# Teoría

Defina el concepto de un SGBD (sistema gestor de base de datos).

**Un SGBD es un conjunto de programas que permiten administrar y gestionar la información de una Base de Datos. Proporcionan a los usuarios de la BBDD facilidades para realizar las siguientes tareas:**

- **Definición de los datos en los distintos niveles de abstracción.**

- **Manipulación de los datos, ed., inserción, actualización, borrado y consulta.**

- **Mantenimiento de la integridad de la BBDD. Con la integridad se hace referencia a los datos en sí, a sus valores y a sus relaciones: los datos que se almacenan en la base han de satisfacer las restricciones definidas en el esquema de ésta.**

- **Control de la privacidad y seguridad de los datos en la BBDD, posibilitando el acceso a los mismos sólo a los usuarios autorizados.**

Explicite los lenguajes que conoce para la gestión de un SGBD.

**El SGBD ha de proporcionar lenguajes para definir y manipular los datos de la BBDD. Podrán utilizar estos lenguajes tanto los usuarios como los administradores. Son los siguientes:**

- **Lenguaje de Definición de Datos: Se utiliza para definir el esquema conceptual y los distintos subesquemas de la BBDD.**

- **Lenguaje de Manipulación de Datos: Mediante este lenguaje podemos manipular los datos de la BBDD, ed., podemos insertar datos, modificar los ya existentes, eliminar y recuperar los datos almacenados…**

- **Lenguaje de Control de Datos (funciones de administrador)**

PROCEDIMIENTOS

CREATE OR REPLACE PROCEDURE nombreproc()

AS

//declaración variables

BEGIN

//procesos

END nombreproc;

BLOQUE ANÓNIMO

DECLARE

BEGIN

END;

PARA MOSTRAR COSAS POR PANTALLA

DBMS\_OUTPUT.PUT\_LINE(‘el … es:’||variable);

PARA PEDIR COSAS EN EJECUCIÓN

en los bloques anónimos no podemos pasar nada al procedimiento por eso introduciremos las cosas mientras se ejecuta.

ejemplo:

NUMDE=’&NUMDE’

ESTRUCTURAS DE CONTROL:

**1.- ALTERNATIVA SIMPLE**

IF<CONDICIÓN>THEN

INSTRUCCIONES;

…

END IF;

**2.- ALTERNATIVA DOBLE:**

IF<CONDICIÓN>THEN

INSTRUCCIONES;

…

ELSE

INSTRUCCIONES;

…

END IF;

**3.- ALTERNATIVA MÚLTIPLE:**

IF<CONDICIÓN>THEN

INSTRUCCIONES;

…

ELSIF<CONDICIÓN>THEN

INSTRUCCIONES;

…

ELSIF<CONDICIÓN>THEN

INSTRUCCIONES;

…

END IF;

**4.- BUCLE**

LOOP

INSTRUCCIONES;

IF<CONDICIÓN>THEN

EXIT;

ENDIF;

INSTRUCCIONES;

…;

END LOOP;

**5.- MIENTRAS:**

WHILE<CONDICIÓN>LOOP

INSTRUCCIONES;

…;

END LOOP;

**6.- PARA:**

FOR<VARIABLE>IN<VALORINICIO>..<VALORFINAL>LOOP

INSTRUCCIONES;

…;

END LOOP;

CURSORES

A) Declaración del cursor:

CURSOR<NOMBRECURSOR> IS SELECT <SENTENCIA SELECT>

B) Apertura del cursor:

OPEN<NOMBRECURSOR>

C) Recogida de información:

FETCH<NOMBRECURSOR> INTO {<VARIABLE>, <VARIABLE>,…}

D) Cierre del cursor:

CLOSE<NOMBRECURSOR>

*Ejemplo: (intentad realizarlo sin cursor….)*

**DECLARE**

CURSOR C1 IS

SELECT NOMEM, NUMDE FROM TEMPLE;

VNOMEM TEMPLE.NOMEM%TYPE;

VNUMDE TEMPLE.NUMDE%TYPE;

**BEGIN**

OPEN C1;

LOOP

FETCH C1 INTO VNOMEM, VNUMDE;

EXIT WHEN C1%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(VNOMEM||VNUMDE);

END LOOP;

CLOSE C1;

**END;**

//ATRIBUTOS DEL CURSOR:

%FOUND: devuelve verdadero si el último FETCH ha recuperado valor;

%NOTFOUND: lo contrario que el anterior;

%ROWCONT: devuelve el número de filas recuperado hasta el momento;

%ISOPEN: devuelve verdadero si está abierto;

EL CURSOR FOR...LOOP

PL/SQL proporciona esta estructura que simplifica las tareas del cursor salvo la declaración del mismo. Formato y uso:

1) Se declara la información el cursor.

2) Se procesa el cursor según el siguiente formato:

3)

FOR <NOMBREVAR> IN <NOMBRECURSOR> LOOP

…

END LOOP;

Ejemplo:

DECLARE

CURSOR MI\_CURSOR IS

SELECT NOMEM, EXTEL FROM TEMPLE

WHERE NUMDE = 112;

BEGIN

FOR VREG IN MI\_CURSOR LOOP

DBMS\_OUTPUT.PUT\_LINE(VREG.NOMEM||VREG.EXTEL);

END LOOP;

END;

# Ejemplos

**LOS PROCEDIMIENTOS LOS HAREMOS EN LA INTRFAZ DE ORACLE LOS BLOQUES ANÓNIMOS SERÁN EN LA LINE DE CMANDOS DE ORACLE**

**PROCEDIMINETO QUE MUESTRA EL NOMBRE DE LOS 5 EMPLEADOS CON SALARIO MAS ALTO**

VAMOS A UTILIZAR UN CURSOR PORQUE VAMOS A DEVOLVER MAS DE UN DATO

CREATE OR REPLACE PROCEDURE AUX

AS

CURSOR MICURSOR IS SELECT \* FROM TEMPLE ORDER BY SALAR DESC;

TUPLA TEMPLE%ROWTYPE PORQUE COGEMOS TODA LA FILA POR ESO DE TIPO TEMPLE

UTILIZAMOS LA FORMA LARGA DE RECORRER CURSORES PORQUE VAMOS A PONER CONDICIONES DENTRO Y EN LA FORMA ABREBIADA NO TE DEJA

BEGIN

OPEN MICURSOR;

FETCH MICURSOR INTO TUPLA;

FOR I IN 1..5 LOOP

DBMS\_OUTPUT.PUT\_LINE('NOMBRE: ' || TUPLA.NOMEM || 'SALARIO:' || 'TUPLA.SALAR');

FETCH MICURSOR INTO TUPLA;

END LOOP;

CLOSE MICURSOR;

END SAL;

**Otra forma hecha por Marcos que es más facil**

CREATE OR REPLACE PROCEDURE ver\_nombres

AS

CURSOR MICURSOR IS

SELECT NUMEM, NOMEM,EXTEL

FROM TEMPLE;

NUM NUMBER :=1;

BEGIN

FOR VREG IN MICURSOR LOOP

IF (NUM < 4) THEN

DBMS\_OUTPUT.PUT\_LINE('NUMEM: ' || VREG.NUMEM || ' , NOMBRE: ' || VREG.NOMEM || ',EXTENSION TELEFONICA: ' || VREG.EXTEL);

NUM:=NUM+1;

END IF;

END LOOP;

END ver\_nombres;

**SE PRECISA QUE POR ACTUALIZACIÓN DE LA BBDD SI HAY MAS DE 30 EMPLEADOS EN TEMPLE SE GENERE UNA COPIA DE LA MISMA CON NOMBRETEMPLEAUX DONDE**

**A) SE VUELQUEN TODOS LOS EMPLEADOS CON COMIS.**

**B) SE ELIMINE LA TABLA TEMPLE ORIGINAL**

CREATE OR REPLACE PROCEDURE NUEVA\_TEMPLE

AS

NUM\_EMPL NUMBER;

BEGIN

SELECT COUNT(NUMEM) INTO NUM\_EMPL

FROM TEMPLE;

IF(NUM\_EMPL > 30) THEN

INSERT INTO TEMPLE\_AUX

SELECT \*

FROM TEMPLE;

WHERE COMIS IS NOT NULL;

ELSE

DMBS\_OUTPUT.PUT\_LINE(‘NO ES POSIBLE ACTULIZAR…’);

ENDIF;

END NUEVA\_TEMPLE

**BLOQUEA ANÓNIMO QUE NOS DA LOS EMPLEADOS QUE TRABAJAN EN EL MISMO DEPARTAMENTO QUE PABLO ARCONES.**

DECLARE

CURSOR C1 IS SELECT \* FROM TEMPLE WHERE NUMDE = (SELECT NUMDE FROM TEMPLE WHERE NOMEM = ‘”ARCONES,PABLO”’);

BEGIN

FOR VAR INTO C1 LOOP

DBMS\_OUTPUT\_LINE('NOMBRE: ' || VAR.NOMEM || 'SALARIO:' || VAR.SALAR');

END LOOP;

END;

/

**CREAR UN BLOQUE ANÓNIMO TAL QUE SI EL SALARIO DE PILAR MOLINA ENTRE NUMHI >100 SE ACTUALIZE EL SALARIO DE TODOS LOS EMPLEADOS DE SU MISMO DEP EN 50$+.**

DECLARE

OP NUMBER;

P\_NUMDE TEMPLE.NUMDE%TYPE;

BEGIN

SELECT SALAR/NUMHI, NUMDE INTO OP, P\_NUMDE

FROM TEMPLE

WHERE NOMEM= ‘”MOLINA,PILAR”’;

IF (OP>100) THEN

UPDATE TEMPLE

SET SALAR =SALAR+50

WHERE NUMDE=P\_NUMDE;

END IF;

END;

**ej: PROCEDIMINETO EN EL QUE SE VISUALIZA SALARIO DE UN EMPLEADO CUYO NUMERO DE EMPLEADO SE FACILITA POR PARAMETRO**

SET SERVEROUTOUT ON; PARA TENER COMUNICACIÓN CON EL USUARIO

CREATE OR REPLACE PROCEDURE ver\_salario(V\_NUMEM NUMBER)

AS TAMBIEN VALDRIA IS

V\_SALAR TEMPLE.SALAR%TYPE //DECLARACION VARIABLES

BEGIN

SELECT SALAR INTO V\_SALAR

FROM TEMPLE

WHERE NUMEM = V\_NUMEM; //; PORQUE TERMINA LA QUERI

DBMS\_OUTPUT.PUT\_LINE('EL SALARIO ES '||V\_SALAR);

END ver\_salario;

NOTA:SI ESTAMOS EN LA CONSULTA DE ORACLE HAY QUE PONER EL /

EXECUTE ver\_salario(111); nos ejecuta el procedimiento y si esta el empleado 111 nos devuelve su salario.

**MODIFICACION METEMOS EL NOMEM EN VEZ DE NUMEM**

CREATE OR REPLACE PROCEDURE ver\_salario(V\_NUMEM STRING)

AS

V\_SALAR TEMPLE.SALAR%TYPE;

BEGIN

SELECT SALAR INTO V\_SALAR

FROM TEMPLE

WHERE NOMEM = V\_NUMEM;

DBMS\_OUTPUT.PUT\_LINE('EL SALARIO ES '||V\_SALAR);

END ver\_salario;

**PROCEDIMINETO QUE MODIFICA EL SALARIO DE UN EMPLEADO PASANDOLE TANTO EL NÚMERO DEL EMPLEADO COMO EL NUEVO SALARIO**

SET SERVEROUTPUT ON; PARA TENER COMUNICACIÓN CON EL USUARIO

CREATE OR REPLACE PROCEDURE mod\_salario(V\_NUMEM NUMBER,V\_NSALAR NUMBER)

AS

BEGIN

UPDATE TEMPLE

SET SALAR= V\_NSALAR

WHERE NUMEM = V\_NUMEM;

EXCEPTION WHEN NO\_DATA \_FOUND

DBMS\_OUTPUT.PUT\_LINE (‘’);

END mod\_salario;

/ SI ESTUVIERAMOS EN CONSOLA

EXECUTE mod\_salario(111, 4000)

**LOS BLOQUES ANÓNIMOS SOLO SE PUEDEN EJECUTAR UNA VEZ PORQUE NO TIENE NOMBRE Y NO LOS PUEDES VOLVER A LLAMAR, LOS PRECEDIMIENTOS COMO TIENE NOMBRE SI SE PUEDEN LLAMAR VARIAS VECES**

**GENERAR UN BLOQUE ANÓNIMO QUE DEVUELVE EL SALARIO DE UN EMPLEADO CUYO NÚMERO SE LO FACILITAS EN EJECUCIÓN**

DECLARE SE PONE DECLARE PQ ES UN BLOQUE ANONIMO

V\_SALARIO TEMPLE.SALAR%TYPE;

BEGIN

SELECT SALAR INTO V\_SALARIO

FROM TEMPLE

WHERE NUMEM = ‘&DAME\_NUMEM’;

DBMS\_OUTPUT.PUT\_LINE(‘El salario es: ‘ || V\_SALARIO);

END;

/

**HAY DOS TIPOS DE CURSORES CURSOR FOR LOOP QUE ES EL QUE VAMOS A UTILIZAR POR DEFECTO, CUANDO TE VA A DEVOLVER VARIOS VALORES**

**SIEMPRE TIENE ASOCIADA UNA VARIABLE QUE NO HAY QUE DEFINIRLA**

**UN CURSOR EN BLOQUE ANÓNIMO**

DECLARE

CURSOR MICIRSOR IS

SELECT NOMEM FROM TEMPLE WHERE NUMDE=111;

BEGIN

FOR V\_reg IN MICURSOR LOOP

DBMS\_OUTPUT.PUT\_LINE(‘V\_reg.NOMEM’);

END LOOP;

END;

/

**PROCEDIMIENTO QUE RECIBE DOS NUMEROS DE EMPLEADOS Y VERIFICA SI TIENE LA MISMA EXTEL EN CASO CONTRARIO DEVOLVERÁ LAS EXTEL DE CADA EMPLEADO Y UBICACIÓN**

CREATE OR REPLACE PROCEDURE NUM(A\_NUMEM NUMBER, B\_NUMEM NUMBER)

AS

A\_EXTEL TEMPLE.EXTEL%TYPE;

B\_EXTEL TEMPLE.EXTEL%TYPE;

A\_DIRECC TCENTR.SEÑAS%TYPE;

B\_DIRECC TCENTR.SEÑAS%TYPE;

BEGIN

SELECT EXTEL, SEÑAS INTO A\_EXTEL, A\_DIRECC

FROM TEMPLE, TDEPTO, TCENTR

WHERE TEMPLE.NUMDE=TDEPTO.NUMDE AND TDEPTO.NUMCE=TCENTR.NUMCE AND NUMEM =A\_NUMEM;

SELECT EXTEL, SEÑAS INTO B\_EXTEL, B\_DIRECC

FROM TEMPLE, TDEPTO, TCENTR

WHERE TEMPLE.NUMDE=TDEPTO.NUMDE AND TDEPTO.NUMCE=TCENTR.NUMCE AND NUMEM =B\_NUMEM;

IF A\_EXTEL = B\_EXTEL THEN

DBMS\_OUTPUT.PUT\_LINE('VERIFICAMOS');

ELSE

DBMS\_OUTPUT.PUT\_LINE(A\_EXTEL||A\_DIRECC);

DBMS\_OUTPUT.PUT\_LINE(B\_EXTEL||B\_DIRECC);

END IF;

END NUM;

**PROCEDIMIENTO QUE NOS MUESTRA EL NUMERO DE EMPLEADOS QUE TRABAJAN EN CADA UNA DE LAS SEDES SI HAY MÁS DE DOS EMPLEADOS QUE TRABAJAN EN LA SEDE DE ATOCHA.**

**Muestra los empleados Y SU SALARIO del departamento que introduzcamos:**

CREATE OR REPLACE PROCEDURE VER\_POR\_DEPARTAMENTOS(NUM\_DEP NUMBER)

AS

CURSOR C1 IS SELECT NOMEM,SALAR FROM TEMPLE WHERE NUM\_DEP=NUMDE;

BEGIN

FOR VAR IN C1 LOOP

DBMS\_OUTPUT.PUT\_LINE('NOMBRE: ' || VAR.NOMEM || ' ' || VAR.SALAR);

END LOOP;

END VER\_POR\_DEPARTAMENTOS;

SELECT \* FROM TEMPLE

**CREAR BLOQUE ANONIMO QUE TE MUESTRE TODOS LOS EMPLEADOS DEL MISMO DEPARTAMENTO QUE FLOR DOROTEA**

**FORMA 1: EL CURSOR YA SELECCIONA**

DECLARE

CURSOR C1 IS SELECT NOMEM FROM TEMPLE WHERE NUMDE = (SELECT NUMDE FROM TEMPLE WHERE NOMEM = '"FLOR,DOROTEA"');

BEGIN

FOR VAR IN C1 LOOP

DBMS\_OUTPUT.PUT\_LINE('LOS NOMBRES SON: ' || VAR.NOMEM);

END LOOP;

END;

**FORMA 2: SELECCIONAMOS EN EL BEGIN**

DECLARE

CURSOR C1 IS SELECT NOMEM, NUMDE FROM TEMPLE;

DEPT TEMPLE.NUMDE%TYPE;

BEGIN

SELECT NUMDE INTO DEPT FROM TEMPLE WHERE NOMEM LIKE '%DORO%';

FOR VAR IN C1 LOOP

IF(VAR.NUMDE = DEPT) THEN

DBMS\_OUTPUT.PUT\_LINE('NOMBRE: ' || VAR.NOMEM);

END IF;

END LOOP;

END;

**Generar un procedimiento PL/SQL tal que si se le facilita un número de empleado nos devuelva el nombre, salario, número de hijos y extensión telefónica de este.**

HOLI RICHI

CREATE OR REPLACE PROCEDURE VEREMP(EMPL NUMBER)

AS

NOMEM1 TEMPLE.NOMEM%TYPE;

BEGIN

SELECT NOMEM INTO NOMEM1 FROM TEMPLE WHERE EMPL = NUMEM;

DBMS\_OUTPUT.PUT\_LINE(NOMEM1);

END VEREMP;

**Generar un procedimiento que visualice el nombre de los departamentos ubicados en un centro cuyo número de referencia se facilita por parámetro.**

CREATE OR REPLACE PROCEDURE VERDEP(CENTRO NUMBER)

AS

CURSOR C1 IS SELECT NOMDE FROM TDEPTO WHERE CENTRO = NUMCE;

BEGIN

FOR VAR IN C1 LOOP

DBMS\_OUTPUT.PUT\_LINE(VAR.NOMDE);

END LOOP;

END VERDEP;