**CREATE** **OR** **REPLACE** PACKAGE paquete\_empleo

**AS**

**PROCEDURE** agrega\_empleo

(id\_chamba **IN** jobs.job\_id%**TYPE**,

tit\_chamba **IN** jobs.job\_title%**TYPE**);

**PROCEDURE** borra\_empleo

(id\_chamba **IN** jobs.job\_id%**TYPE**);

**PROCEDURE** actualiza\_empleo

(id\_chamba **IN** jobs.job\_id%**TYPE**,

salario\_minimo **IN** jobs.min\_salary%**TYPE**,

salario\_maximo **IN** jobs.max\_salary%**TYPE**);

**FUNCTION** obtiene\_anios\_servicioq

(id\_empleado **IN** employees.employee\_id%**TYPE**)

**RETURN** NUMBER;

**END** paquete\_empleo;

/

**CREATE** **OR** **REPLACE** PACKAGE BODY paquete\_empleo

**AS**

**PROCEDURE** agrega\_empleo

(id\_chamba **IN** jobs.job\_id%**TYPE**,

tit\_chamba **IN** jobs.job\_title%**TYPE**)

**IS**

v\_id jobs.job\_id%**TYPE**:=id\_chamba;

v\_ti jobs.job\_title%**TYPE**:=tit\_chamba;

**BEGIN**

**INSERT** **INTO** jobs(job\_id, job\_title) **VALUES**(**UPPER**(v\_id),INITCAP(v\_ti));

**EXCEPTION**

**WHEN** OTHERS

**THEN**

DBMS\_OUTPUT.PUT\_LINE('No se puede hacer la inserción en la tabla');

**END** agrega\_empleo;

**PROCEDURE** borra\_empleo

(id\_chamba **IN** jobs.job\_id%**TYPE**)

**IS**

v\_id jobs.job\_id%**TYPE**:=**upper**(id\_chamba);

**BEGIN**

**DELETE** **FROM** jobs **where** job\_id=v\_id;

**EXCEPTION**

**WHEN** OTHERS

**THEN**

DBMS\_OUTPUT.PUT\_LINE('No se puede hacer el borrado de la tabla');

**END** borra\_empleo;

**PROCEDURE** actualiza\_empleo

(id\_chamba **IN** jobs.job\_id%**TYPE**,

salario\_minimo **IN** jobs.min\_salary%**TYPE**,

salario\_maximo **IN** jobs.max\_salary%**TYPE**)

**IS**

v\_id jobs.job\_id%**TYPE**:=id\_chamba;

v\_salmin jobs.min\_salary%**TYPE**:=salario\_minimo;

v\_salmax jobs.max\_salary%**TYPE**:=salario\_maximo;

**BEGIN**

**UPDATE** jobs **set** min\_salary=v\_salmin, max\_salary=v\_salmax **WHERE** job\_id=**upper**(v\_id);

**EXCEPTION**

**WHEN** OTHERS

**THEN**

DBMS\_OUTPUT.PUT\_LINE('No se puede hacer la actualización de la tabla');

**END** actualiza\_empleo;

**FUNCTION** obtiene\_anios\_servicio

(id\_empleado **IN** employees.employee\_id%**TYPE**)

**RETURN** NUMBER

**IS**

v\_idemp employees.employee\_id%**TYPE**:=id\_empleado;

v\_fecha employees.hire\_date%**TYPE**;

v\_anti NUMBER;

**BEGIN**

**SELECT** hire\_date **INTO** v\_fecha

**FROM** employees

**WHERE** employee\_id=v\_idemp;

v\_anti:=TRUNC(months\_between(sysdate,v\_fecha)/12);

**RETURN** v\_anti;

**END** obtiene\_anios\_servicio;

**END** paquete\_empleo;

/

**SET** SERVEROUTPUT **ON**

**DECLARE**

texto\_id jobs.job\_id%**TYPE**:='&v\_id';

texto\_tit jobs.job\_title%**TYPE**:='&v\_tit';

**BEGIN**

paquete\_empleo.agrega\_empleo(texto\_id, texto\_tit);

**END**;

/

**SET** SERVEROUTPUT **ON**

**DECLARE**

texto\_id jobs.job\_id%**TYPE**:='&v\_id';

**BEGIN**

paquete\_empleo.borra\_empleo(texto\_id);

**END**;

/

**SET** SERVEROUTPUT **ON**

**DECLARE**

texto\_id jobs.job\_id%**TYPE**:='&v\_id';

v\_salmin jobs.min\_salary%**TYPE**:=&salario\_minimo;

v\_salmax jobs.max\_salary%**TYPE**:=&salario\_maximo;

**BEGIN**

paquete\_empleo.actualiza\_empleo(texto\_id, v\_salmin, v\_salmax);

**END**;

/

**SET** SERVEROUTPUT **ON**

**DECLARE**

**result** NUMBER;

**BEGIN**

**result**:=paquete\_empleo.obtiene\_anios\_servicio(&v\_id);

DBMS\_OUTPUT.PUT\_LINE(**result**);

**END**;

/

**CREATE** **OR** **REPLACE** **TRIGGER** new\_emp\_dept

**INSTEAD** **OF** **INSERT** **ON** emp\_details

**FOR** **EACH** **ROW**

**BEGIN**

**INSERT** **INTO** new\_emps **VALUES** (:**NEW**.employee\_id, :**NEW**.last\_name, :**NEW**.salary, :**NEW**.department\_id, :**NEW**.email, :**NEW**.job\_id, sysdate);

**DELETE** **FROM** new\_depts;

**INSERT** **INTO** new\_depts **select** d.department\_id, d.department\_name, d.location\_id,

**sum** (e.salary) tot\_dept\_sal

**from** new\_emps e, departments d

**where** e.department\_id= d.department\_id

**group** **by** d.department\_id, d.department\_name, d.location\_id;

**END**;

/

**create** **table** new\_emps **as**

**select** employee\_id, last\_name, salary, department\_id,

email, job\_id, hire\_date

**from** employees;

**create** **table** new\_depts **as**

**select** d.department\_id, d.department\_name, d.location\_id,

**sum** (e.salary) tot\_dept\_sal

**from** employees e, departments d

**where** e.department\_id= d.department\_id

**group** **by** d.department\_id, d.department\_name, d.location\_id;

**create** **view** emp\_details **as**

**select** e.employee\_id, e.last\_name, e.salary, e.department\_id,

e.email, e.job\_id, d.department\_name, d.location\_id

**from** employees e, departments d

**where** e.department\_id = d.department\_id;

**INSERT** **INTO** emp\_details **VALUES** (208,'Cabrera',15000,210, 'prueba@prueba.com', 'IT\_SYST', 'IT Support', 1700);

**CREATE** **OR** **REPLACE** **TRIGGER** act\_min\_sal

**AFTER** **UPDATE** **ON** jobs

**FOR** **EACH** **ROW**

**DECLARE**

v\_idtrab jobs.job\_id%**TYPE**:=:**NEW**.job\_id;

v\_salmin jobs.min\_salary%**TYPE**:=:**NEW**.min\_salary;

**BEGIN**

**UPDATE** employees **SET** salary=v\_salmin **where** job\_id=v\_idtrab **and** salary<v\_salmin;

**END**;

/

**CREATE** **OR** **REPLACE** **PROCEDURE** upd\_empm\_sal

(id\_trabajo jobs.job\_id%**TYPE**,

sal\_min jobs.min\_salary%**TYPE**);

**IS**

v\_idtrab jobs.job\_id%**TYPE**:=id\_trabajo;

v\_salmin jobs.min\_salary%**TYPE**:=sal\_min;

**BEGIN**

**UPDATE** jobs **SET** min\_salary=v\_salmin **WHERE** job\_id=v\_idtrab;

**END**;

/

**UPDATE** jobs **SET** min\_salary=5000 **WHERE** job\_id='IT\_PROG';

**CREATE** **TABLE** INFO\_TABLA

(

Usuario VARCHAR2(30),

Nombre\_Tabla VARCHAR2(30),

Fecha DATE,

Hora **TIMESTAMP**

);

**CREATE** **OR** **REPLACE** **TRIGGER** monitor\_tabla

**AFTER** **CREATE** **ON** **SCHEMA**

**DECLARE**

nombre\_tabla user\_objects.object\_name%**TYPE**;

**BEGIN**

IF SYS.DICTIONARY\_OBJ\_TYPE = 'TABLE' **THEN**

**select** object\_name **into** nombre\_tabla **from** user\_objects **where** object\_type='TABLE'

**AND** **TIMESTAMP**=(**select** **MAX**(**TIMESTAMP**) **from** user\_objects **where** object\_type='TABLE');

**INSERT** **INTO** INFO\_TABLA **VALUES** (**USER**,nombre\_tabla,SYSDATE,SYSDATE);

**END** IF;

**END**;

/

**create** **or** **replace** **procedure** separacadena

(v\_cadori **IN** VARCHAR2,

v\_cad1 **OUT** VARCHAR2,

v\_cad2 **OUT** VARCHAR2)

**AS**

**BEGIN**

IF **length**(v\_cadori)<60 **THEN** v\_cad1:=v\_cadori;

v\_cad2:='No hay cadena';

**ELSE**

v\_cad1:=substr(v\_cadori,1,instr(v\_cadori,' ',(60-**length**(v\_cadori)),1));

v\_cad2:=substr(v\_cadori,instr(v\_cadori,' ',(60-**length**(v\_cadori)),1)+1,**length**(v\_cadori)-instr(v\_cadori,' ',(60-**length**(v\_cadori)),1));

**END** IF;

**END** separacadena;

/

**DECLARE**

v\_nombre trabajadores.nombre%**TYPE**;

v\_sueldo trabajadores.salporhora%**TYPE**;

**CURSOR** c\_trabajador **IS**

**SELECT** nombre

**FROM** trabajadores;

**CURSOR** c\_sueldo **IS**

**SELECT** nombre, salporhora\*horastrabajadas sueldo

**FROM** trabajadores

**WHERE** nombre=v\_nombre;

**BEGIN**

**FOR** r\_trabajador **IN** c\_trabajador LOOP

v\_nombre:=r\_trabajador.nombre;

**FOR** r\_sueldo **in** c\_sueldo LOOP

v\_sueldo:=r\_sueldo.sueldo;

DBMS\_OUTPUT.PUT\_LINE('El empleado :['||v\_nombre||']');

DBMS\_OUTPUT.PUT\_LINE('tiene un sueldo sin impuestos de:['||to\_char((v\_sueldo)\*(1-0.16),'$999,999.99')||']');

DBMS\_OUTPUT.PUT\_LINE('Sueldo mas iva:['||to\_char((v\_sueldo)\*(1.16),'$999,999.99')||']');

DBMS\_OUTPUT.PUT\_LINE('Sueldo menos 10% isr, y 10% iva:['||to\_char((v\_sueldo)\*(0.9)\*(0.9),'$999,999.99')||']');

DBMS\_OUTPUT.PUT\_LINE('su sueldo neto es de:['||numero\_a\_texto(round((v\_sueldo)\*(0.9)\*(0.9)+1,0))||']');

**END** LOOP;

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

**END** LOOP;

**END**;

/

**DECLARE**

v\_stud\_id grade.student\_id%**TYPE**;

v\_sec\_id grade.section\_id%**TYPE**;

**CURSOR** c\_estu **IS**

**select** first\_name,last\_name,student\_id

**FROM** student st;

**CURSOR** c\_curso **IS**

**SELECT** **c**.description, e.student\_id, s.section\_id, s.section\_no

**FROM** enrollment e, section s, course **c**

**WHERE** e.student\_id=v\_stud\_id

**AND** s.section\_id=e.section\_id

**AND** **c**.course\_no=s.course\_no;

**CURSOR** c\_cali **IS**

**select** **g**.student\_id,**g**.section\_id, **g**.grade\_type\_code, gt.description, **avg**(**g**.numeric\_grade) **as** prom

**FROM** grade **g**, grade\_type gt

**WHERE** **g**.grade\_type\_code=gt.grade\_type\_code

**and** **g**.student\_id=v\_stud\_id

**and** **g**.section\_id=v\_sec\_id

**GROUP** **BY** **g**.student\_id,**g**.section\_id, **g**.grade\_type\_code, gt.description;

**BEGIN**

**FOR** r\_estu **IN** c\_estu LOOP

DBMS\_OUTPUT.PUT\_LINE ('Alumno: '||r\_estu.first\_name||' '||r\_estu.last\_name);

v\_stud\_id:=r\_estu.student\_id;

**FOR** r\_curso **in** c\_curso LOOP

DBMS\_OUTPUT.PUT\_LINE (chr(9)||'Calificaciones del Curso: '||r\_curso.description||' de la seccion número: '||r\_curso.section\_no);

v\_sec\_id:=r\_curso.section\_id;

**FOR** r\_cali **IN** c\_cali LOOP

DBMS\_OUTPUT.PUT\_LINE (chr(9)||to\_char(r\_cali.prom,'99.99')||' '||r\_cali.description);

**END** LOOP;

**END** LOOP;

**END** LOOP;

**END**;

/

**DECLARE**

**CURSOR** c\_ciudad **IS**

**SELECT** city

**FROM** zipcode

**WHERE** rownum <= 10;

**TYPE** tipo\_ciudad **IS** VARRAY(10) **OF** zipcode.city%**TYPE**;

varray\_ciudad tipo\_ciudad:=tipo\_ciudad(); *--faltaba inicializar*

v\_contador INTEGER := 0;

**BEGIN**

**FOR** reg\_ciudad **IN** c\_ciudad LOOP

v\_contador := v\_contador + 1;

varray\_ciudad.EXTEND; *--faltaba extender para poder escribir*

varray\_ciudad(v\_contador) := reg\_ciudad.city;

DBMS\_OUTPUT.PUT\_LINE('Varray\_ciudad('||v\_contador||'): '||varray\_ciudad(v\_contador));

**END** LOOP;

**END**;

/

**CREATE** **OR** **REPLACE** **FUNCTION** BOOLEAN\_TO\_CHAR(STATUS **IN** BOOLEAN)

**RETURN** VARCHAR2 **IS**

**BEGIN**

**RETURN**

**CASE** STATUS

**WHEN** **TRUE** **THEN** 'VERDADERO'

**WHEN** **FALSE** **THEN** 'FALSO'

**ELSE** 'NULO'

**END**;

**END**;

**CREATE** **OR** **REPLACE** **FUNCTION** cod\_post(codigo zipcode.zip%**TYPE**)

**RETURN** BOOLEAN

**AS**

v\_codigo zipcode.zip%**TYPE**:=**NULL**;

**BEGIN**

**select** **unique**(zip) **INTO** v\_codigo

**from** zipcode

**WHERE** zip=codigo;

**RETURN** **FALSE**;

**EXCEPTION**

**WHEN** OTHERS **THEN**

**RETURN** **TRUE**;

**END**;

/

**begin**

dbms\_output.put\_line(boolean\_to\_char(cod\_post('48104')));

**end**;

/

**DECLARE**

v\_idestudiante student.student\_id%**type** := &sv\_idestudiante;

v\_total\_cursos NUMBER;

e\_id\_invalido **EXCEPTION**;

**BEGIN**

IF v\_idestudiante < 0 **THEN**

RAISE e\_id\_invalido;

**ELSE**

**SELECT** **COUNT**(\*)

**INTO** v\_total\_cursos

**FROM** enrollment **WHERE** student\_id = v\_idestudiante;

DBMS\_OUTPUT.PUT\_LINE ('El estudiante esta registrado en '||

v\_total\_cursos||' courses');

**END** IF;

DBMS\_OUTPUT.PUT\_LINE ('Ninguna excepcion ha sido encontrada');

**EXCEPTION**

**WHEN** e\_id\_invalido **THEN** DBMS\_OUTPUT.PUT\_LINE ('Un id no puede ser negativo');

**END**;

/

**DECLARE**

e\_fechdeb **EXCEPTION**; *--declaracion externa*

v\_numact NUMBER;

**BEGIN**

**DECLARE** *---------- empieza sub-bloque*

e\_fechdeb **EXCEPTION**; *-- esta declaración (interna) prevalece*

v\_numact NUMBER;

fecha\_debida DATE := SYSDATE - 1;

fecha\_actual DATE := SYSDATE;

**BEGIN**

IF fecha\_debida < fecha\_actual **THEN**

RAISE e\_fechdeb; *-- Este no se maneja*

**END** IF;

**END**; *------------- termina sub-bloque*

**EXCEPTION**

**WHEN** e\_fechdeb **THEN**

DBMS\_OUTPUT.PUT\_LINE

('Manejo de la excepcion e\_fechdeb.');

**WHEN** OTHERS **THEN**

DBMS\_OUTPUT.PUT\_LINE

('No es posible reconocer la excepcion e\_fechdeb en este alcance.');

**END**;

/

**SET** SERVEROUTPUT **ON**

**DECLARE**

v\_nombre\_est VARCHAR2(15); *---linea 3*

**BEGIN**

v\_nombre\_est := 'JOSE ANTONIO CORIA'; *---- linea 5*

DBMS\_OUTPUT.PUT\_LINE ('Mi nombre es '||v\_nombre\_est);

**DECLARE**

v\_nombre\_din VARCHAR2(15);

**BEGIN**

v\_nombre\_din := '&sv\_nombre\_din';

DBMS\_OUTPUT.PUT\_LINE ('Tu nombre es '||v\_nombre\_din);

**EXCEPTION**

**WHEN** VALUE\_ERROR **THEN**

DBMS\_OUTPUT.PUT\_LINE ('El nombre es muy largo (interno');

**END**;

**EXCEPTION**

**WHEN** VALUE\_ERROR **THEN**

DBMS\_OUTPUT.PUT\_LINE ('El nombre es muy largo (externo)');

**END**;

/

**create** **or** **replace** **procedure** valida\_cupo

**is**

v\_estudiantes NUMBER(3) := 0;

**begin**

**SELECT** **COUNT**(\*) **INTO** v\_estudiantes

**FROM** enrollment e, section s

**WHERE** e.section\_id = s.section\_id

**AND** s.course\_no = 25

**AND** s.section\_id = 89;

If v\_estudiantes > 10 **then**

RAISE\_APPLICATION\_ERROR (-20001, 'El curso tiene mas de 10 estudiantes', **true**);

**else**

DBMS\_OUTPUT.PUT\_LINE ('El Curso 25, sección 89 tiene '||v\_estudiantes|| ' estudiantes');

**End** if;

**End**;

**SET** SERVEROUTPUT **ON**

**DECLARE**

v\_estudiantes NUMBER(3) := 0;

e\_curso\_lleno **EXCEPTION**;

PRAGMA EXCEPTION\_INIT(e\_curso\_lleno, -20001);

**BEGIN**

valida\_cupo;

**EXCEPTION**

**WHEN** e\_curso\_lleno **THEN**

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

**END**;

/

**DECLARE**

emp\_column VARCHAR2(30) := 'last\_name';

**table\_name** VARCHAR2(30) := 'employees';

temp\_var VARCHAR2(30);

**BEGIN**

temp\_var := emp\_column;

**SELECT** **COLUMN\_NAME** **INTO** temp\_var **FROM** USER\_TAB\_COLS

**WHERE** **TABLE\_NAME** = 'EMPLOYEES'

**AND** **COLUMN\_NAME** = **UPPER**(emp\_column);

temp\_var := **table\_name**;

**SELECT** OBJECT\_NAME **INTO** temp\_var **FROM** USER\_OBJECTS

**WHERE** OBJECT\_NAME = **UPPER**(**table\_name**)

**AND** OBJECT\_TYPE = 'TABLE';

**EXCEPTION**

**WHEN** NO\_DATA\_FOUND **THEN**

DBMS\_OUTPUT.PUT\_LINE('No se encontraron Datos para SELECT en ' || temp\_var);

**END**;

/

**set** serveroutput **on**

**DECLARE**

**TYPE** indice\_por\_tipo **IS** **TABLE** **OF** NUMBER

**INDEX** **BY** BINARY\_INTEGER;

indice\_por\_tabla indice\_por\_tipo;

**TYPE** tipo\_anidado **IS** **TABLE** **OF** NUMBER;

tabla\_anidada tipo\_anidado := tipo\_anidado(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

**BEGIN**

**FOR** i **IN** 1..10 LOOP

indice\_por\_tabla(i) := i;

**END** LOOP;

IF indice\_por\_tabla.**EXISTS**(3) **THEN**

DBMS\_OUTPUT.PUT\_LINE ('indice\_por\_tabla(3) = '||indice\_por\_tabla(3));

**END** IF;

tabla\_anidada.**DELETE**(10);

tabla\_anidada.**DELETE**(1,3);

indice\_por\_tabla.**DELETE**(10);

DBMS\_OUTPUT.PUT\_LINE ('tabla\_anidada.COUNT = '||tabla\_anidada.**COUNT**);

DBMS\_OUTPUT.PUT\_LINE ('indice\_por\_tabla.COUNT = '|| indice\_por\_tabla.**COUNT**);

DBMS\_OUTPUT.PUT\_LINE ('tabla\_anidada.FIRST = '||tabla\_anidada.**FIRST**);

DBMS\_OUTPUT.PUT\_LINE ('tabla\_anidada.LAST = '||tabla\_anidada.**LAST**);

DBMS\_OUTPUT.PUT\_LINE ('indice\_por\_tabla.FIRST = '||indice\_por\_tabla.**FIRST**);

DBMS\_OUTPUT.PUT\_LINE ('indice\_por\_tabla.LAST = '|| indice\_por\_tabla.**LAST**);

DBMS\_OUTPUT.PUT\_LINE ('tabla\_anidada.PRIOR(2) = '|| tabla\_anidada.**PRIOR**(2));

DBMS\_OUTPUT.PUT\_LINE ('tabla\_anidada.NEXT(2) = '|| tabla\_anidada.**NEXT**(2));

DBMS\_OUTPUT.PUT\_LINE ('indice\_por\_tabla.PRIOR(2) = '|| indice\_por\_tabla.**PRIOR**(2));

DBMS\_OUTPUT.PUT\_LINE ('indice\_por\_tabla.NEXT(2) = '|| indice\_por\_tabla.**NEXT**(2));

tabla\_anidada.**TRIM**(2);

tabla\_anidada.**TRIM**;

DBMS\_OUTPUT.PUT\_LINE('tabla\_anidada.LAST = '||tabla\_anidada.**LAST**);

**END**;

/

*-- Estructura de Control: LOOP*

**DECLARE**

x NUMBER := 0;

**BEGIN**

LOOP *-- Depués de la declaración CONTINUE, el control continua aqui control*

DBMS\_OUTPUT.PUT\_LINE ('Ciclo interno: x = ' || TO\_CHAR(x));

x := x + 1;

*--CONTINUE WHEN x < 3; -- Sentencia válida para oracle 11g*

DBMS\_OUTPUT.PUT\_LINE

('Ciclo interno, despues de CONTINUE: x = ' || TO\_CHAR(x));

EXIT **WHEN** x = 5;

**END** LOOP;

DBMS\_OUTPUT.PUT\_LINE ('Despues del Loop: x = ' || TO\_CHAR(x));

**END**;

/

*-- Estructura de Control: WHILE*

**declare**

v\_test varchar2(8) := 'SIGUE';

n\_numb number := 2;

**begin**

while v\_test <> 'PARA' loop

if n\_numb > 5

**then** v\_test := 'PARA';

**end** if;

dbms\_output.put\_line (v\_test||': '||n\_numb);

n\_numb := n\_numb + 1;

**end** loop;

v\_test := 'ABAJO';

while n\_numb > 1 **AND** v\_test = 'ABAJO' loop

dbms\_output.put\_line (v\_test||': '||n\_numb);

n\_numb := n\_numb - 1;

**end** loop;

while 7 = 4 loop

**NULL**; *-- no entra aqui*

**end** loop;

**end**;

*-- Estructura de Control: FOR*

**BEGIN**

**FOR** i **IN** REVERSE 1..10 LOOP

dbms\_output.put\_line('El valor del indice es ['||i||']');

**END** LOOP;

**END**;

/

*-- Estructura de Control: CASE*

**BEGIN**

**CASE** **TRUE**

**WHEN** (1 > 3) **THEN**

dbms\_output.put\_line('Uno es mayor que tres');

**WHEN** (3 < 5) **THEN**

dbms\_output.put\_line('Tres es menor que cinco');

**WHEN** (1 = 2) **THEN**

dbms\_output.put\_line('Uno es igual a dos');

**ELSE**

dbms\_output.put\_line('Nada es cierto.');

**END** **CASE**;

**END**;

/

**SET** SERVEROUTPUT **ON**

**DECLARE**

**CURSOR** ctrabajador **IS**

**SELECT** nombre, SALPORHORA, HORASTRABAJADAS

**FROM** trabajadores;

vnombre VARCHAR2(50);

vsalporhora number(8,2);

vhorastrabajadas number(10);

**BEGIN**

**OPEN** ctrabajador;

LOOP

**FETCH** ctrabajador **INTO** vnombre,vsalporhora,vhorastrabajadas;

EXIT **WHEN** ctrabajador%NOTFOUND;

**END** LOOP;

**CLOSE** ctrabajador;

**END**;

/