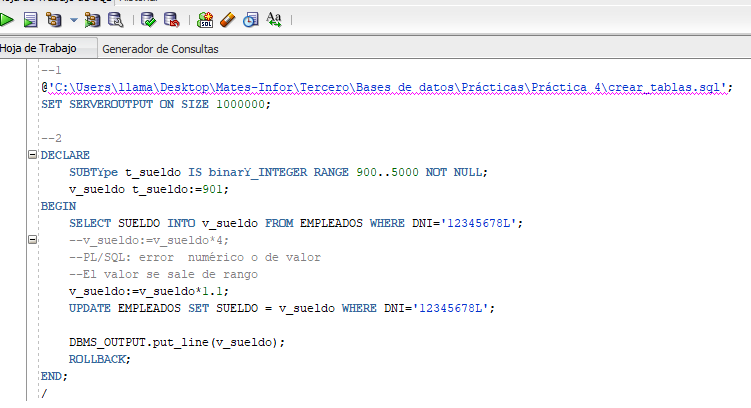
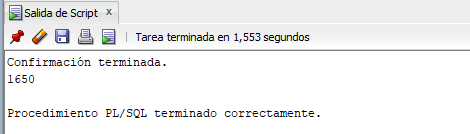
**PRÁCTICA 4**

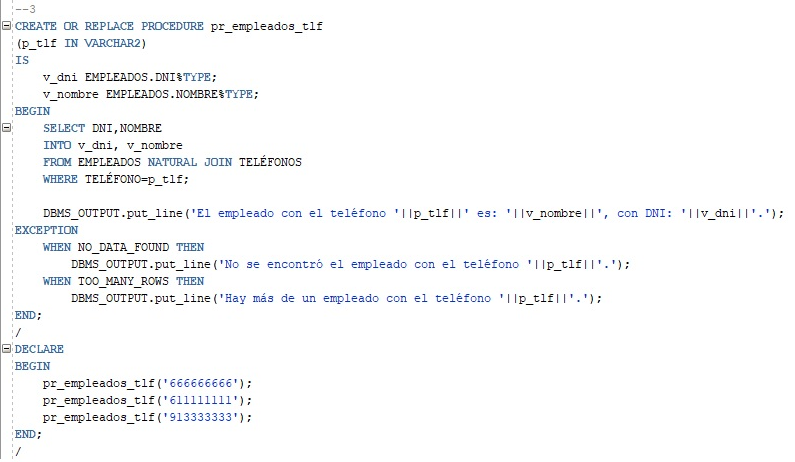
JUAN CARLOS LLAMAS NÚÑEZ 3ºDG

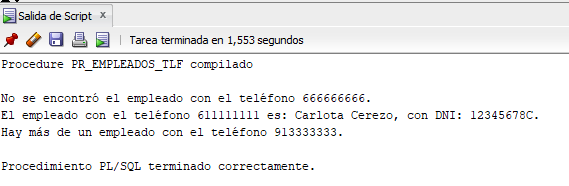
En este documento se presentan las capturas del código para facilitar la legibilidad, así como la salida por consola que generan las consultas. Al final del documento se añade el texto plano del archivo .sql y la salida de consola.

**1 y 2**

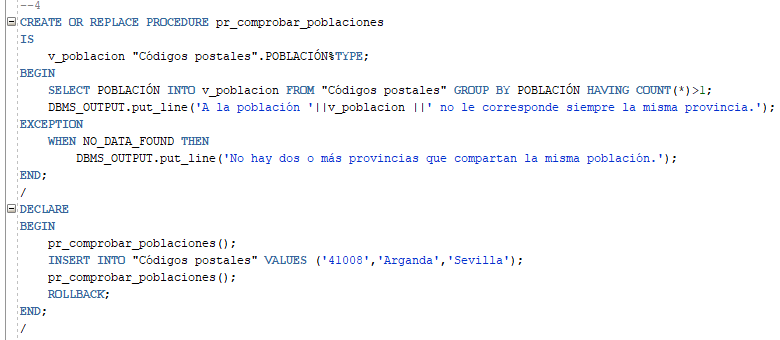


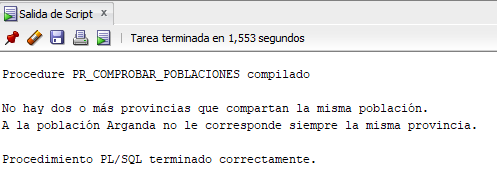
**3**



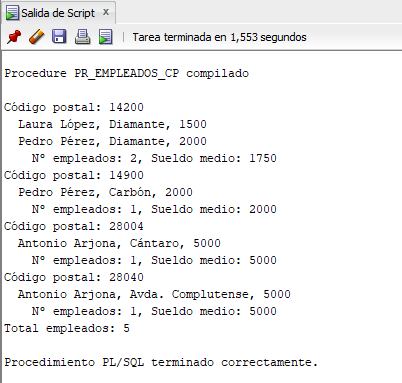
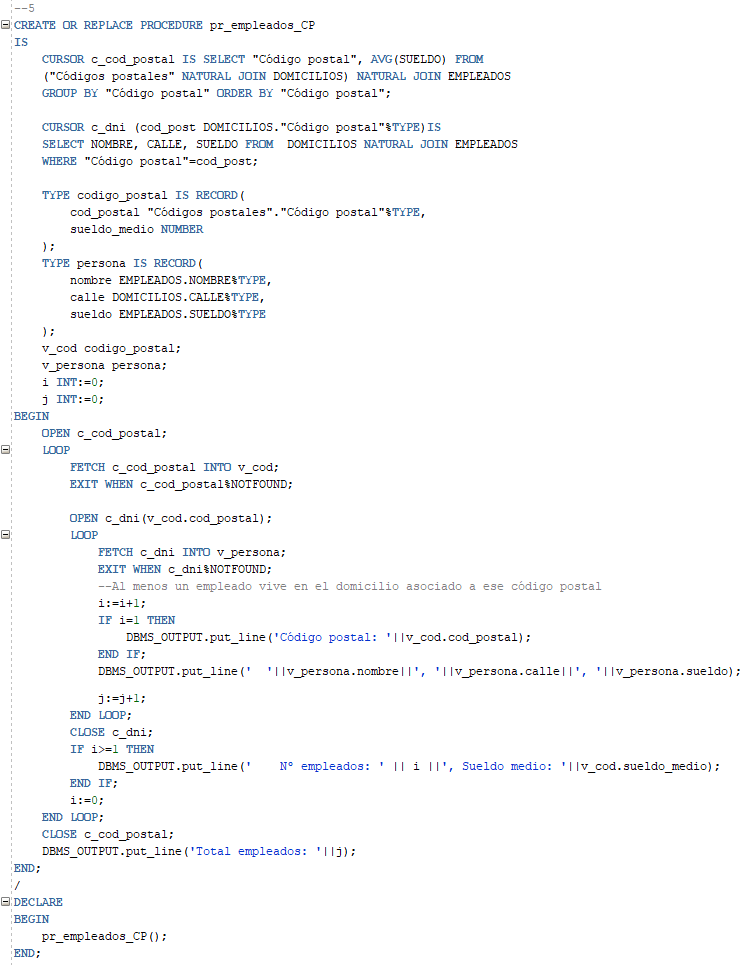


**4**

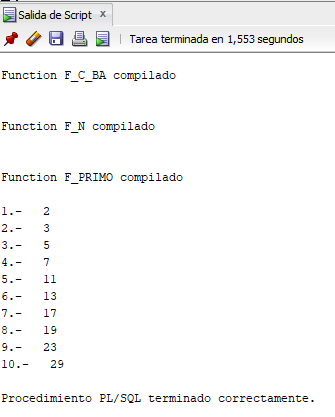
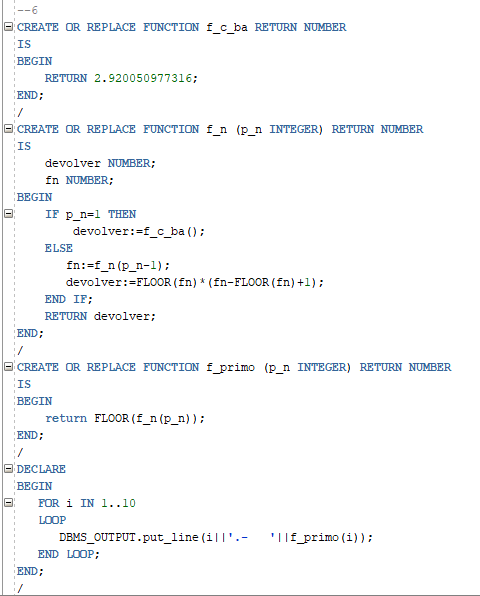




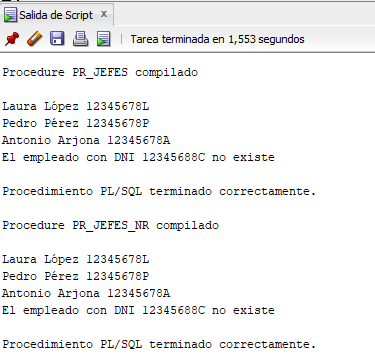
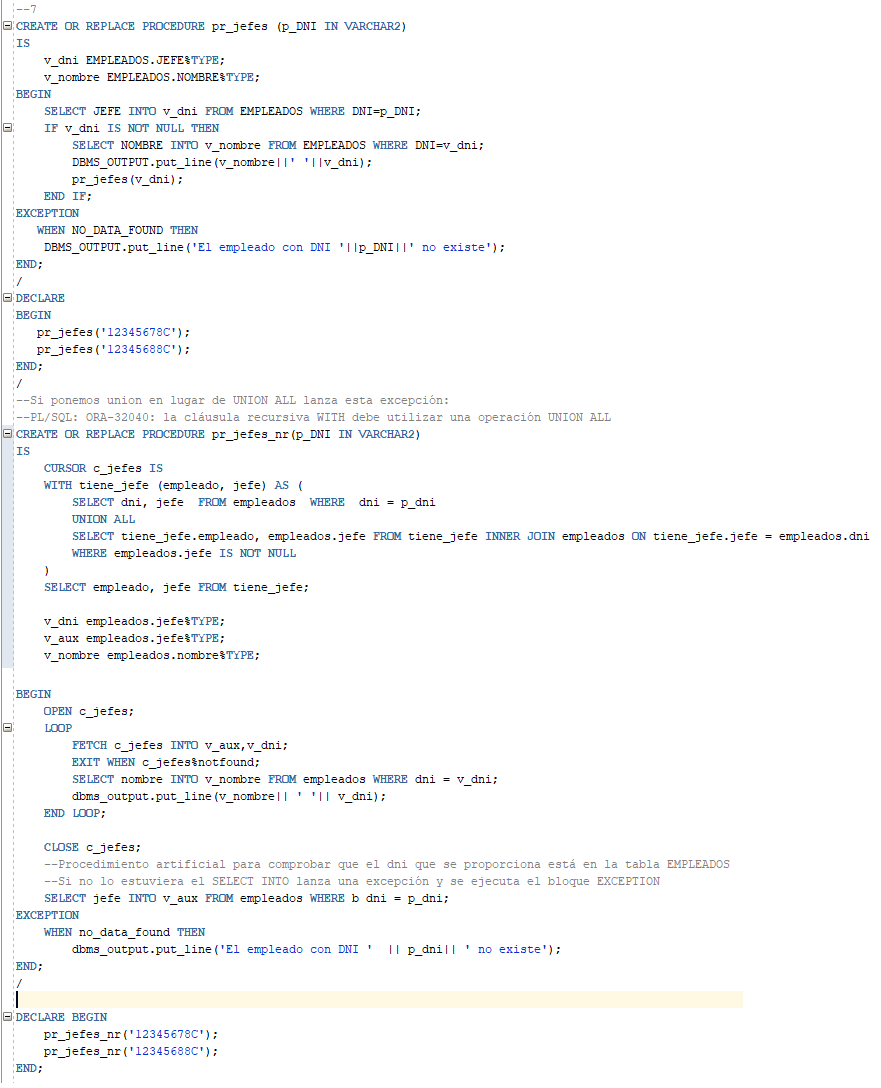
**5**



**6**



**7**



**ARCHIVO .SQL EN TEXTO PLANO**

--1

@'C:\Users\llama\Desktop\Mates-Infor\Tercero\Bases de datos\Prácticas\Práctica 4\crear\_tablas.sql';

SET SERVEROUTPUT ON SIZE 1000000;

--2

DECLARE

SUBTYpe t\_sueldo IS binarY\_INTEGER RANGE 900..5000 NOT NULL;

v\_sueldo t\_sueldo:=901;

BEGIN

SELECT SUELDO INTO v\_sueldo FROM EMPLEADOS WHERE DNI='12345678L';

--v\_sueldo:=v\_sueldo\*4;

--PL/SQL: error numérico o de valor

--El valor se sale de rango

v\_sueldo:=v\_sueldo\*1.1;

UPDATE EMPLEADOS SET SUELDO = v\_sueldo WHERE DNI='12345678L';

DBMS\_OUTPUT.put\_line(v\_sueldo);

ROLLBACK;

END;

/

--3

CREATE OR REPLACE PROCEDURE pr\_empleados\_tlf

(p\_tlf IN VARCHAR2)

IS

v\_dni EMPLEADOS.DNI%TYPE;

v\_nombre EMPLEADOS.NOMBRE%TYPE;

BEGIN

SELECT DNI,NOMBRE

INTO v\_dni, v\_nombre

FROM EMPLEADOS NATURAL JOIN TELÉFONOS

WHERE TELÉFONO=p\_tlf;

DBMS\_OUTPUT.put\_line('El empleado con el teléfono '||p\_tlf||' es: '||v\_nombre||', con DNI: '||v\_dni||'.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.put\_line('No se encontró el empleado con el teléfono '||p\_tlf||'.');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.put\_line('Hay más de un empleado con el teléfono '||p\_tlf||'.');

END;

/

DECLARE

BEGIN

pr\_empleados\_tlf('666666666');

pr\_empleados\_tlf('611111111');

pr\_empleados\_tlf('913333333');

END;

/

--4

CREATE OR REPLACE PROCEDURE pr\_comprobar\_poblaciones

IS

v\_poblacion "Códigos postales".POBLACIÓN%TYPE;

BEGIN

SELECT POBLACIÓN INTO v\_poblacion FROM "Códigos postales" GROUP BY POBLACIÓN HAVING COUNT(\*)>1;

DBMS\_OUTPUT.put\_line('A la población '||v\_poblacion ||' no le corresponde siempre la misma provincia.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.put\_line('No hay dos o más provincias que compartan la misma población.');

END;

/

DECLARE

BEGIN

pr\_comprobar\_poblaciones();

INSERT INTO "Códigos postales" VALUES ('41008','Arganda','Sevilla');

pr\_comprobar\_poblaciones();

ROLLBACK;

END;

/

--5

CREATE OR REPLACE PROCEDURE pr\_empleados\_CP

IS

CURSOR c\_cod\_postal IS SELECT "Código postal", AVG(SUELDO) FROM

("Códigos postales" NATURAL JOIN DOMICILIOS) NATURAL JOIN EMPLEADOS

GROUP BY "Código postal" ORDER BY "Código postal";

CURSOR c\_dni (cod\_post DOMICILIOS."Código postal"%TYPE)IS

SELECT NOMBRE, CALLE, SUELDO FROM DOMICILIOS NATURAL JOIN EMPLEADOS

WHERE "Código postal"=cod\_post;

TYPE codigo\_postal IS RECORD(

cod\_postal "Códigos postales"."Código postal"%TYPE,

sueldo\_medio NUMBER

);

TYPE persona IS RECORD(

nombre EMPLEADOS.NOMBRE%TYPE,

calle DOMICILIOS.CALLE%TYPE,

sueldo EMPLEADOS.SUELDO%TYPE

);

v\_cod codigo\_postal;

v\_persona persona;

i INT:=0;

j INT:=0;

BEGIN

OPEN c\_cod\_postal;

LOOP

FETCH c\_cod\_postal INTO v\_cod;

EXIT WHEN c\_cod\_postal%NOTFOUND;

OPEN c\_dni(v\_cod.cod\_postal);

LOOP

FETCH c\_dni INTO v\_persona;

EXIT WHEN c\_dni%NOTFOUND;

--Al menos un empleado vive en el domicilio asociado a ese código postal

i:=i+1;

IF i=1 THEN

DBMS\_OUTPUT.put\_line('Código postal: '||v\_cod.cod\_postal);

END IF;

DBMS\_OUTPUT.put\_line(' '||v\_persona.nombre||', '||v\_persona.calle||', '||v\_persona.sueldo);

j:=j+1;

END LOOP;

CLOSE c\_dni;

IF i>=1 THEN

DBMS\_OUTPUT.put\_line(' Nº empleados: ' || i ||', Sueldo medio: '||v\_cod.sueldo\_medio);

END IF;

i:=0;

END LOOP;

CLOSE c\_cod\_postal;

DBMS\_OUTPUT.put\_line('Total empleados: '||j);

END;

/

DECLARE

BEGIN

pr\_empleados\_CP();

END;

/

--6

CREATE OR REPLACE FUNCTION f\_c\_ba RETURN NUMBER

IS

BEGIN

RETURN 2.920050977316;

END;

/

CREATE OR REPLACE FUNCTION f\_n (p\_n INTEGER) RETURN NUMBER

IS

devolver NUMBER;

fn NUMBER;

BEGIN

IF p\_n=1 THEN

devolver:=f\_c\_ba();

ELSE

fn:=f\_n(p\_n-1);

devolver:=FLOOR(fn)\*(fn-FLOOR(fn)+1);

END IF;

RETURN devolver;

END;

/

CREATE OR REPLACE FUNCTION f\_primo (p\_n INTEGER) RETURN NUMBER

IS

BEGIN

return FLOOR(f\_n(p\_n));

END;

/

DECLARE

BEGIN

FOR i IN 1..10

LOOP

DBMS\_OUTPUT.put\_line(i||'.- '||f\_primo(i));

END LOOP;

END;

/

--7

CREATE OR REPLACE PROCEDURE pr\_jefes (p\_DNI IN VARCHAR2)

IS

v\_dni EMPLEADOS.JEFE%TYPE;

v\_nombre EMPLEADOS.NOMBRE%TYPE;

BEGIN

SELECT JEFE INTO v\_dni FROM EMPLEADOS WHERE DNI=p\_DNI;

IF v\_dni IS NOT NULL THEN

SELECT NOMBRE INTO v\_nombre FROM EMPLEADOS WHERE DNI=v\_dni;

DBMS\_OUTPUT.put\_line(v\_nombre||' '||v\_dni);

pr\_jefes(v\_dni);

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.put\_line('El empleado con DNI '||p\_DNI||' no existe');

END;

/

DECLARE

BEGIN

pr\_jefes('12345678C');

pr\_jefes('12345688C');

END;

/

--Si ponemos union en lugar de UNION ALL lanza esta excepción:

--PL/SQL: ORA-32040: la cláusula recursiva WITH debe utilizar una operación UNION ALL

CREATE OR REPLACE PROCEDURE pr\_jefes\_nr(p\_DNI IN VARCHAR2)

IS

CURSOR c\_jefes IS

WITH tiene\_jefe (empleado, jefe) AS (

SELECT dni, jefe FROM empleados WHERE dni = p\_dni

UNION ALL

SELECT tiene\_jefe.empleado, empleados.jefe FROM tiene\_jefe INNER JOIN empleados ON tiene\_jefe.jefe = empleados.dni

WHERE empleados.jefe IS NOT NULL

)

SELECT empleado, jefe FROM tiene\_jefe;

v\_dni empleados.jefe%TYPE;

v\_aux empleados.jefe%TYPE;

v\_nombre empleados.nombre%TYPE;

BEGIN

OPEN c\_jefes;

LOOP

FETCH c\_jefes INTO v\_aux,v\_dni;

EXIT WHEN c\_jefes%notfound;

SELECT nombre INTO v\_nombre FROM empleados WHERE dni = v\_dni;

dbms\_output.put\_line(v\_nombre|| ' '|| v\_dni);

END LOOP;

CLOSE c\_jefes;

--Procedimiento artificial para comprobar que el dni que se proporciona está en la tabla EMPLEADOS

--Si no lo estuviera el SELECT INTO lanza una excepción y se ejecuta el bloque EXCEPTION

SELECT jefe INTO v\_aux FROM empleados WHERE b dni = p\_dni;

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('El empleado con DNI ' || p\_dni|| ' no existe');

END;

/

DECLARE BEGIN

pr\_jefes\_nr('12345678C');

pr\_jefes\_nr('12345688C');

END;

**SALIDA POR CONSOLA EN TEXTO PLANO**

1650

Procedimiento PL/SQL terminado correctamente.

Procedure PR\_EMPLEADOS\_TLF compilado

No se encontró el empleado con el teléfono 666666666.

El empleado con el teléfono 611111111 es: Carlota Cerezo, con DNI: 12345678C.

Hay más de un empleado con el teléfono 913333333.

Procedimiento PL/SQL terminado correctamente.

Procedure PR\_COMPROBAR\_POBLACIONES compilado

No hay dos o más provincias que compartan la misma población.

A la población Arganda no le corresponde siempre la misma provincia.

Procedimiento PL/SQL terminado correctamente.

Procedure PR\_EMPLEADOS\_CP compilado

Código postal: 14200

Laura López, Diamante, 1500

Pedro Pérez, Diamante, 2000

Nº empleados: 2, Sueldo medio: 1750

Código postal: 14900

Pedro Pérez, Carbón, 2000

Nº empleados: 1, Sueldo medio: 2000

Código postal: 28004

Antonio Arjona, Cántaro, 5000

Nº empleados: 1, Sueldo medio: 5000

Código postal: 28040

Antonio Arjona, Avda. Complutense, 5000

Nº empleados: 1, Sueldo medio: 5000

Total empleados: 5

Procedimiento PL/SQL terminado correctamente.

Function F\_C\_BA compilado

Function F\_N compilado

Function F\_PRIMO compilado

1.- 2

2.- 3

3.- 5

4.- 7

5.- 11

6.- 13

7.- 17

8.- 19

9.- 23

10.- 29

Procedimiento PL/SQL terminado correctamente.

Procedure PR\_JEFES compilado

Laura López 12345678L

Pedro Pérez 12345678P

Antonio Arjona 12345678A

El empleado con DNI 12345688C no existe

Procedimiento PL/SQL terminado correctamente.

Procedure PR\_JEFES\_NR compilado

Laura López 12345678L

Pedro Pérez 12345678P

Antonio Arjona 12345678A

El empleado con DNI 12345688C no existe

Procedimiento PL/SQL terminado correctamente.