

MIERDAS PLSQL

-SQL

CAMPOS ESPECIFICOS

SELECT campo1, campo2 FROM tabla;

TODOS:

SELECT * FROM tabla;

CONTAR FILAS:

SELECT COUNT(*) FROM tabla; -- Cuenta todas las filas de la tabla

-JOINS:

SELECT t1.campo1, t2.campo2

FROM tabla1 t1

(loquesea) JOIN tabla2 t2 ON t1.id = t2.id;

Tipos de JOIN:

- INNER JOIN → Devuelve solo las filas que tienen coincidencia en ambas tablas.
- LEFT JOIN → Devuelve todas las filas de la tabla de la izquierda y solo las que coinciden con la derecha.
- RIGHT JOIN → Devuelve todas las filas de la tabla de la derecha y solo las que coinciden con la izquierda.
- FULL JOIN → Devuelve todas las filas de ambas tablas, tira null donde no coincidan



FUNCIONES DE "MATES":

SUM(campo), AVG(campo), MAX(campo), MIN(campo)

WHERE

SELECT * FROM tabla WHERE id =1;

PLSQL

–ESTRUCTURA FUNCIONES:

```
CREATE OR REPLACE FUNCTION nombreFuncion(parametro1 NUMBER, parametro2
NUMBER)
RETURN NUMBER IS
BEGIN
    -- Código
    RETURN parametro1 + parametro2;
END;
```

EJEMPLO:

mostrar cuantos trabajadores hay en un departamento:

```
CREATE OR REPLACE FUNCTION nom_empleats(codi IN NUMBER) RETURN NUMBER
IS
    nombre_empleats NUMBER; -- variable donde guardar el resultado del
select
BEGIN
    SELECT COUNT(*) INTO nombre_empleats FROM employees WHERE
department_id = codi;
    RETURN nombre_empleats;
END;
```

MANERA DE EJECUTARLO:

```
DECLARE
    nombre_empleats NUMBER; →funcion para almacenar
BEGIN
    nombre_empleats := nom_empleats(90); → llamada a la funcion
    DBMS_OUTPUT.PUT_LINE("Treballadors per departament" ||
nombre_empleats);--> print
END;
```

PROCEDURE

sin cursor:

```
CREATE OR REPLACE PROCEDURE ActualizarSalario(  
    emp_id IN NUMBER,          -- ID del empleado  
    nuevo_salario IN NUMBER    -- Nuevo salario  
) IS  
BEGIN  
    -- Actualizar salario al empleado  
    UPDATE employees  
    SET salary = nuevo_salario  
    WHERE employee_id = emp_id;  
  
    -- Confirmamos la actualización  
    DBMS_OUTPUT.PUT_LINE('Salario actualizado para el empleado ID ' ||  
emp_id || ' a ' || nuevo_salario);  
EXCEPTION  
    WHEN NO_DATA_FOUND THEN  
        DBMS_OUTPUT.PUT_LINE('No se encontró el empleado con ID ' ||  
emp_id);  
    WHEN OTHERS THEN  
        DBMS_OUTPUT.PUT_LINE('Error al actualizar el salario: ' ||  
SQLERRM);  
END;
```

BLOQUE DE EXCEPCIONES(no suele cambiar mucho)

```
EXCEPTION  
  
    WHEN NO_DATA_FOUND THEN →sin datos  
        RETURN NULL; →aquí puedes poner lo q pida la marina  
  
    WHEN TOO_MANY_ROWS THEN → si devuelve muchas filas.  
        RETURN NULL;  
  
    WHEN OTHERS THEN →otros errores  
        RETURN NULL;  
END;
```

CURSORS EN UN PROCEDURE:

```
CREATE OR REPLACE PROCEDURE TreballadorsDepartaments(  
    dep_id IN NUMBER -> parametro(id de departamento)  
) IS  
    CURSOR emp_cursor IS -> Cursor(almacena el resultado de la query)  
        SELECT first_name, last_name, salary  
        FROM employees  
        WHERE department_id = dep_id;  
  
    --valores pa guardar lo que dé el cursor.  
    v_first_name employees.first_name%TYPE;  
    v_last_name employees.last_name%TYPE;  
    v_salary employees.salary%TYPE;  
BEGIN  
    FOR emp_rec IN emp_cursor LOOP -recorremos el cursor  
        print de cada trabajador con su sueldo.  
        DBMS_OUTPUT.PUT_LINE(emp_rec.first_name || ' ' ||  
-emp_rec.last_name || ' amb salari ' || emp_rec.salary);  
    END LOOP;  
END;
```

PA LLAMARLO:

```
BEGIN  
    TreballadorsDepartaments(90); → llamada al procedure  
END;
```

PAQUETES:

primero hay que crear la cabecera del paquete y definimos lo que queramos(funciones,procedures...)

```
CREATE OR REPLACE PACKAGE paquet_practicar AS
    FUNCTION nom_empleats(codi IN NUMBER) RETURN NUMBER;
    PROCEDURE MostrarEmpleats(p_dept_id IN NUMBER);
END;
```

y en el cuerpo de este hay que definir las funciones/procedures lo que sea(copia pega de las funciones de arriba si la marina no pide hacer nuevas, si no hay que crearlas aqui en el body):

```
CREATE OR REPLACE PACKAGE BODY paquet_practicar AS

    FUNCTION nom_empleats(codi IN NUMBER) RETURN NUMBER IS
        nombre_empleats NUMBER;
    BEGIN
        SELECT COUNT(*) INTO nombre_empleats
        FROM employees
        WHERE department_id = codi;

        RETURN nombre_empleats;
    END nom_empleats;

    PROCEDURE MostrarEmpleats(p_dept_id IN NUMBER) IS
        CURSOR cur IS
            SELECT first_name, last_name
            FROM employees
            WHERE department_id = p_dept_id;
    BEGIN
        DBMS_OUTPUT.PUT_LINE('PAQUET: Department ID: ' || p_dept_id);

        FOR TREBALLADOR IN cur LOOP
            DBMS_OUTPUT.PUT_LINE('PAQUET:Empleat: ' ||
TREBALLADOR.first_name || ' ' || TREBALLADOR.last_name);
        END LOOP;
    END MostrarEmpleats;

END paquet_practicar;
```

PARA LLAMAR A LAS COSAS DEL PAQUETE:

-- Mostrar número d'empleats

DECLARE

total NUMBER;

BEGIN

total := paquet_practicar.nom_empleats(90);

DBMS_OUTPUT.PUT_LINE('PAQUET: Treballadors per departament: ' ||
total);

END;

/

-- Mostrar llista d'empleats

BEGIN

paquet_practicar.MostrarEmpleats(90);

END;