

Tarea para BD05.

Enunciado.

Para realizar los ejercicios se utilizarán las tablas creadas para la tarea 4.

¡¡IMPORTANTE!! Antes de realizar estos ejercicios **desmarca la casilla de Autocommit** de la pantalla de Comandos SQL (para que no haga commit automáticamente después de ejecutar cada sentencia) y así tengas la oportunidad de deshacer.

Escribe las sentencias para resolver cada ejercicio (**escribiéndolas con claridad, cada cláusula en una línea**, y en formato que puedan ser ejecutadas en el editor de comandos del SGDB de Oracle Database 11g):

1. Inserta un nuevo cliente y un pedido para este cliente (sin gastos de envío y total con valor 0 de momento)

```
INSERT INTO CLIENTES
(CODIGO, NOMBRE, APELLIDOS, EDAD)
VALUES
('C08', 'Martínez', 'Tabernero', 27);
```

```
INSERT INTO PEDIDOS
(NUM, FECHA, GASTOS_ENVIO, FECHA_PREVISTA, TOTAL, CLIENTE)
VALUES
(15,
TO_DATE('03/15/2025','MM/DD/YYYY'),
0,
TO_DATE('04/15/2025','MM/DD/YYYY'),
0,
'C08');
```

2. Inserta al menos 2 líneas al pedido anterior (cuidado al introducir los datos de los atributos que son calculados a partir de otros datos que se tienen)

```
INSERT INTO LINEAS  
(NUM, NUMPEDIDO, PRODUCTO, CANTIDAD, IMPORTE)  
VALUES  
(1, 15, 1001, 3, 3*50);
```

```
INSERT INTO LINEAS  
(NUM, NUMPEDIDO, PRODUCTO, CANTIDAD, IMPORTE)  
VALUES  
(2, 15, 4003, 2, 2*50);
```

3. Modifica el último pedido introducido para que el importe total tenga el valor que corresponda según los importes de sus líneas.

```
UPDATE PEDIDOS  
SET TOTAL = (  
SELECT SUM(IMPORTE)  
FROM LINEAS  
WHERE NUMPEDIDO = 15  
)  
WHERE NUM = 15;
```

4. Haz un SELECT que muestre los datos de pedido anterior y de sus líneas. Copia aquí el resultado de la consulta. Si todo está bien confirma definitivamente los cambios realizados, si no, repítelos y confírmalos cuando esté todo bien.

```
SELECT
p.NUM,
p.FECHA,
p.GASTOS_ENVIO,
p.FECHA_PREVISTA,
p.TOTAL,
p.CLIENTE,
l.NUM AS LINEA,
l.PRODUCTO,
l.CANTIDAD,
l.IMPORTE
FROM PEDIDOS p
JOIN LINEAS l
ON p.NUM = l.NUMPEDIDO
WHERE p.NUM = 15;
```

Results Explain Describe Saved SQL History										
NUM	FECHA	GASTOS_ENVIO	FECHA_PREVISTA	TOTAL	CLIENTE	LINEA	PRODUCTO	CANTIDAD	IMPORTE	
15	3/15/2025 00:00:00	0	4/15/2025 00:00:00	250	C08	1	1001	3	150	
15	3/15/2025 00:00:00	0	4/15/2025 00:00:00	250	C08	2	4003	2	100	

2 rows returned in 0.01 seconds

Hacemos COMMIT

```
COMMIT;
```

5. Hay que retrasar 30 días las fechas previstas de entrega de los pedidos realizados en marzo de 2025 que incluyan el producto HDD Z

```
UPDATE PEDIDOS p
SET FECHA_PREVISTA = FECHA_PREVISTA + 30
WHERE TO_CHAR(p.FECHA, 'MM/YYYY') = '03/2025'
AND EXISTS (
SELECT 1
FROM LINEAS l
WHERE l.NUMPEDIDO = p.NUM
```

```
AND l.PRODUCTO = 4003  
);
```

6. A todos los pedidos realizados en 2025, que no tengan gastos de envío, se les pondrá 15€ de gastos de envío si su importe total no supera los 100€.

```
UPDATE PEDIDOS  
SET GASTOS_ENVIO = 15  
WHERE EXTRACT(YEAR FROM FECHA) = 2025  
AND (GASTOS_ENVIO IS NULL OR GASTOS_ENVIO = 0)  
AND TOTAL <= 100;
```

7. Baje en un 20% el precio de todos los productos que no se hayan pedido nunca. Indica qué cambios se desharian si ahora ejecutamos ROLLBACK.

```
UPDATE PRODUCTOS  
SET PRECIO = PRECIO * 0.8  
WHERE CODIGO NOT IN (  
SELECT DISTINCT PRODUCTO  
FROM LINEAS  
);
```

8. Crea una tabla (PRODUCTOS_BORRADOS) donde se van a ir guardando los productos que se borren de la tabla PRODUCTOS, debe incluir los datos del producto y la fecha en la que se han borrado.

```
CREATE TABLE PRODUCTOS_BORRADOS (  
CODIGO NUMBER(5,0) PRIMARY KEY,  
NOMBRE VARCHAR2(30) NOT NULL,  
PRECIO NUMBER(7,2) NOT NULL,  
FECHA_BORRADO DATE  
);
```

9. Recuerda que debes guardarlos en la tabla creada en el ejercicio anterior.

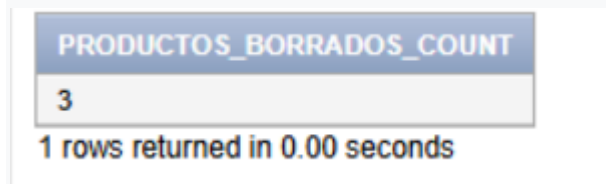
```
INSERT INTO PRODUCTOS_BORRADOS (CODIGO, NOMBRE, PRECIO, FECHA_BORRADO)  
SELECT CODIGO, NOMBRE, PRECIO, SYSDATE  
FROM PRODUCTOS  
WHERE CODIGO NOT IN (  
SELECT DISTINCT PRODUCTO  
FROM LINEAS  
);
```

Borra los productos que no hayan sido pedidos nunca.

```
DELETE FROM PRODUCTOS
WHERE CODIGO NOT IN (
SELECT DISTINCT PRODUCTO
FROM LINEAS
);
```

¿Cuántos se han borrado?.

```
SELECT COUNT(*) AS PRODUCTOS_BORRADOS_COUNT
FROM PRODUCTOS_BORRADOS;
```



PRODUCTOS_BORRADOS_COUNT
3

1 rows returned in 0.00 seconds

Explica si afecta, o no, esta acción a otras tablas y por qué.

No afecta a otras tablas porque la restricción de FOREIGN KEY impide borrar productos usados

10. La cliente M^a PILAR PEREZ BERMEJO nos ha pedido darla de baja. Ejecuta la sentencia que borra a esta cliente. Explica qué ocurre y por qué.

```
DELETE FROM CLIENTES
WHERE NOMBRE = 'Ma Pilar'
AND APELLIDOS = 'Perez Bermejo';
```

Al ejecutar me da este error:

ORA-02292: integrity constraint (BOBOVINO.CLIENTES_FK) violated - child record found

Ocurre porque existe otros registros en otras tablas con referencia a la cliente que intentamos borrar.

11. Ejecuta y justifica las acciones necesarias para borrar definitivamente a la cliente anterior.

Voy a cambiar la CONSTRAINT para que se borren en cascada los pedidos de un cliente si

```
ALTER TABLE PEDIDOS DROP CONSTRAINT CLIENTES_FK;
```

```
ALTER TABLE PEDIDOS ADD CONSTRAINT CLIENTES_FK FOREIGN KEY (CLIENTE)  
REFERENCES CLIENTES (CODIGO)  
ON DELETE CASCADE;
```

Y ahora sí que podemos borrar a la cliente con total normalidad, y eso hace que se borren también sus registros que hacen referencia a esta cliente.

12. Modifica la cantidad del producto de la línea 1 del pedido 10,

```
UPDATE LINEAS  
SET CANTIDAD = 5  
WHERE NUMPEDIDO = 10  
AND NUM = 1;
```

Cambiamos la cantidad a por ejemplo 5

y realiza las actualizaciones necesarias para que los datos de esa línea

```
UPDATE LINEAS  
SET IMPORTE = 5 * (SELECT PRECIO FROM PRODUCTOS WHERE CODIGO = PRODUCTO)  
WHERE NUMPEDIDO = 10  
AND NUM = 1;
```

y del pedido sean correctos teniendo en cuenta la modificación realizada.

```
UPDATE PEDIDOS
SET TOTAL = (SELECT SUM(IMPORTE) FROM LINEAS WHERE NUMPEDIDO = 10)
WHERE NUM = 10;
```

13. Ejecuta la sentencia que modifica el código del producto HDD Y. Explica qué ocurre y por qué.

```
UPDATE PRODUCTOS
SET CODIGO = 5002
WHERE CODIGO = 4002
AND NOMBRE = 'HDD Y';
```

```
ORA-02292: integrity constraint (BOBOVINO.PRODUCTO_FK) violated - child record found
```

Violación de la clave foránea porque aún existen registros con el código antiguo

14. Ejecuta las sentencias necesarias para realizar la actualización anterior en la base de datos (sin hacer cambios con sentencias DDL, solo sentencias DML)

Como no puedo usar DDL, y hemos entrado en un círculo en el cual no me deja actualizar PRODUCTOS ni LINEAS por problemas con child record y parent record, vamos a insertar un nuevo registro que nos ayude a saltar el problema momentaneamente.

```
INSERT INTO PRODUCTOS (CODIGO, NOMBRE, PRECIO)
SELECT 5002, NOMBRE, PRECIO
FROM PRODUCTOS
WHERE CODIGO = 4002
AND NOMBRE = 'HDD Y';
```

Ahora podemos actualizar LINEAS

```
UPDATE LINEAS
SET PRODUCTO = 5002
WHERE PRODUCTO = 4002;
```

Y borramos productos

```
DELETE FROM PRODUCTOS
```

```
WHERE CODIGO = 4002  
AND NOMBRE = 'HDD Y';
```

Así, hemos hecho el UPDATE sin hacer un UPDATE directamente.