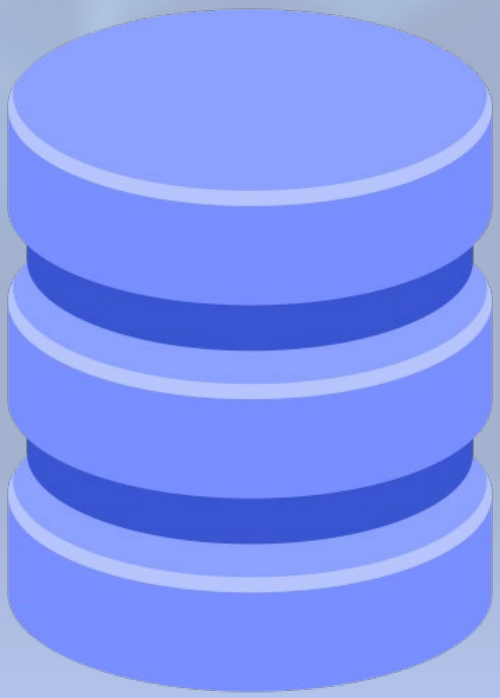
MYSQL: REPASO III



GESTIÓN DE BASES DE DATOS

JUAN CARLOS NAVIDAD GARCÍA

1. **Obtener un listado de los productos que se han vendido con precio superior a la media. Hacer con Inner Join.**

```
SELECT descripcion
FROM productos
INNER JOIN lineaspedido ON productos.idprod=lineaspedido.idproducto
INNER JOIN pedidos ON lineaspedido.idped=pedidos.idped
WHERE productos.precio<(SELECT AVG(total) FROM pedidos);
```

2. **Crear una rutina para añadir un producto. Hay que tener en cuenta claves primarias y ajenas y mostrar mensajes adecuados.**

```
CREATE PROCEDURE ej2(IN xprod VARCHAR(4), IN xdesc VARCHAR(30), IN xprecio DECIMAL,
IN xtipo ENUM('H','B','P','E','F','I','X'))
BEGIN
DECLARE idcont INT;
SELECT COUNT(*) INTO idcont FROM productos WHERE idprod=xprod;
IF idcont>0 THEN
SELECT "El id del producto ya ha sido introducido." AS "Aviso: ";
ELSE
IF xtipo='H' OR xtipo='B'
OR xtipo='P' OR xtipo='E'
OR xtipo='F' OR xtipo='I'
OR xtipo='X' THEN
INSERT INTO productos VALUES (xprod, xdesc, xprecio, xtipo);
SELECT "El producto ha sido introducido." AS "Mensaje: ";
ELSE
SELECT "El tipo no es correcto." AS "Aviso: ";
END IF;
END IF;
END //
```

```
mysql> CALL ej2("B001","Coca Cola",2.50,"B")//
+-----------------------------------------+
| Aviso:                                  |
+-----------------------------------------+
| El id del producto ya ha sido introducido. |
+-----------------------------------------+
1 row in set (0.00 sec)
```

```
mysql> CALL ej2("B050","Coca Cola Frambuesa",3.00,"V")//
+------------------------+
| Aviso:                 |
+------------------------+
| El tipo no es correcto. |
+------------------------+
1 row in set (0.00 sec)
```

```
mysql> CALL ej2("B050","Coca Cola Frambuesa",3.00,'B')//
+---------------------------------+
| Mensaje:                        |
+---------------------------------+
| El producto ha sido introducido. |
+---------------------------------+
1 row in set (0.01 sec)
```

3. **Crear una función para que calcule el total de un pedido. Se lee de LineasPedido Utilizar dicha función para actualizar los totales de la tabla pedidos.**

```
CREATE FUNCTION ej3(xidped INT) RETURNS DECIMAL(8,2) DETERMINISTIC
BEGIN
DECLARE resultado DECIMAL(8,2);
SELECT SUM(precio*unidades) INTO resultado
FROM lineaspedido
INNER JOIN productos ON lineaspedido.idproducto=productos.idprod
WHERE idped=xidped;
RETURN resultado;
END //
```

```
mysql> SELECT ej3(idped) FROM pedidos//
+------------+
| ej3(idped) |
+------------+
|      14.00 |
|      57.25 |
|      61.50 |
|     107.75 |
|      35.25 |
|      44.00 |
|      87.00 |
|      50.00 |
|      30.00 |
|     162.00 |
|      49.00 |
|      30.50 |
+------------+
12 rows in set (0.01 sec)
```

```
CREATE PROCEDURE ej_3()
BEGIN
UPDATE pedidos SET total=(SELECT ej3(idped));
END //
```

```
mysql> SELECT idped,total FROM pedidos//
+-------+--------+
| idped | total  |
+-------+--------+
|     1 |  14.00 |
|     2 |  57.25 |
|     3 |  61.50 |
|    14 |  30.50 |
|     5 | 107.75 |
|     6 |  35.25 |
|     7 |  44.00 |
|     8 |  87.00 |
|     9 |  50.00 |
|    10 |  30.00 |
|    11 | 162.00 |
|    13 |  49.00 |
+-------+--------+
12 rows in set (0.00 sec)
```

4. **Crear una función que calcule la caja hecha un día concreto. Si ese dia no hay ventas, devolverá un cero. Utilizar la función en una consulta. Realizar el mismo proceso con un procedimiento almacenando el resultado en una variable de salida.**

```
CREATE FUNCTION ej4(xdia DATE) RETURNS DECIMAL(8,2) DETERMINISTIC
BEGIN
DECLARE resultado DECIMAL(8,2);
SELECT SUM(precio*unidades) INTO resultado
FROM lineaspedido
INNER JOIN productos ON lineaspedido.idproducto=productos.idprod
INNER JOIN pedidos ON lineaspedido.idped=pedidos.idped
WHERE fecha=xdia;
RETURN resultado;
END //
```

```
mysql> SELECT ej4(CURDATE())//
+----------------+
| ej4(CURDATE()) |
+----------------+
|          79.50 |
+----------------+
1 row in set (0.01 sec)
```

```
CREATE PROCEDURE ej4(IN xdia DATE, OUT caja DECIMAL(8,2))
BEGIN
SELECT SUM(precio*unidades) INTO caja
FROM lineaspedido
INNER JOIN productos ON lineaspedido.idproducto=productos.idprod
INNER JOIN pedidos ON lineaspedido.idped=pedidos.idped
WHERE fecha=xdia;
END //
```

```
mysql> CALL ej4(CURDATE(),@caja)//
Query OK, 1 row affected (0.00 sec)

mysql> SELECT @caja//
+-------+
| @caja |
+-------+
| 79.50 |
+-------+
1 row in set (0.00 sec)
```

5. **Obtener los datos del pedido en el que se han vendido más productos.**

```
mysql> SELECT *,COUNT(idproducto) AS productos FROM pedidos
    -> INNER JOIN lineaspedido ON pedidos.idped=lineaspedido.idped
    -> GROUP BY lineaspedido.idped
    -> ORDER BY COUNT(idproducto) DESC LIMIT 1//
+-------+-------+-------+------------+--------+---------+-------+------------+----------+-----------+
| idped | idemp | idcli | fecha      | total  | idlinea | idped | idproducto | unidades | productos |
+-------+-------+-------+------------+--------+---------+-------+------------+----------+-----------+
|     5 | R02   | 007   | 2023-02-01 | 107.75 |       1 |     5 | P004       |        2 |         7 |
+-------+-------+-------+------------+--------+---------+-------+------------+----------+-----------+
1 row in set (0.00 sec)
```

6. **Crear un evento para que cada día, a las 1:00 am, en una tabla llamada Cajadia, genere las ventas totales del día anterior. La tabla contendrá los campos fecha y total.**

```
CREATE EVENT IF NOT EXISTS ej6
ON SCHEDULE EVERY 1 DAY
STARTS '2023-04-27 01:00:00'
DO
INSERT INTO cajadia VALUES(DATE_SUB(CURDATE(), INTERVAL 1 DAY),(SELECT
ej4(DATE_SUB(CURDATE(), INTERVAL 1 DAY))));
END //
```

3	11	R02	010	2023-04-25	162.00

```
mysql> SELECT * FROM cajadia//
+------------+--------+
| Fecha      | Total  |
+------------+--------+
| 2023-04-25 | 162.00 |
+------------+--------+
1 row in set (0.00 sec)
```

7. **Crear una rutina que permita obtener el listado de productos que componen un pedido junto con el número de unidades vendidas, su precio y su total parcial. El total del pedido hay que devolverlo como variable.**

   Num.Pedido Descripción Producto Unidades Precio Total

8. **Crear una función que obtenga la suma total de los pedidos gestionados por un empleado en concreto. Hay que comprobar que ese empleado existe previamente.**

```
CREATE FUNCTION ej8(xidemp VARCHAR(3)) RETURNS VARCHAR(50) DETERMINISTIC
BEGIN
DECLARE resultado VARCHAR(50);
DECLARE total INT;
DECLARE cont INT;
SELECT COUNT(*) INTO cont FROM empleados WHERE idemp=xidemp;
IF cont>0 THEN
SELECT COUNT(*) INTO total FROM pedidos WHERE idemp=xidemp;
SELECT CONCAT("El empleado: ",xidemp," ha gestionado: ",total," pedidos.") INTO
resultado;
ELSE
SELECT CONCAT("El empleado: ",xidemp," no existe.") INTO resultado;
END IF;
RETURN resultado;
END //
```

```
mysql> SELECT ej8("C10")//
+---------------------------+
| ej8("C10")                |
+---------------------------+
| El empleado: C10 no existe. |
+---------------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT ej8("C01")//
+-----------------------------------------+
| ej8("C01")                              |
+-----------------------------------------+
| El empleado: C01 ha gestionado: 1 pedidos. |
+-----------------------------------------+
1 row in set (0.00 sec)
```

9. **Crear una vista para obtener el producto más comprado por las mujeres.**

```
CREATE VIEW ej9 AS
SELECT descripcion FROM productos
INNER JOIN lineaspedido ON lineaspedido.idproducto=productos.idprod
INNER JOIN pedidos ON lineaspedido.idped=pedidos.idped
INNER JOIN clientes ON pedidos.idcli=clientes.idcli
WHERE sexo='F' GROUP BY descripcion ORDER BY COUNT(idproducto) DESC LIMIT 1;//
```

```
mysql> SELECT * FROM ej9;//
+-----------------+
| descripcion     |
+-----------------+
| Pizza Carbonara |
+-----------------+
1 row in set (0.00 sec)
```

10. **Crear un evento para crear una copia de seguridad de toda la base de datos. Este se ejecutará dos veces al día. Tendremos una copia de mañana y otra de tarde.**

```
CREATE PROCEDURE ej10()
BEGIN
DROP TABLE IF EXISTS
categorias_bk,clientes_bk,empleados_bk,lineaspedido_bk,pedidos_bk,productos_bk;
CREATE TABLE categorias_bk LIKE categorias;
CREATE TABLE clientes_bk LIKE clientes;
CREATE TABLE empleados_bk LIKE empleados;
CREATE TABLE lineaspedido_bk LIKE lineaspedido;
CREATE TABLE pedidos_bk LIKE pedidos;
CREATE TABLE productos_bk LIKE productos;
END //
```

```
CREATE EVENT IF NOT EXISTS ej10
ON SCHEDULE EVERY 12 HOUR
STARTS '2023-04-27 08:00:00'
DO
CALL ej10();
END //
```