo de una cuenta. En caso de que la cuenta no exista, tiene que devolver "Cuenta XXXXXXXXX inexistente".**

```
create PROCEDURE saldo(IN xcuenta varchar(10))
begin
declare cont int;
select count(*) into cont from cuentas where codcuenta=xcuenta;
if cont>0 THEN
select saldo from cuentas where codcuenta=xcuenta;
else
select concat ("Cuenta: ",xcuenta," inexistente");
end if;
end //
```

```
mysql> call saldo("234567891") //
+--------+
| saldo  |
+--------+
| 179.00 |
+--------+
1 row in set (0.00 sec)
```

```
mysql> call saldo("234567890") //
+-----------------------------------------+
| concat ("Cuenta: ",xcuenta," inexistente") |
+-----------------------------------------+
| Cuenta: 234567890 inexistente           |
+-----------------------------------------+
1 row in set (0.00 sec)
```

5. **Crear un procedimiento que devuelva el total vendido por la pizzería y el nombre del producto más vendido hasta la fecha. Estos datos hay que devolverlos en variables de salida.**

```
CREATE Procedure total_pizza (out caja DECIMAL(6,2),out pizza varchar(30))
BEGIN
select sum(total*unidades) into caja from pedidos,lineaspedido where
pedidos.idped=lineaspedido.idped;
select descripcion into pizza from productos,lineaspedido where idproducto=idprod
group by idproducto order by (count(idproducto)*unidades) desc LIMIT 1;
end //
```

```
mysql> select @caja;
+---------+
| @caja   |
+---------+
| 4041.48 |
+---------+
1 row in set (0.00 sec)
```

```
mysql> select @pizza;
+----------+
| @pizza   |
+----------+
| Heineken |
+----------+
1 row in set (0.00 sec)
```

6. **Crear un procedimiento que me devuelva en una variable de salida la suma de todos los saldos de las cuentas de una determinada sucursal.**

```
CREATE Procedure total_sucursal (IN xsucursal varchar(4),OUT xtotal DECIMAL(6,2))
BEGIN
SELECT SUM(saldo) INTO xtotal FROM cuentas where sucursal=xsucursal;
END //
```

```
mysql> call total_sucursal("1235",@xtotal);
Query OK, 1 row affected (0.00 sec)

mysql> select @xtotal;
+---------+
| @xtotal |
+---------+
| 9675.00 |
+---------+
1 row in set (0.00 sec)
```

7. **Crear un procedimiento para listar los artículos que no se han vendido en la última semana.**

```
CREATE PROCEDURE no_vendido()
BEGIN
SELECT descripcion
FROM artículos, ventas
WHERE artículos.codart=ventas.codart and artículos.codart NOT IN (
    SELECT codart
    FROM ventas
    WHERE fechaventa between DATE_SUB(curdate(), INTERVAL 1 WEEK) and Curdate());
END //
```

```
mysql> call no_vendido();
+--------------------------+
| descripcion              |
+--------------------------+
| Pala Padel Head Evolution |
| Camiseta Térmica Negra L  |
| Chaquetón Snow Negro XL   |
| Mizuno Wave Unitus 4      |
| Mizuno Wave Unitus 4      |
| Mizuno Wave Unitus 4      |
| Zapatillas Running 40-43  |
| Zapatillas Basket 44-46   |
| Zapatillas Basket 44-46   |
| Malla Gimnasio Mujer      |
| Calcetines Unisex 38-42 P |
| Calcetines Unisex 38-42 P |
+--------------------------+
12 rows in set (0.00 sec)
```

8. **Crear un procedimiento al que se le pasa un tipo de producto y un porcentaje y actualice los precios de dichos productos, incrementándolos en el porcentaje indicado.**

```
CREATE PROCEDURE act_precio (IN xtipo VARCHAR(1), IN xporcentaje int)
BEGIN
UPDATE productos SET precio=precio+((precio*xporcentaje)/100) WHERE tipo=xtipo;
END //
```

```
mysql> call act_precio("E", 20)//
Query OK, 3 rows affected, 3 warnings (0.00 sec)
```