Rodrigo Notario Pérez y David Mata Lorenzo (3ºB)

Práctica 2: El lenguaje de programación PL/SQL

1) Generar manualmente los ficheros ASCII con los siguientes datos.

**Archivo *‘Códigos postales I.txt’*:**

08050;Parets;Barcelona;

14200;Peñarroya;Córdoba;

14900;Lucena;Córdoba;

;Arganda;Sevilla;

08050;Zaragoza;Zaragoza;

28040;Arganda;Madrid;

28004;Madrid;Madrid;

**Archivo *‘Domicilios I.txt’*:**

12345678A;Avda. Complutense;28040;

12345678A;Cántaro;28004;

12345678P;Diamante;14200;

12345678P;Carbón;14901;

2) Crear las tablas "Domicilios I" y "Códigos postales I" con el mismo esquema que Domicilios y " Códigos postales" pero sin restricciones de integridad.

**Archivo ‘*crear\_tablas.sql’*:**

CREATE TABLE "Códigos postales I"("Código postal" Char(5),

                                Población Char(50),

                                Provincia Char(50));

CREATE TABLE "Domicilios I"(DNI Char(9),

                        Calle varchar(50),

                        "Código postal" Char(5));

3) Importar con Oracle Loader las tablas del punto 1.

**Archivo *‘Códigos postales I.ctl’*:**

LOAD DATA

INFILE 'C:\hlocal\Códigos postales I.txt'

APPEND

INTO TABLE "Códigos postales I"

FIELDS TERMINATED BY ';'

("Código postal",Población,Provincia)

**Archivo *‘Códigos postales I informe.txt’*:**

SQL\*Loader: Release 11.2.0.1.0 - Production on Vie Dic 4 09:42:22 2015

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Archivo de Control: Códigos postales I.ctl

Archivo de Datos: C:\hlocal\Códigos postales I.txt

Archivo de Errores: Códigos postales I.bad

Desechar Archivo: ninguno especificado

(Permitir todos los registros desechados)

Número a cargar: ALL

Número a ignorar: 0

Errores permitidos: 50

Matriz de enlace: 64 filas, máximo de 256000 bytes

Continuación: ninguno especificado

Ruta de acceso utilizada: Convencional

Tabla "Códigos postales I", cargada de cada registro lógico.

Opción INSERT activa para esta tabla: APPEND

Nombre Columna Posición Long Term Entorno Tipo de Dato

------------------------------ ---------- ----- ---- ---- ---------------------

"Código postal" FIRST \* ; CHARACTER

POBLACIÓN NEXT \* ; CHARACTER

PROVINCIA NEXT \* ; CHARACTER

Tabla "Códigos postales I":

7 Filas se ha cargado correctamente.

0 Filas no cargada debido a errores de datos.

0 Filas no cargada porque todas las cláusulas WHEN han fallado.

0 Filas no cargada porque todos los campos eran nulos.

Espacio asignado a matriz de enlace: 49536 bytes (64 filas)

Bytes de buffer de lectura: 1048576

Total de registros lógicos ignorados: 0

Total de registros lógicos leídos: 7

Total de registros lógicos rechazados: 0

Total de registros lógicos desechados: 0

La ejecución empezó en Vie Dic 04 09:42:22 2015

La ejecución terminó en Vie Dic 04 09:42:22 2015

Tiempo transcurrido: 00:00:00.16

Tiempo de CPU: 00:00:00.04

**Archivo *‘Domicilios I.ctl’*:**

LOAD DATA

INFILE 'C:\hlocal\Domicilios I.txt'

APPEND

INTO TABLE "Domicilios I"

FIELDS TERMINATED BY ';'

(DNI,Calle,"Código postal")

**Archivo *‘Domicilios I informe.txt’*:**

SQL\*Loader: Release 11.2.0.1.0 - Production on Vie Dic 4 09:43:23 2015

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Archivo de Control: Domicilios I.ctl

Archivo de Datos: C:\hlocal\Domicilios I.txt

Archivo de Errores: Domicilios I.bad

Desechar Archivo: ninguno especificado

(Permitir todos los registros desechados)

Número a cargar: ALL

Número a ignorar: 0

Errores permitidos: 50

Matriz de enlace: 64 filas, máximo de 256000 bytes

Continuación: ninguno especificado

Ruta de acceso utilizada: Convencional

Tabla "Domicilios I", cargada de cada registro lógico.

Opción INSERT activa para esta tabla: APPEND

Nombre Columna Posición Long Term Entorno Tipo de Dato

------------------------------ ---------- ----- ---- ---- ---------------------

DNI FIRST \* ; CHARACTER

CALLE NEXT \* ; CHARACTER

"Código postal" NEXT \* ; CHARACTER

Tabla "Domicilios I":

4 Filas se ha cargado correctamente.

0 Filas no cargada debido a errores de datos.

0 Filas no cargada porque todas las cláusulas WHEN han fallado.

0 Filas no cargada porque todos los campos eran nulos.

Espacio asignado a matriz de enlace: 49536 bytes (64 filas)

Bytes de buffer de lectura: 1048576

Total de registros lógicos ignorados: 0

Total de registros lógicos leídos: 4

Total de registros lógicos rechazados: 0

Total de registros lógicos desechados: 0

La ejecución empezó en Vie Dic 04 09:43:23 2015

La ejecución terminó en Vie Dic 04 09:43:24 2015

Tiempo transcurrido: 00:00:00.12

Tiempo de CPU: 00:00:00.03

4) Escribir un procedimiento almacenado denominado "ComprobarNulos" que permita la detección de valores nulos en la tabla "Códigos postales I" y que emita un error por pantalla que identifique el problema (usar para ello la instrucción RAISE). Para ello hay que definir el cursor con una instrucción SQL que devuelva las tuplas con nulos en alguno de sus campos. Después se recorrerá este cursor y se mostrará la primera tupla con un error.

SET SERVEROUTPUT ON SIZE 100000;

CREATE OR REPLACE PROCEDURE ComprobarNulos IS

--DECLARE

v\_CodigoPostal "Códigos postales I"."Código postal"%TYPE;

v\_Poblacion "Códigos postales I".Población%TYPE;

v\_Provincia "Códigos postales I".Provincia%TYPE;

CURSOR c\_CodigosPostalesI IS

SELECT "Código postal", Población, Provincia

FROM "Códigos postales I";

excepcion\_nulo EXCEPTION;

BEGIN

OPEN c\_CodigosPostalesI;

LOOP

FETCH c\_CodigosPostalesI INTO v\_CodigoPostal, v\_Poblacion, v\_Provincia;

IF v\_CodigoPostal IS NULL OR v\_Poblacion IS NULL OR v\_Provincia IS NULL THEN

RAISE excepcion\_nulo;

END IF;

EXIT WHEN c\_CodigosPostalesI%NOTFOUND;

END LOOP;

CLOSE c\_CodigosPostalesI;

--END;

EXCEPTION

WHEN excepcion\_nulo THEN

DBMS\_OUTPUT.PUT\_LINE ('Código postal nulo.');

END;

**Resultado:**

bloque anónimo terminado

Código postal nulo.

5) Crear un procedimiento almacenado denominado "ComprobarPK" que permita detectar la violación de clave primaria en la tabla la tabla "Códigos postales I" y que emita por pantalla un error que identifique el problema. Se debe aplicar un procedimiento similar al apartado 4.

SET SERVEROUTPUT ON SIZE 100000;

CREATE OR REPLACE PROCEDURE ComprobarPK IS

--DECLARE

v\_CodigoPostal "Códigos postales I"."Código postal"%TYPE;

v\_CodigoPostal2 "Códigos postales I"."Código postal"%TYPE;

CURSOR c\_CodigosPostalesI IS

SELECT "Código postal"

FROM "Códigos postales I";

CURSOR c\_CodigosPostalesI2 IS

SELECT "Código postal"

FROM "Códigos postales I";

excepcion\_repetido EXCEPTION;

BEGIN

OPEN c\_CodigosPostalesI;

OPEN c\_CodigosPostalesI2;

LOOP

FETCH c\_CodigosPostalesI INTO v\_CodigoPostal;

FETCH c\_CodigosPostalesI2 INTO v\_CodigoPostal2;

IF v\_CodigoPostal = v\_CodigoPostal2 THEN

RAISE excepcion\_repetido;

END IF;

EXIT WHEN c\_CodigosPostalesI%NOTFOUND;

END LOOP;

CLOSE c\_CodigosPostalesI;

CLOSE c\_CodigosPostalesI2;

--END;

EXCEPTION

WHEN excepcion\_repetido THEN

DBMS\_OUTPUT.PUT\_LINE ('Violación de clave primaria en la tabla la tabla Códigos postales I.');

DBMS\_OUTPUT.PUT\_LINE (v\_CodigoPostal);

END;

**Resultado:**

bloque anónimo terminado

Violación de clave primaria en la tabla la tabla Códigos postales I.

08050

6) Crear un procedimiento almacenado denominado "ComprobarFD"que permita detectar la violación de dependencia funcional en la tabla "Códigos postales I" y que emita por pantalla un error que identifique el problema. Se debe aplicar un procedimiento similar al apartado 4.

SET SERVEROUTPUT ON SIZE 100000;

CREATE OR REPLACE PROCEDURE ComprobarFD IS

--DECLARE

CURSOR c\_CodigosPostalesI IS

SELECT CP1."Código postal", CP1.Población, CP1.Provincia

FROM "Códigos postales I" CP1, "Códigos postales I" CP2

WHERE CP1.Población = CP2.Población AND CP1.Provincia <> CP2.Provincia;

BEGIN

FOR c\_CP IN c\_CodigosPostalesI LOOP

DBMS\_OUTPUT.PUT\_LINE('Violación de dependencia funcional en la tabla Códigos postales I.');

DBMS\_OUTPUT.PUT\_LINE(c\_CP.Población || ', ' || c\_CP.Población || ', ' || c\_CP.Provincia);

END LOOP;

END;

**Resultado:**

bloque anónimo terminado

Violación de dependencia funcional en la tabla Códigos postales I.

Arganda , Arganda , Madrid

Violación de dependencia funcional en la tabla Códigos postales I.

Arganda , Arganda , Sevilla

7) Crear un procedimiento almacenado denominado "ComprobarFK"que permita detectar la violación de integridad referencial en la tabla "Domicilios I" y que emita un error por pantalla que identifique el problema. Se debe aplicar un procedimiento similar al apartado 4. Para comprobar este apartado hay que reemplazar el valor nulo encontrado en el apartado 4 por el valor '14900'.

SET SERVEROUTPUT ON SIZE 100000;

CREATE OR REPLACE PROCEDURE ComprobarFK IS

--DECLARE

CURSOR c\_CodigosPostalesI IS

SELECT CP."Código postal", CP.Población, CP.Provincia

FROM "Códigos postales I" CP LEFT JOIN "Domicilios I" D

ON CP."Código postal" = D."Código postal"

WHERE D."Código postal" IS NULL;

BEGIN

FOR r\_CP IN c\_CodigosPostalesI LOOP

DBMS\_OUTPUT.PUT\_LINE('Violación de integridad referencial en la tabla "Domicilios I"');

DBMS\_OUTPUT.PUT\_LINE(r\_CP."Código postal" || ', ' || r\_CP.Provincia || ', ' || r\_CP.Población);

END LOOP;

END;

**Resultado:**

bloque anónimo terminado

Violación de integridad referencial en la tabla "Domicilios I"

14900, Sevilla , Arganda

Violación de integridad referencial en la tabla "Domicilios I"

14900, Córdoba , Lucena

Violación de integridad referencial en la tabla "Domicilios I"

08050, Zaragoza , Zaragoza

Violación de integridad referencial en la tabla "Domicilios I"

08050, Barcelona , Parets