

# UNIDAD 13 ACTIVIDAD PRÁCTICA MANEJO DE DATOS

## EJERCICIO A.

Crea una tabla con los mismos campos que la tabla PRODUCTO pero que se llame HIST\_PRODUCTO. Los campos que son clave foránea estarán en la nueva tabla, pero no serán claves foráneas.

- Incluye a continuación el código SQL necesario para crearla, y una captura de pantalla donde se vea que se ha ejecutado correctamente.

## OPCION 1: CON PL/SQL

```
1 DECLARE
2   -- Variables para almacenar los nombres de tabla y columna
3   tabla_origen VARCHAR(20) := 'PRODUCTO';
4   tabla_destino VARCHAR(20) := 'HIST_PRODUCTO';
5   campo_origen VARCHAR(30);
6   tipo_origen VARCHAR(30);
7   primer BOOLEAN := TRUE;
8   CURSOR recorre IS SELECT column_name, data_type FROM user_tab_columns WHERE table_name = tabla_origen;
9 BEGIN
10  -- Loop para obtener los nombres de las columnas de la tabla original
11  FOR c IN recorre LOOP
12    -- Obtener el nombre y tipo de dato de columna actual
13    campo_origen := c.column_name;
14    tipo_origen := c.data_type;
15    -- Verificar si es la primera iteración
16    IF primer THEN
17      EXECUTE IMMEDIATE 'CREATE TABLE ' || tabla_destino || ' (' || campo_origen || ' ' || tipo_origen || '(30) PRIMARY KEY)';
18      primer := FALSE;
19    ELSE
20      IF tipo_origen = 'VARCHAR2' THEN
21        EXECUTE IMMEDIATE 'ALTER TABLE ' || tabla_destino || ' ADD (' || campo_origen || ' ' || tipo_origen || '(30))';
22      ELSE IF tipo_origen = 'NUMBER' THEN
23        EXECUTE IMMEDIATE 'ALTER TABLE ' || tabla_destino || ' ADD (' || campo_origen || ' ' || tipo_origen || '(4,2))';
24      ELSE
25        EXECUTE IMMEDIATE 'ALTER TABLE ' || tabla_destino || ' ADD (' || campo_origen || ' ' || tipo_origen || ')';
26      END IF;
27    END IF;
28  END LOOP;
29 END;
```

Statement processed.

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	COD_PRODUCTO	VARCHAR2	30			Yes	Yes	
2	NOMBRE	VARCHAR2	30			Yes	Yes	
3	NOMBRE	VARCHAR2	30			Yes	Yes	
4	NOMBRE	VARCHAR2	30			Yes	Yes	
5	PRECIO_PROVEEDOR	NUMBER	22	4	2	Yes		
6	PRECIO_VENTA	NUMBER	22	4	2	Yes		
7	STOCK	NUMBER	22	4	2	Yes		
8	PESO	NUMBER	22	4	2	Yes		
9	VEGANO	CHAR	1			Yes	Yes	
10	CATEGORIA	VARCHAR2	30			Yes	Yes	
11	OF_PROVEEDOR	VARCHAR2	30			Yes	Yes	
12	IND_STU	CHAR	1			Yes	Yes	

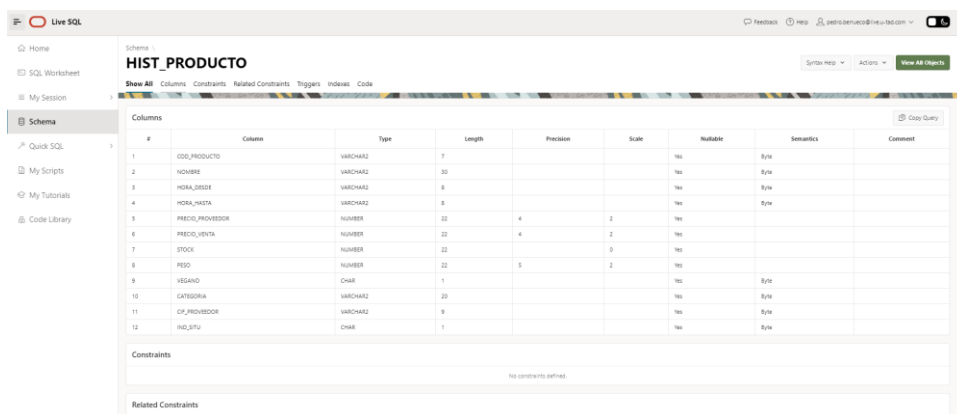
Constraint	Type	Condition	Related Constraint	Related Table	Constraint Columns	On Delete	Status	Last Change	Inherited?
------------	------	-----------	--------------------	---------------	--------------------	-----------	--------	-------------	------------

## OPCION 2:

CREATE TABLE HIST\_PRODUCTO AS SELECT \* FROM PRODUCTO WHERE 1 = 2;

```
57  
58 ✓ CREATE TABLE HIST_PRODUCTO AS  
59 SELECT *  
60 FROM PRODUCTO  
61 WHERE 1 = 2;  
62  
63
```

Table created.



#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	COD_PRODUCTO	VARCHAR2	7			Yes	Byte	
2	NOMBRE	VARCHAR2	30			Yes	Byte	
3	NOMBRE_ORIG	VARCHAR2	8			Yes	Byte	
4	NOMBRE_DEST	VARCHAR2	8			Yes	Byte	
5	PRECIO_PROVEEDOR	NUMBER	22	4	2	Yes		
6	PRECIO_VENTA	NUMBER	22	4	2	Yes		
7	STOCK	NUMBER	22		0	Yes		
8	PESO	NUMBER	22	5	2	Yes		
9	VEGANO	CHAR	1			Yes	Byte	
10	CATEGORIA	VARCHAR2	20			Yes	Byte	
11	CP_PROVEEDOR	VARCHAR2	9			Yes	Byte	
12	IND_SITU	CHAR	1			Yes	Byte	

## EJERCICIO B.

Crea un bloque de código anónimo que realice las siguientes acciones:

- Liste por pantalla los códigos y nombres de los productos que están en situación de baja lógica ( $Ind\_situ = 'B'$ ).
- Elimine de los menús esos mismos productos. Los menús se quedarán solo con el resto de productos que tengan.
- Copie los productos que están de baja lógica a la tabla hist\_producto.
- Al final, si ha copiado al menos un producto, debe mostrar por pantalla cuántos productos ha copiado al histórico (acompañado de un texto indicativo para indicar lo que se está mostrando). Si no ha copiado ningún producto, debe mostrar el mensaje "No se han encontrado productos para copiar". El bloque de código debe tener comentarios que indiquen lo que se está haciendo en cada momento. Una vez implementados todos los apartados, incluye en la respuesta el código completo del bloque de código anónimo que realiza todas las acciones solicitadas.

```

-- Ejercicio B.

DECLARE
  v_cod VARCHAR(20);
  v_nom VARCHAR(20);
  CONTADOR INTEGER := 0;
  -- Se crea un cursor que recorre la consulta solicitada.
  CURSOR recorre IS SELECT * FROM PRODUCTO where IND_SITU = 'B';
BEGIN
  -- Bucle FOR para utilizar el cursor antes declarado.
  FOR c IN RECORRE LOOP
    -- En cada iteración del bucle asignamos valor a las variables que mostrarán el valor de los campos solicitados.
    v_cod := c.COD_PRODUCTO;
    v_nom := c.NOMBRE;
    -- Sacamos por pantalla el valor del nombre y código de producto por cada iteración.
    DBMS_OUTPUT.PUT_LINE('El código del producto es ' || v_cod || ' y el nombre es ' || v_nom);
    -- Borraremos de la tabla MENU_PRODUCTO el producto resultante de cada iteración.
    -- Esto eliminará esos productos de los menús.
    DELETE FROM MENU_PRODUCTO WHERE COD_PRODUCTO = c.COD_PRODUCTO;
    -- Insertamos cada producto en la tabla HIST_PRODUCTO, con todos sus campos.
    INSERT INTO HIST_PRODUCTO VALUES (c.COD_PRODUCTO, c.NOMBRE, c.HORA_DESDE, c.HORA_HASTA, c.PRECIO_PROVEEDOR, c.PRECIO_VENTA, c.STOCK, c.PESO, c.VEGANO, c.CATEGORIA);
    -- Se suma 1 por cada iteración a un contador que usaremos más adelante.
    CONTADOR := CONTADOR + 1;
  END LOOP;

  -- Si la consulta devolvió más de un registro el contador será mayor que 0 y se muestra el número de registros copiados.
  -- Si la consulta no devolvió registros y no se copió ninguna también lo indicará.
  IF CONTADOR > 0 THEN
    DBMS_OUTPUT.PUT_LINE('Se han copiado ' || CONTADOR || ' registros a la BD HIST_PRODUCTO.');
```

## EJERCICIO C.

Después, realiza una prueba completa de ejecución paso a paso (que incluya consultas que consideres necesarias para mostrar el contenido de las tablas antes y después de la ejecución del bloque, así como capturas de pantalla de los resultados), explicando y mostrando en cada paso lo que se está haciendo. • La prueba debe hacerse con la situación inicial de las tablas, es decir, si has ejecutado ya el bloque de código anónimo, debes volver a cargar el script de la base de datos para restaurar la situación de todas las tablas antes de realizar la prueba

Para completar la actividad he creado las consultas siguientes que ejecutaré antes y después de lanzar el PLSQL:

```
-- Productos sobre los que actuara PL/SQL
```

```
SELECT Cod_Producto, Nombre FROM producto WHERE ind_situ = 'B';
```

1  
2  
3

SELECT Cod\_Producto, Nombre FROM producto WHERE ind\_situ = 'B';

COD_PRODUCTO	NOMBRE
HAM-004	Mc Bacon
FRI-002	Patatas gajo
CON-003	Sobre de sacarina

Download CSV

3 rows selected.

--De los productos mostrados arriba, se especifican los incluidos en algún menú.

SELECT \* FROM menu\_producto WHERE COD\_PRODUCTO IN (SELECT Cod\_Producto FROM producto WHERE ind\_situ = 'B');

4  
5  
6  
7  
8  
9

SELECT \* FROM menu\_producto WHERE COD\_PRODUCTO IN (SELECT Cod\_Producto FROM producto WHERE ind\_situ = 'B');

COD_PRODUCTO	NOMBRE
HAM-004	Mc Bacon
FRI-002	Patatas gajo
CON-003	Sobre de sacarina

Download CSV

3 rows selected.

-- Productos de alta en la tabla hist\_producto

SELECT \* FROM HIST\_PRODUCTO;

```
10
11 SELECT * FROM HIST_PRODUCTO;
12
13
```

no data found

--Lanzamos PL/SQL

```
1 DECLARE
2   v_cod VARCHAR(20);
3   v_nom VARCHAR(20);
4   CONTADOR INTEGER := 0;
5   -- Se crea un cursor que recorre la consulta solicitada.
6   CURSOR recorre IS SELECT * FROM PRODUCTO where IND_SITU = 'B';
7 BEGIN
8   -- Bucle FOR para utilizar el cursor antes declarado.
9   FOR c in RECORRE LOOP
10      -- En cada iteración del bucle asignamos valor a las variables que mostaran el valor de los campos solicitados.
11      v_cod := c.COD_PRODUCTO;
12      v_nom := c.NOMBRE;
13      -- Sacamos por pantalla el valor del nombre y código de producto por cada iteración.
14      DBMS_OUTPUT.PUT_LINE('El código del producto es ' || v_cod || ' y el nombre es ' || v_nom );
15   END LOOP;
16
```

Statement processed.  
El código del producto es HAM-004 y el nombre es Mc Bacon  
El código del producto es FRI-002 y el nombre es Patatas gajo  
El código del producto es CON-003 y el nombre es Sobre de sacarina  
Se han copiado 3 registros a la BD HIST\_PRODUCTO.

-- Productos sobre los que actuó PL/SQL (Siguen siendo los mismos)

SELECT Cod\_Producto, Nombre FROM producto WHERE ind\_situ = 'B';

```
33 SELECT Cod_Producto, Nombre FROM producto WHERE ind_situ = 'B';
34
35 SELECT * FROM menu_producto WHERE COD_PRODUCTO IN (SELECT Cod_Producto FROM producto WHERE ind_situ = 'B');
36
37 SELECT * FROM HIST_PRODUCTO;
38
39
```

Statement processed.  
El código del producto es HAM-004 y el nombre es Mc Bacon  
El código del producto es FRI-002 y el nombre es Patatas gajo  
El código del producto es CON-003 y el nombre es Sobre de sacarina  
Se han copiado 3 registros a la BD HIST\_PRODUCTO.

COD_PRODUCTO	NOMBRE
HAM-004	Mc Bacon
FRI-002	Patatas gajo
CON-003	Sobre de sacarina

--De los productos mostrados arriba, se especifican los incluidos en algún menú. (Ya no queda ninguno).

```
SELECT * FROM menu_producto WHERE COD_PRODUCTO IN (SELECT Cod_Producto FROM producto WHERE ind_situ = 'B');
```

```
34
35 SELECT * FROM menu_producto WHERE COD_PRODUCTO IN (SELECT Cod_Producto FROM producto WHERE ind_situ = 'B');
36
37 SELECT * FROM HIST_PRODUCTO;
38
39
```

no data found

-- Productos de alta en la tabla hist\_producto. (Ahora aparecen los 3).

```
SELECT * FROM HIST_PRODUCTO;
```

```
36 SELECT * FROM HIST_PRODUCTO;
37
38
```

COD_PRODUCTO	NOMBRE	HORA_DESDE	HORA_HASTA	PRECIO_PROVEEDOR	PRECIO_VENTA	STOCK	PESO	VEGANO	CATEGORIA	CIF_PROVEEDOR	IND_SITU
HAM-004	Mc Bacon	00:00:00	23:59:59	1.3	5.5	420	180	F	Hamburguesas	A12326787	B
FRI-002	Patatas gajo	00:00:00	23:59:59	.4	3	300	200	F	Acompañantes	A82364522	B
CON-003	Sobre de sacarina	00:00:00	23:59:59	.1	0	900	10	T	Condimentos	A34364367	B

Download CSV

3 rows selected.