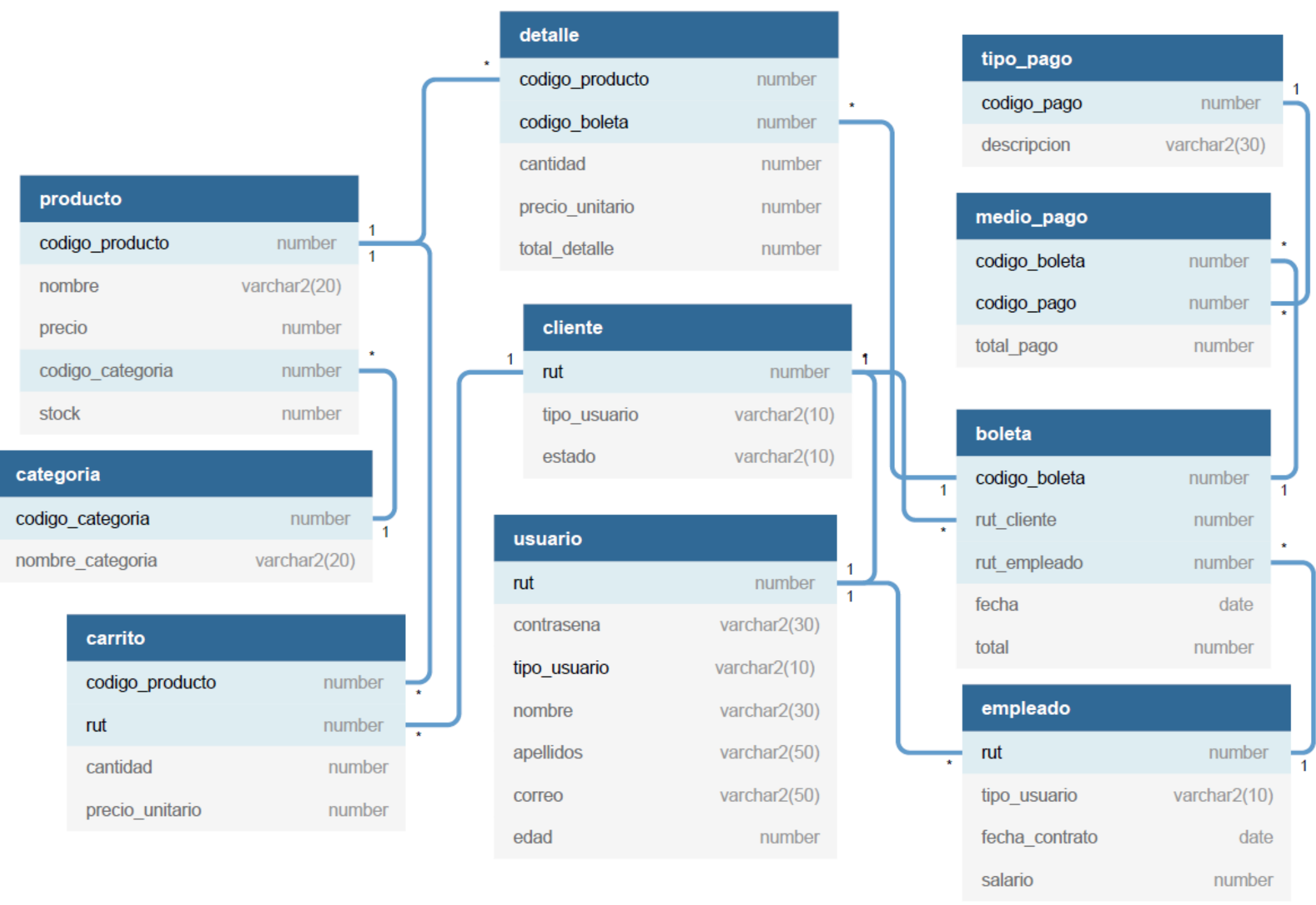


Procedimientos, cursores y triggers

Programa de hoy

- Creación de tablas utilizando procedimientos
- Cursores en procedimientos
- Triggers



Creación del modelo

Hasta el momento, la creación del modelo de datos tiene la siguiente estructura.

SECCIÓN DE LOS **DROP TABLE**

SECCIÓN DE LOS **CREATE TABLE**

SECCIÓN DE LOS **ALTER TABLE**

Creación del modelo

Hasta el momento, la creación del modelo de datos tiene la siguiente estructura.

SECCIÓN DE LOS **DROP TABLE**

SECCIÓN DE LOS **CREATE TABLE**

SECCIÓN DE LOS **ALTER TABLE**

Hoy exploraremos otra opción que PL/SQL permite realizar. Ejecutar instrucciones del lenguaje de definición de datos utilizando procedimientos.

Lenguaje de definición de datos

```
CREATE TABLE | ROLE | PROCEDURE | FUNCTION | TRIGGER | VIEW | USER |
```

```
DROP TABLE | ROLE | PROCEDURE | FUNCTION | TRIGGER | VIEW | USER |
```

```
ALTER TABLE | ROLE | PROCEDURE | FUNCTION | TRIGGER | VIEW | USER |
```

Creación del modelo

Dicho lo anterior, podemos utilizar procedimientos que permitan crear, modificar o eliminar nuestro modelo de datos.

Creación del modelo

Dicho lo anterior, podemos utilizar procedimientos que permitan crear, modificar o eliminar nuestro modelo de datos.

Por ejemplo, crear un procedimiento que cree las tablas del modelo presentado anteriormente.

Creación del modelo

Dicho lo anterior, podemos utilizar procedimientos que permitan crear, modificar o eliminar nuestro modelo de datos.

Por ejemplo, crear un procedimiento que cree las tablas del modelo presentado anteriormente.

Claramente al ser demasiadas tablas, crear un procedimiento que se encargue de crearlas todas llevará algún tiempo. Pero para comenzar, veamos un pequeño ejemplo.

Creación del modelo

Supongamos el ejemplo de una tabla llamada persona que posee 2 atributos. Un rut y nombre.

```
CREATE TABLE PERSONA (  
    RUT NUMBER,  
    NOMBRE VARCHAR2 (30)  
)
```

Creación del modelo

Supongamos el ejemplo de una tabla llamada persona que posee 2 atributos. Un rut y nombre.

```
CREATE OR REPLACE PROCEDURE CREAM_PERSONA
IS
    TABLA_PRUEBA VARCHAR2(1000) := 'CREATE TABLE PERSONA (
                                RUT NUMBER,
                                NOMBRE VARCHAR2(30)
                                )';

BEGIN
    EXECUTE IMMEDIATE TABLA_PRUEBA;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('TABLA CREADA CON ÉXITO');
    EXCEPTION
        WHEN OTHERS THEN
            IF SQLCODE = -955 THEN
                DBMS_OUTPUT.PUT_LINE('LA TABLA PERSONA YA ESTÁ CREADA');
                ROLLBACK;
            ELSE
                DBMS_OUTPUT.PUT_LINE('ERROR: ' || SQLERRM);
            END IF;
END;
```

Creación del modelo

Supongamos otro ejemplo de un procedimiento que agrega, modifique o elimine una columna de una tabla.

Creación del modelo

Supongamos otro ejemplo de un procedimiento que agrega, modifique o elimine una columna de una tabla.

El procedimiento puede ser genérico para alterar cualquier tabla y columna. De esta forma, ejecutar la llamada del procedimiento puede permitir alterar cualquier tabla.

```

CREATE OR REPLACE PROCEDURE ALTERAR_TABLA(
    NOMBRE_TABLA IN VARCHAR2,
    ACCION IN VARCHAR2,
    TIPO_DATO IN VARCHAR2,
    NOMBRE_COLUMNNA IN VARCHAR2
)
IS
    ALTER_TEXTO VARCHAR2(1000);
    OPCION_INVALIDA EXCEPTION;
BEGIN
    CASE ACCION
        WHEN 'A' THEN
            ALTER_TEXTO := 'ALTER TABLE ' || NOMBRE_TABLA || ' ADD ' || NOMBRE_COLUMNNA || ' ' || TIPO_DATO;
        WHEN 'M' THEN
            ALTER_TEXTO := 'ALTER TABLE ' || NOMBRE_TABLA || ' MODIFY ' || NOMBRE_COLUMNNA || ' ' || TIPO_DATO;
        WHEN 'E' THEN
            ALTER_TEXTO := 'ALTER TABLE ' || NOMBRE_TABLA || ' DROP COLUMN ' || NOMBRE_COLUMNNA;
        ELSE
            RAISE OPCION_INVALIDA;
    END CASE;
    EXECUTE IMMEDIATE ALTER_TEXTO;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('LA TABLA ' || NOMBRE_TABLA || ' SE MODIFICÓ');
    EXCEPTION
        WHEN OPCION_INVALIDA THEN
            DBMS_OUTPUT.PUT_LINE('LA OPCION INGRESADA NO ES VÁLIDA: SOLO A-M-E PERMITIDO');
            ROLLBACK;
        WHEN OTHERS THEN
            CASE SQLCODE
                WHEN -942 THEN
                    DBMS_OUTPUT.PUT_LINE('LA TABLA QUE INTENTA MODIFICAR NO EXISTE');
                WHEN -904 THEN
                    DBMS_OUTPUT.PUT_LINE('LA COLUMNA QUE INTENTA MODIFICAR NO EXISTE');
                WHEN -902 THEN
                    DBMS_OUTPUT.PUT_LINE('EL TIPO DE DATO INGRESADO NO ES VALIDO');
                ELSE
                    DBMS_OUTPUT.PUT_LINE('ERROR: ' || SQLERRM);
            END CASE;
            ROLLBACK;
    END;

```

Cursores

Como bien vimos anteriormente, la finalidad de un cursor es recorrer las filas de una sentencia SELECT.

Cursores

Como bien vimos anteriormente, la finalidad de un cursor es recorrer las filas de una sentencia SELECT.

El cursor siempre se debe recorrer utilizando una estructura iterativa o LOOP. Tal como está ejemplificado en la sintaxis.

Cursores (Sintaxis)

Creación del cursor:

```
CURSOR NOMBRE_CURSOR  
IS SENTENCIA_SELECT;
```

Recorrer el cursor:

```
OPEN NOMBRE_CURSOR; --ABRIR EL CURSOR  
LOOP --RECORRER EL CURSOR USANDO LOOP  
    FETCH NOMBRE_CURSOR INTO COLUMNAS  
        --MANIPULAR CADA FILA DEL SELECT  
    EXIT WHEN NOMBRE_CURSOR%NOTFOUND; --CONDICION SALIDA CURSOR  
END LOOP;  
CLOSE NOMBRE_CURSOR; --CERRAR EL CURSOR
```

* Esto va dentro de un bloque anónimo, procedimiento o función.

Cursores (Ejemplo)

Supongamos que nuestra tienda necesita encargar productos a un proveedor cuando el stock de productos es inferior a 5 unidades.

Cursores (Ejemplo)

Supongamos que nuestra tienda necesita encargar productos a un proveedor cuando el stock de productos es inferior a 5 unidades.

PRODUCTO				
CODIGO_PRODUCTO	NOMBRE	PRECIO	CODIGO_CATEGORIA	STOCK
1	NOTEBOOK LENOVO	\$380.000	1	20
2	CELULAR MOTOROLA	\$110.000	2	15
3	AUDIFONOS MACROTEL	\$3.500	3	12
4	NOTEBOOK SAMSUNG	\$500.000	1	3
5	MONITOR 17" AOC	\$67.990	4	80
6	MONITOR 21" DELL	\$81.990	4	20
7	NOTEBOOK HP	\$280.990	1	1
8	CELULAR IPHONE 6S	\$560.000	2	6
9	CELULAR LG	\$450.000	2	7
10	MOUSE BLUETOOTH	\$28.990	5	9
11	MONITOR 17" SAMSUNG	\$150.000	4	7
12	MONITOR 17" LENOVO	\$250.000	4	7
13	MONITOR 17" LG	\$125.000	4	4
14	MONITOR 15" LENOVO	\$200.000	4	3
15	MOTO G 2013	\$130.000	2	45
16	MOTO G 2015	\$180.000	2	32
17	MOTO X PLAY	\$370.000	2	12
18	IPAD MINI 2	\$200.000	6	4
19	IPAD AIR	\$260.000	6	2
20	NOTEBOOK MSI	\$850.000	1	4
21	MOUSE MICROSOFT	\$8.500	5	5

Cursores (Ejemplo)

Supongamos que nuestra tienda necesita encargar productos a un proveedor cuando el stock de productos es inferior a 5 unidades.

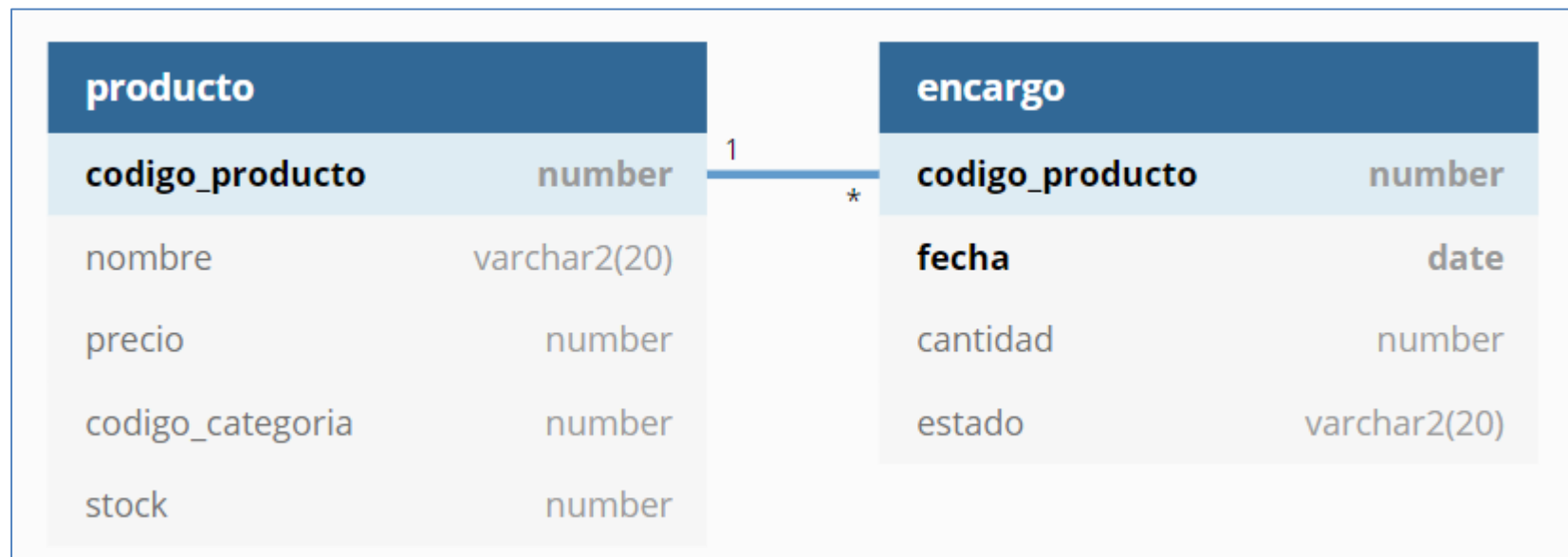
PRODUCTO				
CODIGO_PRODUCTO	NOMBRE	PRECIO	CODIGO_CATEGORIA	STOCK
1	NOTEBOOK LENOVO	\$380.000	1	20
2	CELULAR MOTOROLA	\$110.000	2	15
3	AUDIFONOS MACROTEL	\$3.500	3	12
4	NOTEBOOK SAMSUNG	\$500.000	1	3
5	MONITOR 17" AOC	\$67.990	4	80
6	MONITOR 21" DELL	\$81.990	4	20
7	NOTEBOOK HP	\$280.990	1	1
8	CELULAR IPHONE 6S	\$560.000	2	6
9	CELULAR LG	\$450.000	2	7
10	MOUSE BLUETOOTH	\$28.990	5	9
11	MONITOR 17" SAMSUNG	\$150.000	4	7
12	MONITOR 17" LENOVO	\$250.000	4	7
13	MONITOR 17" LG	\$125.000	4	4
14	MONITOR 15" LENOVO	\$200.000	4	3
15	MOTO G 2013	\$130.000	2	45
16	MOTO G 2015	\$180.000	2	32
17	MOTO X PLAY	\$370.000	2	12
18	IPAD MINI 2	\$200.000	6	4
19	IPAD AIR	\$260.000	6	2
20	NOTEBOOK MSI	\$850.000	1	4
21	MOUSE MICROSOFT	\$8.500	5	5

Cursores (Ejemplo)

Este encargo se realiza agregando los productos necesarios a una nueva tabla llamada “encargo”, la cual está conectada a la tabla “producto” que ya teníamos.

Cursores (Ejemplo)

Este encargo se realiza agregando los productos necesarios a una nueva tabla llamada “encargo”, la cual está conectada a la tabla “producto” que ya teníamos.



Cursores (Ejemplo)

Este encargo se realiza agregando los productos necesarios a una nueva tabla llamada “encargo”, la cual está conectada a la tabla “producto” que ya teníamos.

Ahora la pregunta es, ¿cómo agregar esos productos a la tabla encargo?.

Cursores (Ejemplo)

Este encargo se realiza agregando los productos necesarios a una nueva tabla llamada “encargo”, la cual está conectada a la tabla “producto” que ya teníamos.

Ahora la pregunta es, ¿cómo agregar esos productos a la tabla encargo?.

La forma tradicional o que podrían realizar es hacer un SELECT obteniendo los código de productos con stock inferior a 5.

Cursores (Ejemplo)

Este encargo se realiza agregando los productos necesarios a una nueva tabla llamada “encargo”, la cual está conectada a la tabla “producto” que ya teníamos.

Ahora la pregunta es, ¿cómo agregar esos productos a la tabla encargo?.

La forma tradicional o que podrían realizar es hacer un SELECT obteniendo los código de productos con stock inferior a 5.

```
SELECT CODIGO_PRODUCTO  
FROM PRODUCTO  
WHERE STOCK < 5;
```

Cursores (Ejemplo)

Para el ejemplo, sabemos que son 7 productos y es totalmente abordable ingresarlos a “mano”. Pero qué sucede si son 50 o 100 productos los que se deben ingresar.

Cursores (Ejemplo)

Para el ejemplo, sabemos que son 7 productos y es totalmente abordable ingresarlos a “mano”. Pero qué sucede si son 50 o 100 productos los que se deben ingresar.

PRODUCTO				
CODIGO_PRODUCTO	NOMBRE	PRECIO	CODIGO_CATEGORIA	STOCK
4	NOTEBOOK SAMSUNG	\$500.000	1	3
7	NOTEBOOK HP	\$280.990	1	1
13	MONITOR 17" LG	\$125.000	4	4
14	MONITOR 15" LENOVO	\$200.000	4	3
18	IPAD MINI 2	\$200.000	6	4
19	IPAD AIR	\$260.000	6	2
20	NOTEBOOK MSI	\$850.000	1	4

Cursores (Ejemplo)

Para el ejemplo, sabemos que son 7 productos y es totalmente abordable ingresarlos a “mano”. Pero qué sucede si son 50 o 100 productos los que se deben ingresar.

PRODUCTO				
CODIGO_PRODUCTO	NOMBRE	PRECIO	CODIGO_CATEGORIA	STOCK
4	NOTEBOOK SAMSUNG	\$500.000	1	3
7	NOTEBOOK HP	\$280.990	1	1
13	MONITOR 17" LG	\$125.000	4	4
14	MONITOR 15" LENOVO	\$200.000	4	3
18	IPAD MINI 2	\$200.000	6	4
19	IPAD AIR	\$260.000	6	2
20	NOTEBOOK MSI	\$850.000	1	4

Para resolver este problema, utilizamos un cursor. El cursor se encargará de analizar fila por fila los productos, considerando solamente los que tengan un stock inferior a 5.

Cursores (Ejemplo)

Para comenzar, creamos la tabla llamada “encargo”.

```
CREATE TABLE ENCARGO (  
    CODIGO_PRODUCTO NUMBER,  
    FECHA DATE,  
    CANTIDAD NUMBER,  
    ESTADO VARCHAR2(30) DEFAULT 'ACTIVO'  
);
```

Cursores (Ejemplo)

Para comenzar, creamos la tabla llamada “encargo”.

```
CREATE TABLE ENCARGO (  
    CODIGO_PRODUCTO NUMBER,  
    FECHA DATE,  
    CANTIDAD NUMBER,  
    ESTADO VARCHAR2(30) DEFAULT 'ACTIVO'  
);
```

Luego, creamos la clave primaria de la tabla “encargo” y la clave foránea para unirla con la tabla “producto”.

Cursores (Ejemplo)

Para comenzar, creamos la tabla llamada “encargo”.

```
CREATE TABLE ENCARGO (  
    CODIGO_PRODUCTO NUMBER,  
    FECHA DATE,  
    CANTIDAD NUMBER,  
    ESTADO VARCHAR2(30) DEFAULT 'ACTIVO'  
);
```

Luego, creamos la clave primaria de la tabla “encargo” y la clave foránea para unirla con la tabla “producto”.

```
ALTER TABLE ENCARGO ADD CONSTRAINT PK_ENCARGO PRIMARY KEY(CODIGO_PRODUCTO, FECHA);  
ALTER TABLE ENCARGO ADD CONSTRAINT FK_ENCARGO_PRODUCTO  
FOREIGN KEY(CODIGO_PRODUCTO) REFERENCES PRODUCTO(CODIGO_PRODUCTO);
```

Cursores (Ejemplo)

Ahora, creamos un procedimiento llamado “verificar_stock”. El procedimiento tiene solamente dos parámetros de salida. Los dos parámetros de salida corresponden a las variables resultado y mensaje. Esto nos indicará si el proceso fue exitoso o no más adelante.

Cursores (Ejemplo)

Ahora, creamos un procedimiento llamado “verificar_stock”. El procedimiento tiene solamente dos parámetros de salida. Los dos parámetros de salida corresponden a las variables resultado y mensaje. Esto nos indicará si el proceso fue exitoso o no más adelante.

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
  
BEGIN  
  
END;
```

Cursores (Ejemplo)


Con la configuración inicial del procedimiento, es necesario crear el cursor de los productos. De la tabla producto solo nos interesa el código de producto y el stock de aquel producto. Por lo que la creación del cursor sería solamente:

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

Cursores (Ejemplo)

Con la configuración inicial del procedimiento, es necesario crear el cursor de los productos. De la tabla producto solo nos interesa el código de producto y el stock de aquel producto. Por lo que la creación del cursor sería solamente:

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
 CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
BEGIN  
  
END;
```

Cursores (Ejemplo)

El cursor “PRODUCTOS” declarado en la instrucción anterior posee la siguiente información de la tabla productos.

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

Lo que ahora queda por hacer es abrir el cursor, recorrerlo fila por fila y agregar a la tabla “encargo” los productos con stock inferior a 5.

Cursores (Ejemplo)

El cursor “PRODUCTOS” declarado en la instrucción anterior posee la siguiente información de la tabla productos.

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

Lo que ahora queda por hacer es abrir el cursor, recorrerlo fila por fila y agregar a la tabla “encargo” los productos con stock inferior a 5.
Pero antes de llegar a la resolución de este problema, veamos como funciona el cursor.

Cursores (Explicación)

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

Cada vez que el procedimiento es llamado, se declara el cursor PRODUCTOS.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

El cursor al declararse, almacena el resultado obtenido por la consulta asociada.

Cursores (Explicación)

CONSULTA ASOCIADA

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

El cursor al declararse, almacena el resultado obtenido por la consulta asociada.

Cursores (Explicación)

CONSULTA ASOCIADA

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

El cursor al declararse, almacena el resultado obtenido por la consulta asociada.

El resultado de esa “consulta” está dentro de la variable PRODUCTOS.

Cursores (Explicación)



PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

El cursor al declararse, almacena el resultado obtenido por la consulta asociada.

El resultado de esa “consulta” está dentro de la variable PRODUCTOS.

Cursores (Explicación)



PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5



```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

El cursor al declararse, almacena el resultado obtenido por la consulta asociada.

El resultado de esa “consulta” está dentro de la variable PRODUCTOS.

Lo que equivaldría a la tabla de la izquierda.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

El cursor al declararse, almacena el resultado obtenido por la consulta asociada.

El resultado de esa “consulta” está dentro de la variable PRODUCTOS.

Lo que equivaldría a la tabla de la izquierda.

Entonces, para solo tomar en cuenta los productos con STOCK inferior a 5, necesitamos recorrer esta tabla fila por fila.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

```
OPEN PRODUCTOS;  
FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
DBMS_OUTPUT.PUT_LINE('PRODUCTO: '||CODIGO_AUX||' - STOCK: '||STOCK_AUX);  
CLOSE PRODUCTOS;
```

Para recorrer fila por fila, se utiliza la instrucción **FETCH**. Esta instrucción permite obtener una fila o elemento del cursor.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

```
OPEN PRODUCTOS;
FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
DBMS_OUTPUT.PUT_LINE('PRODUCTO: '||CODIGO_AUX||' - STOCK: '||STOCK_AUX);
CLOSE PRODUCTOS;
```

Para recorrer fila por fila, se utiliza la instrucción FETCH. Esta instrucción permite obtener una fila o elemento del cursor.

Si compilamos el procedimiento de abajo y lo ejecutamos, solo podemos obtener la primera fila del cursor.

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
    DBMS_OUTPUT.PUT_LINE('PRODUCTO: '||CODIGO_AUX||' - STOCK: '||STOCK_AUX);
    CLOSE PRODUCTOS;

END;
```

CURSOR 1

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

```
OPEN PRODUCTOS;  
FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
CLOSE PRODUCTOS;
```

Para recorrer fila por fila, se utiliza la instrucción FETCH. Esta instrucción permite obtener una fila o elemento del cursor.

Si compilamos el procedimiento de abajo y lo ejecutamos, solo podemos obtener la primera fila del cursor.

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
    DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
    CLOSE PRODUCTOS;  
END;
```

CURSOR 1

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;

CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    LOOP
        EXIT WHEN PRODUCTOS%NOTFOUND;
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
    END LOOP;
    CLOSE PRODUCTOS;
END;
```

CURSOR 2

Si queremos recorrer TODAS las filas del cursor, necesitamos utilizar una instrucción FOR, WHILE o LOOP. Con la ayuda de alguna de estas instrucciones podemos iterar sobre un cursor para obtener todos sus elementos.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;

CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    LOOP
        EXIT WHEN PRODUCTOS%NOTFOUND;
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
    END LOOP;
    CLOSE PRODUCTOS;
END;
```

CURSOS 2

Si queremos recorrer TODAS las filas del cursor, necesitamos utilizar una instrucción FOR, WHILE o LOOP. Con la ayuda de alguna de estas instrucciones podemos iterar sobre un cursor para obtener todos sus elementos.

Además, siempre que iteremos sobre un cursor, debemos verificar cuando se acaban las filas. Para esto, podemos utilizar la instrucción NOTFOUND que arroja el valor TRUE cuando no hay más filas.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;

CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    LOOP
        EXIT WHEN PRODUCTOS%NOTFOUND;
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
    END LOOP;
    CLOSE PRODUCTOS;
END;
```

CURSOR 2

Si queremos recorrer TODAS las filas del cursor, necesitamos utilizar una instrucción FOR, WHILE o LOOP. Con la ayuda de alguna de estas instrucciones podemos iterar sobre un cursor para obtener todos sus elementos.

Además, siempre que iteremos sobre un cursor, debemos verificar cuando se acaban las filas. Para esto, podemos utilizar la instrucción NOTFOUND que arroja el valor TRUE cuando no hay más filas.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

Para el problema inicial del ejemplo sobre agregar a la tabla “encargo” los productos con STOCK < 5. Podemos modificar la parte central del código anterior agregando una restricción sobre el STOCK de cada producto.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

Para el problema inicial del ejemplo sobre agregar a la tabla “encargo” los productos con STOCK < 5. Podemos modificar la parte central del código anterior agregando una restricción sobre el STOCK de cada producto.

```

OPEN PRODUCTOS;
LOOP
    EXIT WHEN PRODUCTOS%NOTFOUND;
    FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
    DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
    IF STOCK_AUX < 5 THEN
        INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)
        VALUES (CODIGO_AUX, SYSDATE, 20);
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO <----');
    ELSE
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');
    END IF;
END LOOP;

```

CURSOR 3

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
```

Para el problema inicial del ejemplo sobre agregar a la tabla “encargo” los productos con STOCK < 5. Podemos modificar la parte central del código anterior agregando una restricción sobre el STOCK de cada producto.

```
OPEN PRODUCTOS;
LOOP
    EXIT WHEN PRODUCTOS%NOTFOUND;
    FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
    DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
    IF STOCK_AUX < 5 THEN
        INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)
        VALUES (CODIGO_AUX, SYSDATE, 20);
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO <----');
    ELSE
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');
    END IF;
END LOOP;
```

CURSOR 3

Cada vez que el procedimiento se ejecute, recorrerá el cursor y consultará si el STOCK < 5. En caso de cumplir la condición, agregará una tupla a la tabla encargo indicando el código del producto y una cantidad determinada por defecto.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```

CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    LOOP
        EXIT WHEN PRODUCTOS%NOTFOUND;
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
        IF STOCK_AUX < 5 THEN
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)
            VALUES (CODIGO_AUX, SYSDATE, 20);
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');
        ELSE
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');
        END IF;
    END LOOP;
    CLOSE PRODUCTOS;
END;

```

CURSOS 3

Cursores (Explicación)

Para ejecutar el procedimiento, utilizamos un bloque anónimo:

```
DECLARE  
    RESULTADO VARCHAR2 (100) ;  
    MENSAJE VARCHAR2 (100) ;  
BEGIN  
    VERIFICAR_STOCK (RESULTADO, MENSAJE) ;  
    DBMS_OUTPUT.PUT_LINE (RESULTADO) ;  
    DBMS_OUTPUT.PUT_LINE (MENSAJE) ;  
END;
```

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOR 3



Cuando se ejecuta el procedimiento, primero se crea el cursor.


Cursores (Explicación)



PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```


CURSOR 3



Cuando se ejecuta el procedimiento, primero se crea el cursor. Luego, se crean las variables de apoyo para extraer las columnas del cursor.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;   
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Luego, se abre el cursor para poder comenzar a utilizarlo.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```

CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    LOOP ←
        EXIT WHEN PRODUCTOS%NOTFOUND;
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
        IF STOCK_AUX < 5 THEN
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)
            VALUES (CODIGO_AUX, SYSDATE, 20);
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');
        ELSE
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');
        END IF;
    END LOOP;
    CLOSE PRODUCTOS;
END;


```

CURSOS 3

Luego, se abre el cursor para poder comenzar a utilizarlo.
Y comienza la iteración...

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;   
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Aquí pregunta si el cursor está vacío. Como no lo está sigue a la siguiente línea.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
1	20
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.

SQLDEVELOPER: PRODUCTO: 1 – STOCK: 20

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.

SQLDEVELOPER: PRODUCTO: 1 – STOCK: 20

El STOCK es menor a 5?

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.


SQLDEVELOPER: PRODUCTO: 1 – STOCK: 20

El STOCK es menor a 5?, NO

SQLDEVELOPER: PRODUCTO: 1 DESCARTADO

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP   
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Sigue la iteración....

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```

CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(
    RESULTADO OUT VARCHAR2,
    MENSAJE OUT VARCHAR2
)
IS
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;
    CODIGO_AUX NUMBER;
    STOCK_AUX NUMBER;
BEGIN
    OPEN PRODUCTOS;
    LOOP
        EXIT WHEN PRODUCTOS%NOTFOUND;
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);
        IF STOCK_AUX < 5 THEN
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)
            VALUES (CODIGO_AUX, SYSDATE, 20);
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');
        ELSE
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');
        END IF;
    END LOOP;
    CLOSE PRODUCTOS;
END;


```

CURSOR 3

Pregunta nuevamente si el cursor está vacío. Como no lo está sigue a la siguiente línea.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
2	15
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 2 – STOCK: 15

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 2 – STOCK: 15

El STOCK es menor a 5?

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor


SQLDEVELOPER: PRODUCTO: 2 – STOCK: 15

El STOCK es menor a 5?, NO

SQLDEVELOPER: PRODUCTO: 2 DESCARTADO

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP   
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Sigue la iteración....

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;   
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Pregunta nuevamente si el cursor está vacío. Como no lo está sigue a la siguiente línea.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
3	12
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 3 – STOCK: 12

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.

SQLDEVELOPER: PRODUCTO: 3 – STOCK: 12

El STOCK es menor a 5?

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos un elemento del cursor.


SQLDEVELOPER: PRODUCTO: 3 – STOCK: 12

El STOCK es menor a 5?, NO

SQLDEVELOPER: PRODUCTO: 3 DESCARTADO

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP   
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Sigue la iteración....

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;   
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Pregunta nuevamente si el cursor está vacío. Como no lo está sigue a la siguiente línea.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
4	3
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 4 – STOCK: 3

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 4 – STOCK: 3

El STOCK es menor a 5?

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOR 3

Obtenemos el siguiente elemento del cursor


SQLDEVELOPER: PRODUCTO: 4 – STOCK: 3

El STOCK es menor a 5?, SI

SQLDEVELOPER: PRODUCTO: 4 INGRESADO

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
5	80
6	20
7	1
8	6
9	7
10	9
11	7
12	7
13	4
14	3
15	45
16	32
17	12
18	4
19	2
20	4
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP   
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Y así seguimos iterando un buen rato...

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;   
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOR 3

Pregunta nuevamente si el cursor está vacío. Como no lo está sigue a la siguiente línea.

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK
21	5


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;   
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 21 – STOCK: 5

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor

SQLDEVELOPER: PRODUCTO: 21 – STOCK: 5

El STOCK es menor a 5?

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Obtenemos el siguiente elemento del cursor


SQLDEVELOPER: PRODUCTO: 21 – STOCK: 5

El STOCK es menor a 5?, NO

SQLDEVELOPER: PRODUCTO: 21 DESCARTADO

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP   
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO (CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Sigue la iteración..

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK


```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;   
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;  
END;
```

CURSOS 3

Pregunta nuevamente si el cursor está vacío. Ahora sí lo está, por lo tanto, sale de la iteración

Cursores (Explicación)

PRODUCTO	
CODIGO_PRODUCTO	STOCK

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' - STOCK: ' || STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' INGRESADO');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: ' || CODIGO_AUX || ' DESCARTADO');  
        END IF;  
    END LOOP;  
    CLOSE PRODUCTOS;   
END;
```

CURSOS 3

Pregunta nuevamente si el cursor está vacío. Ahora sí lo está, por lo tanto, sale de la iteración, cierra el cursor y finaliza la ejecución del procedimiento.

Cursores (Explicación)

```
CREATE OR REPLACE PROCEDURE VERIFICAR_STOCK(  
    RESULTADO OUT VARCHAR2,  
    MENSAJE OUT VARCHAR2  
)  
IS  
    CURSOR PRODUCTOS IS SELECT CODIGO_PRODUCTO, STOCK FROM PRODUCTO;  
    CODIGO_AUX NUMBER;  
    STOCK_AUX NUMBER;  
    CONTADOR NUMBER := 0;  
BEGIN  
    OPEN PRODUCTOS;  
    LOOP  
        EXIT WHEN PRODUCTOS%NOTFOUND;  
        FETCH PRODUCTOS INTO CODIGO_AUX, STOCK_AUX;  
        DBMS_OUTPUT.PUT_LINE('PRODUCTO: '||CODIGO_AUX||' - STOCK: '||STOCK_AUX);  
        IF STOCK_AUX < 5 THEN  
            LOCK TABLE ENCARGO IN ROW EXCLUSIVE MODE;  
            INSERT INTO ENCARGO(CODIGO_PRODUCTO, FECHA, CANTIDAD)  
            VALUES (CODIGO_AUX, SYSDATE, 20);  
            COMMIT;  
            RESULTADO := 'TRUE';  
            CONTADOR := CONTADOR + 1;  
            MENSAJE := CONTADOR||' PRODUCTOS INGRESADOS';  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: '||CODIGO_AUX||' INGRESADO <-----');  
        ELSE  
            DBMS_OUTPUT.PUT_LINE('PRODUCTO: '||CODIGO_AUX||' DESCARTADO');  
        END IF;  
    END LOOP;  
    EXCEPTION  
        WHEN OTHERS THEN  
            RESULTADO := 'FALSE';  
            MENSAJE := 'SE ALCANZARON INGRESAR '||CONTADOR||' PRODUCTOS Y SE PRODUJO UN ERROR - ERROR:'||SQLCODE;  
            ROLLBACK;  
END;
```

CURSOR 4

Trigger

Un trigger o disparador en español, es otro tipo de bloque almacenado asociado a una tabla o una base de datos.

Trigger

Un trigger o disparador en español, es otro tipo de bloque almacenado asociado a una tabla o una base de datos.

Su funcionamiento es en base a eventos sobre una vista, tabla o una base de datos específica. De acuerdo al evento especificado, el bloque se ejecuta de forma automática.

Trigger

Estos tipos de bloques son útiles para realizar cálculos, transformaciones, verificaciones, etc sobre los eventos de inserción, modificación y eliminación.

Trigger

Estos tipos de bloques son útiles para realizar cálculos, transformaciones, verificaciones, etc sobre los eventos de inserción, modificación y eliminación.

Los triggers no requieren especificar bloqueo de tablas, commit o rollback. La ejecución de estos comandos es automática.

Trigger (Sintaxis)

```
CREATE OR REPLACE TRIGGER NOMBRE_TRIGGER
{ BEFORE | AFTER }
{ DELETE | INSERT | UPDATE }
[ OR { DELETE | INSERT | UPDATE } ]
ON NOMBRE_TABLA
FOR EACH ROW [ WHEN (CONDITION) ]
DECLARE
    --DECLARACION DE VARIABLES
BEGIN
    --SENTENCIAS
[ EXCEPTION ]
    --SENTENCIAS DE CONTROL
END;
```

Trigger

Los triggers se pueden activar antes o después de la ejecución de un evento asociado a una tabla.

Trigger

Los triggers se pueden activar antes o después de la ejecución de un evento asociado a una tabla.

El asociar un trigger a una tabla, significa que puede trabajar con sus columnas al momento de una inserción, modificación o eliminación.

Trigger

Los triggers se pueden activar antes o después de la ejecución de un evento asociado a una tabla.

El asociar un trigger a una tabla, significa que puede trabajar con sus columnas al momento de una inserción, modificación o eliminación.

La manipulación de las columnas dentro del trigger utilizan los prefijos :NEW y :OLD seguido luego por el nombre de la columna sobre la cual se quiere trabajar.

*explicar en pizarra.

Trigger (Ejemplo)

Un ejemplo sencillo de trigger puede ser estandarizar todo el texto que se inserta sobre una tabla dejándolo en mayúsculas. Esto para tener un formato único de strings o texto sobre la base de datos.

Trigger (Ejemplo)

Un ejemplo sencillo de trigger puede ser estandarizar todo el texto que se inserta sobre una tabla dejándolo en mayúsculas. Esto para tener un formato único de strings o texto sobre la base de datos.

Para esto, debemos especificar la tabla sobre la cual actuará el trigger y el evento que deberá ocurrir para su “gatillación” o activación.

Trigger (Ejemplo)

```
CREATE OR REPLACE TRIGGER MAYUSCULA_PRODUCTOS
BEFORE INSERT ON PRODUCTO
FOR EACH ROW
DECLARE
    --DECLARACION VARIABLES
BEGIN
    :NEW.NOMBRE := UPPER(:NEW.NOMBRE);
END;
```

El trigger se encarga de transformar el nombre del producto a mayúsculas antes de realizar un insert sobre la tabla producto.

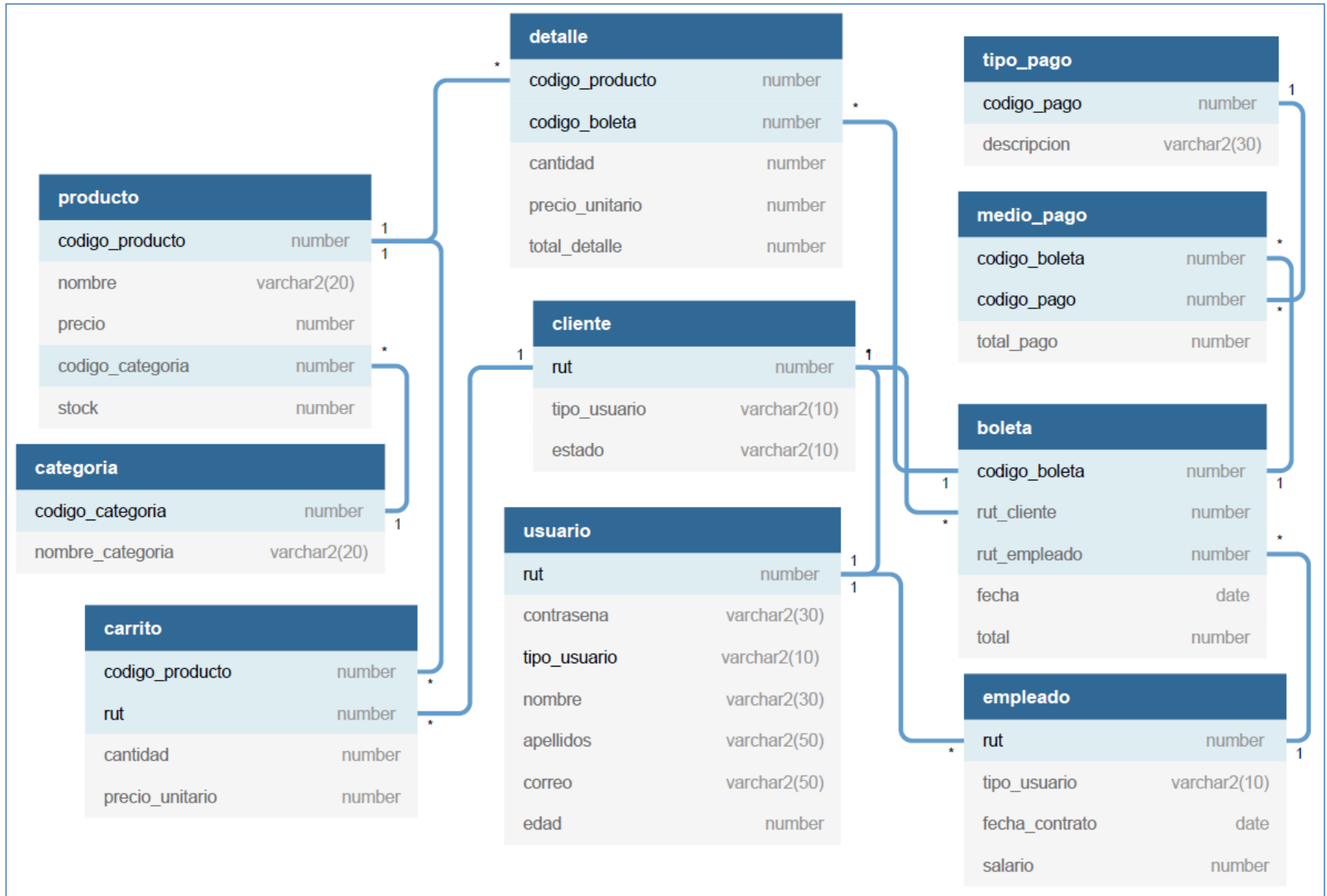
Trigger (Ejemplo)

```
CREATE OR REPLACE TRIGGER MAYUSCULA_PRODUCTOS
BEFORE INSERT ON PRODUCTO
FOR EACH ROW
DECLARE
    --DECLARACION VARIABLES
BEGIN
    :NEW.NOMBRE := UPPER (:NEW.NOMBRE) ;
END;
```

El trigger se encarga de transformar el nombre del producto a mayúsculas antes de realizar un insert sobre la tabla producto.

Como el valor que está ingresando en el INSERT es nuevo, utilizamos el prefijo :NEW para manipular el valor de la columna. En caso de que el valor de una columna ya exista en la tabla, utilizamos el prefijo :OLD.

Trigger (Ejemplo 2)



Trigger (Ejemplo 2)

Consideremos un trigger que inserte datos sobre la tabla cliente luego de ingresar un usuario.



Trigger (Ejemplo 2)

Consideremos un trigger que inserte datos sobre la tabla cliente luego de ingresar un usuario.

Debemos indicarle al trigger que realice una inserción sobre la tabla “cliente” luego de que se ingresen los datos en la tabla “usuario”.

cliente	
rut	number
tipo_usuario	varchar2(10)
estado	varchar2(10)

usuario	
rut	number
contrasena	varchar2(30)
tipo_usuario	varchar2(10)
nombre	varchar2(30)
apellidos	varchar2(50)
correo	varchar2(50)
edad	number

Trigger (Ejemplo 2)

Consideremos un trigger que inserte datos sobre la tabla cliente luego de ingresar un usuario.

cliente	
rut	number
tipo_usuario	varchar2(10)
estado	varchar2(10)

usuario	
rut	number
contrasena	varchar2(30)
tipo_usuario	varchar2(10)
nombre	varchar2(30)
apellidos	varchar2(50)
correo	varchar2(50)
edad	number

Debemos indicarle al trigger que realice una inserción sobre la tabla “cliente” luego de que se ingresen los datos en la tabla “usuario”.

```
CREATE OR REPLACE TRIGGER INSERT_USUARIO
AFTER INSERT ON USUARIO
FOR EACH ROW
DECLARE
    --DECLARACION VARIABLES
BEGIN
    IF VERIFICAR_CLIENTE (:NEW.RUT) = FALSE THEN
        INSERT INTO CLIENTE (RUT) VALUES (:NEW.RUT);
    END IF;
END;
```

Ejercicios

- Realice un trigger para la tabla usuario para que deje en mayúsculas el nombre, apellidos y correo. Además, que genere una contraseña automática considerando la primera letra del nombre, el primer apellido y los 3 últimos dígitos del rut antes del guión.
 - Ej: Felipe Tapia – Rut: 185756203
 - Contraseña: FTAPIA620

Ejercicios

- Realice un trigger que permita actualizar el total de la boleta cada vez que se ingresa un producto en la tabla detalle.
- Realice un trigger que permita disminuir el stock de la tabla producto cada vez que se ingresa un producto en la tabla detalle.
- Realice un trigger que permita ajustar el total de la tabla detalle de acuerdo a la cantidad del producto que se inserta.

