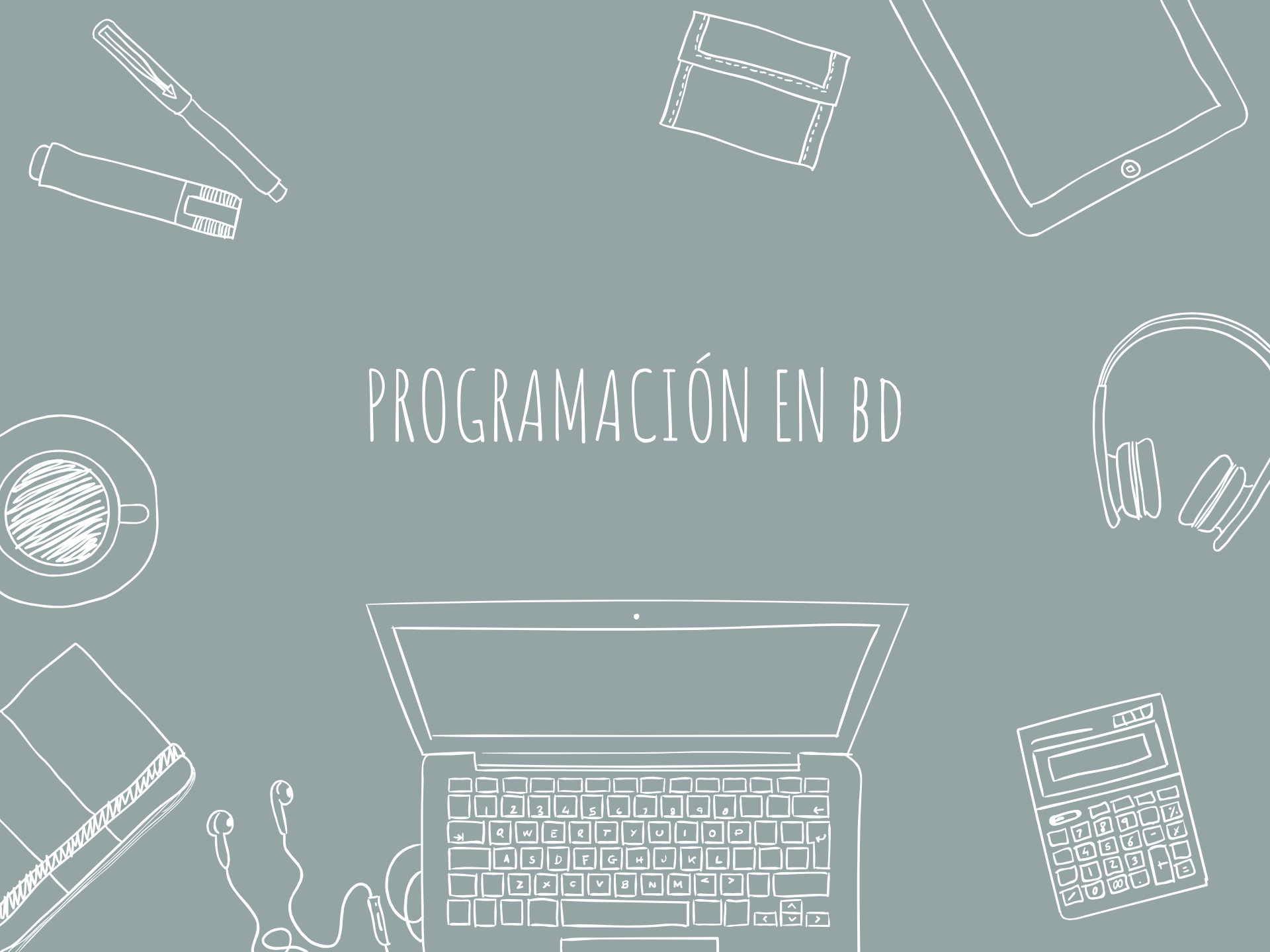


PROGRAMACIÓN EN BD



PROCEDIMIENTOS

✖ Grupo de instrucciones PL/SQL o T/SQL que se almacenan en el diccionario de la BD y que son ejecutados o llamados desde un programa remoto, otro SP o desde una línea de comandos.

```
CREATE PROCEDURE HumanResources.uspFindEmployee
    @BusinessEntityID [int]
AS
BEGIN
    SET NOCOUNT ON;
    SELECT BusinessEntityID,
           NationalIDNumber,
           LoginID,
           JobTitle,
           HireDate
    FROM HumanResources.Employee
    WHERE BusinessEntityID = @BusinessEntityID
END
```

Store Proc Name

Parameter

Select Statement to return
one employee row

Parameter used as criteria

Result Set Returned

Inputs

Execution

Output

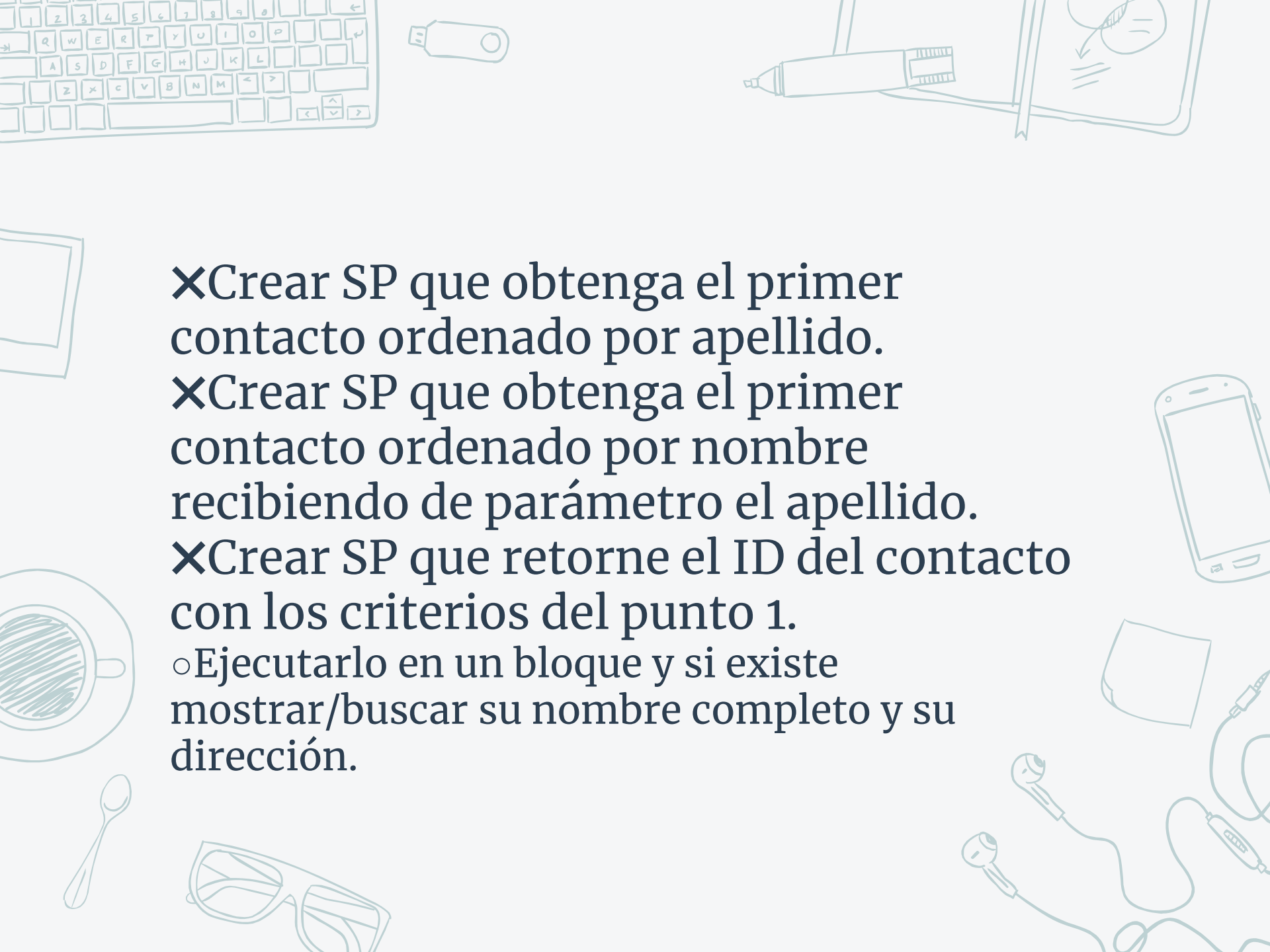



VENTAJAS

- ×Tráfico de red eficiente
- ×Mayor seguridad
- ×Reutilización de código
- ×Mantenimiento fácil
- ×Encapsulamiento de la lógica del negocio

OPERACIONES

- ×Crear
- ×Modificar
- ×Eliminar
- ×Ejecutar
- ×Parametrizar
- ×Declaración de variables
- ×Manejo de errores

- 
- ✕ Crear SP que obtenga el primer contacto ordenado por apellido.
 - ✕ Crear SP que obtenga el primer contacto ordenado por nombre recibiendo de parámetro el apellido.
 - ✕ Crear SP que retorne el ID del contacto con los criterios del punto 1.
 - Ejecutarlo en un bloque y si existe mostrar/buscar su nombre completo y su dirección.



✕ Crear un SP que duplique el precio de lista de los productos, hasta que el promedio de los mismos sea mayor a un parametro ingresado.

- Muestre cuantas veces se realizó la actualización para cada product.
- Si el color del product es “Red”, triplique el valor.



FUNCIONES

✗ Grupo de instrucciones PL/SQL o T/SQL que se almacenan en el diccionario de la BD y que son ejecutados o llamados desde un programa remoto, otra función/SP o desde una línea de comandos, y que devuelve un valor.

✗ No se puede actualizar data

DISPARADORES (TRIGGERS)

✗ Es un objeto compilado (sp) en la bd que se ejecuta cuando ocurre un evento en particular.

Momento

Tabla: BEFORE, AFTER

Vista: INSTEAD OF

Evento

INSERT, UPDATE, DELETE, LOGON, ETC

Otros

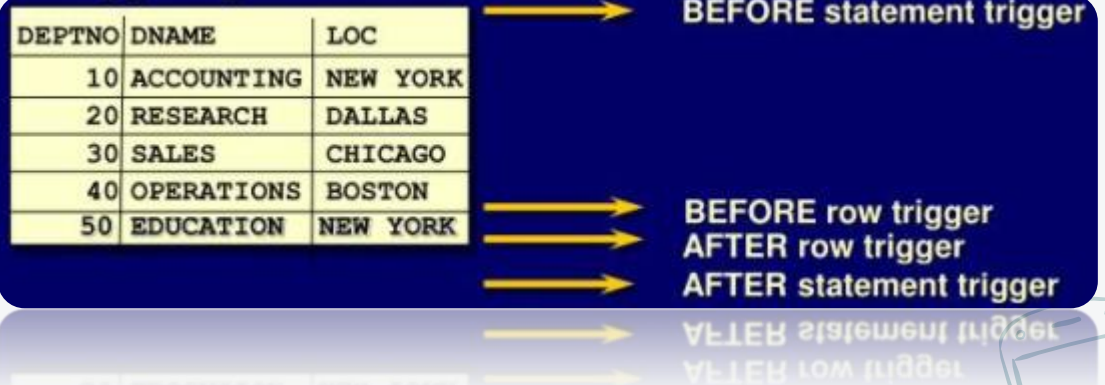
Para una fila o definición

Bajo ciertas condiciones

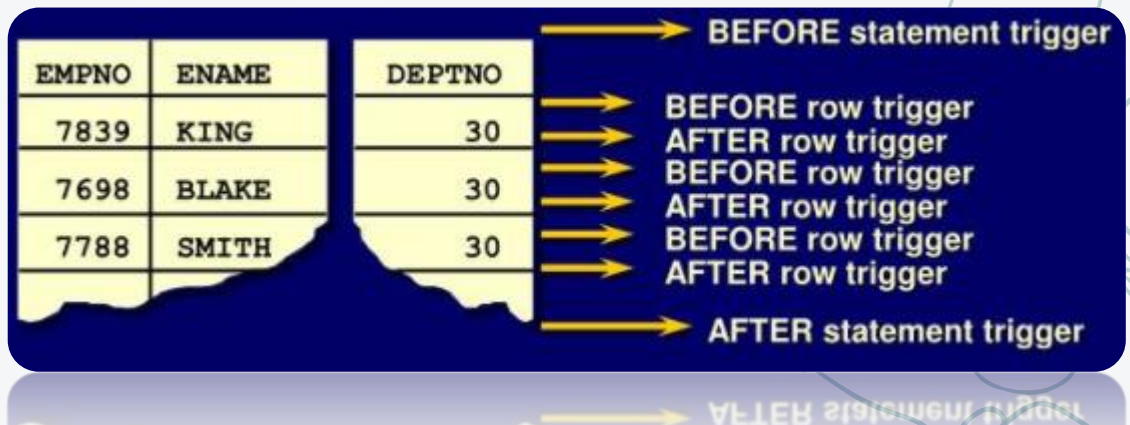
Valores : nuevo y anterior

SECUENCIA DISPARO

```
SQL> INSERT INTO dept (deptno, dname, loc)
2 VALUES (50, 'EDUCATION', 'NEW YORK');
```



```
SQL> UPDATE emp
2 SET sal = sal * 1.1
3 WHERE deptno = 30;
```





USOS

- ✕ Auditorías en DB
- ✕ Implementar reglas del negocio
- ✕ Cálculos derivados
- ✕ Mantener la integridad referencial



SINTAXIS

```
CREATE [ OR ALTER ] TRIGGER [ schema_name . ]trigger_name
ON { table }
[ WITH <dml_trigger_option> [ ,...n ] ]
{ FOR | AFTER }
{ [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ] }
AS { sql_statement [ ; ] [ ,...n ] }
```

```
CREATE [OR REPLACE] TRIGGER trigger_name
{BEFORE | AFTER } triggering_event ON table_name
[FOR EACH ROW]
[FOLLOWS | PRECEDES another_trigger]
[ENABLE / DISABLE ]
[WHEN condition]
DECLARE
    declaration statements
BEGIN
    executable statements
EXCEPTION
    exception_handling statements
END;
```



EJEMPLO

- ✗Trigger que notifique cuando exista algún cambio (insert o update) en la tabla clientes.
- ✗Trigger que notifique con correo cuando exista cambio (insert, update, delete) en la tabla clientes.



EJERCICIO

✕ Crear una función que reciba como parámetro un rango de fechas (fecha y hora) y regrese como resultado el número total de horas laborales.

✕ Días laborales: lunes-jueves

✕ Horas laborales: 8:00 a 17:00