

Centro de Estudios Superiores AFUERA

Curso 2021-2022

BASES DE DATOS

Cuadernillo de prácticas de PL/SQL

1ª PARTE

Instrucciones:

En este documento Word los alumnos irán realizando los ejercicios de Base de Datos y se usará como documento para las tareas y entregas parciales según se vayan completando cada uno de los temas. El documento se salvará con el nombre PLSQL_1erApellido_Nombre_A (formato pdf o docx u odt)

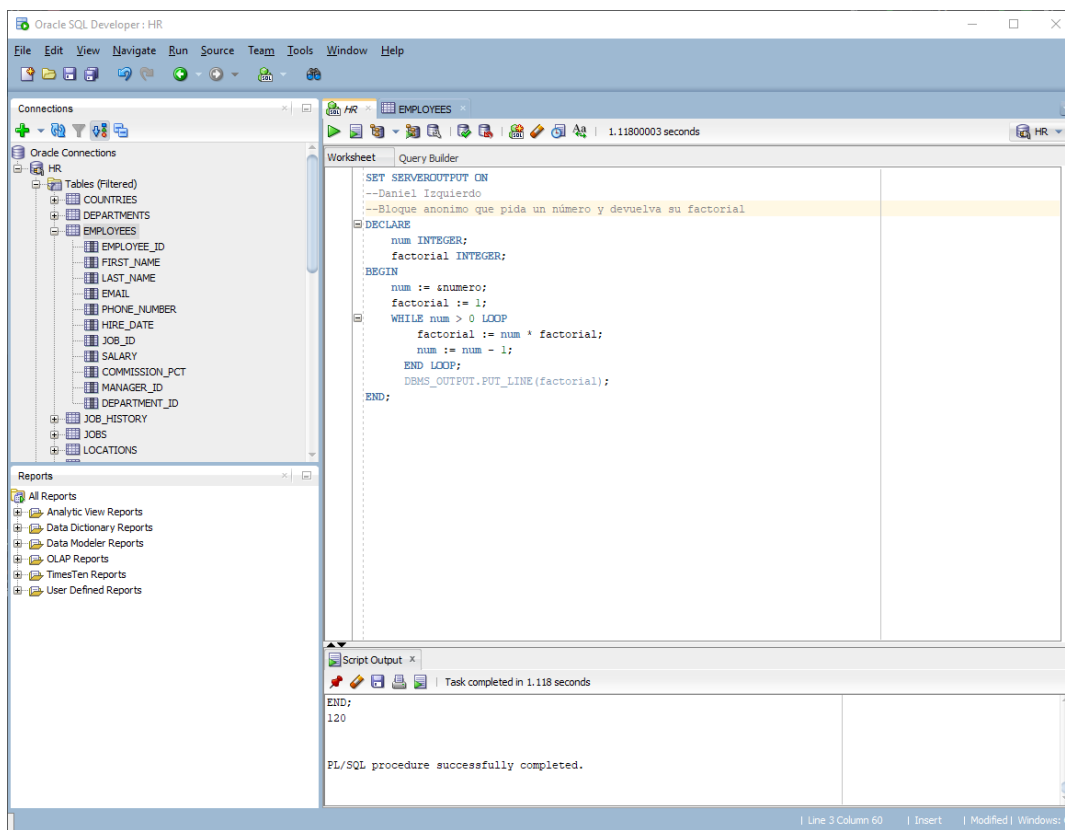
NOTA: Las soluciones de cada ejercicio constarán de una captura (o dos si fuese necesario) incluyendo el código y los resultados arrojados por el programa. El código debe incluir una línea de comentario con el nombre completo del autor y otra línea explicando lo que hace.

-

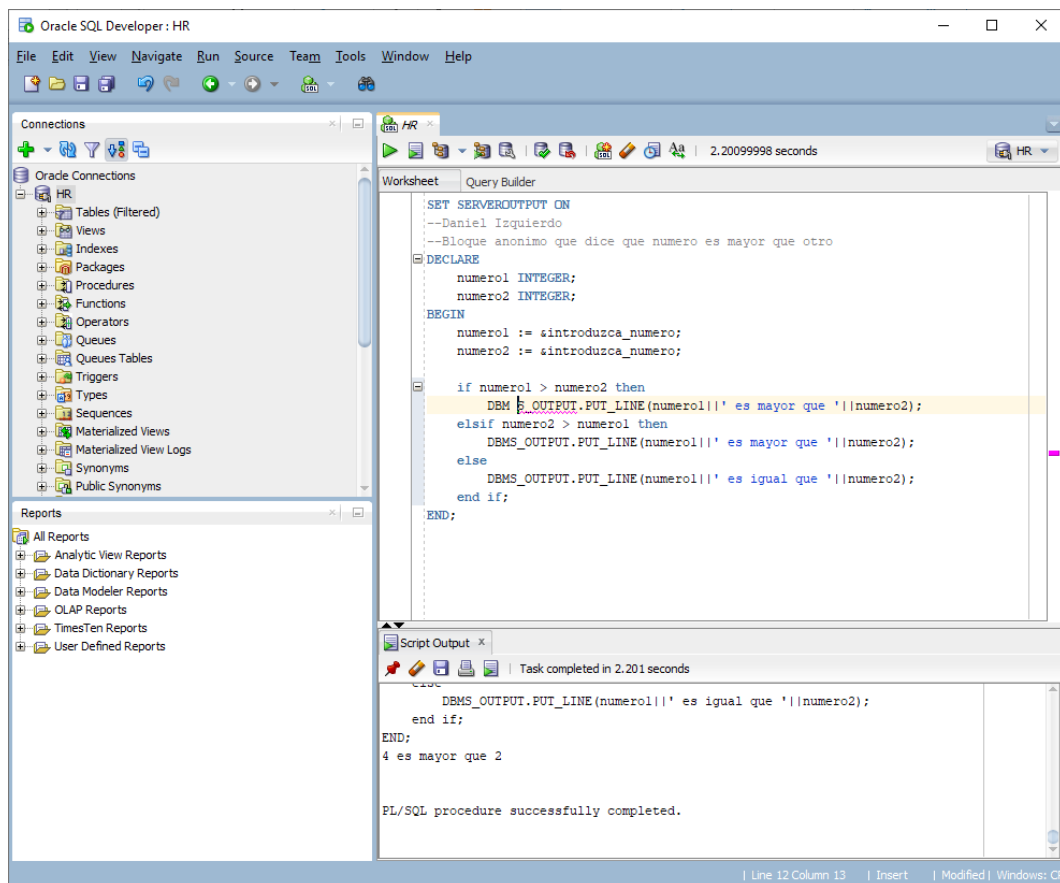
Nombre completo del alumno: Daniel Izquierdo Bonilla

EJERCICIOS DE BLOQUES ANÓNIMOS

1. Hacer un bloque anónimo que pida un número y devuelva su factorial

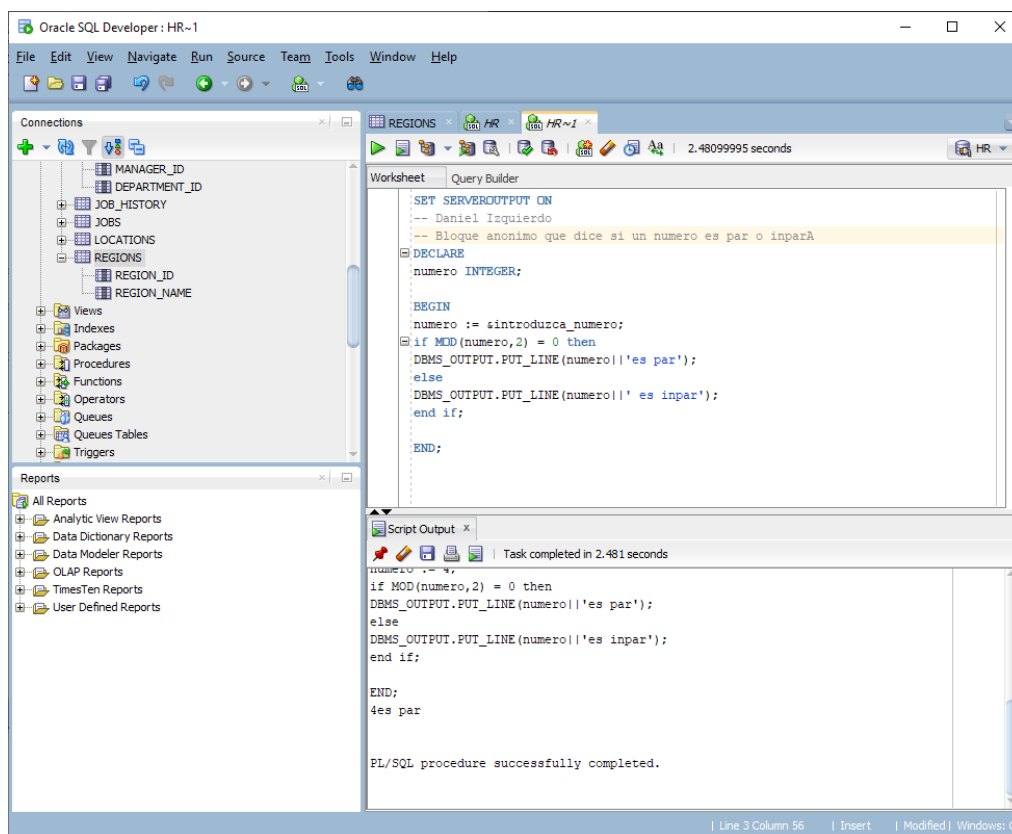


2. Hacer un bloque anónimo que pida dos números y cual es mayor que el otro

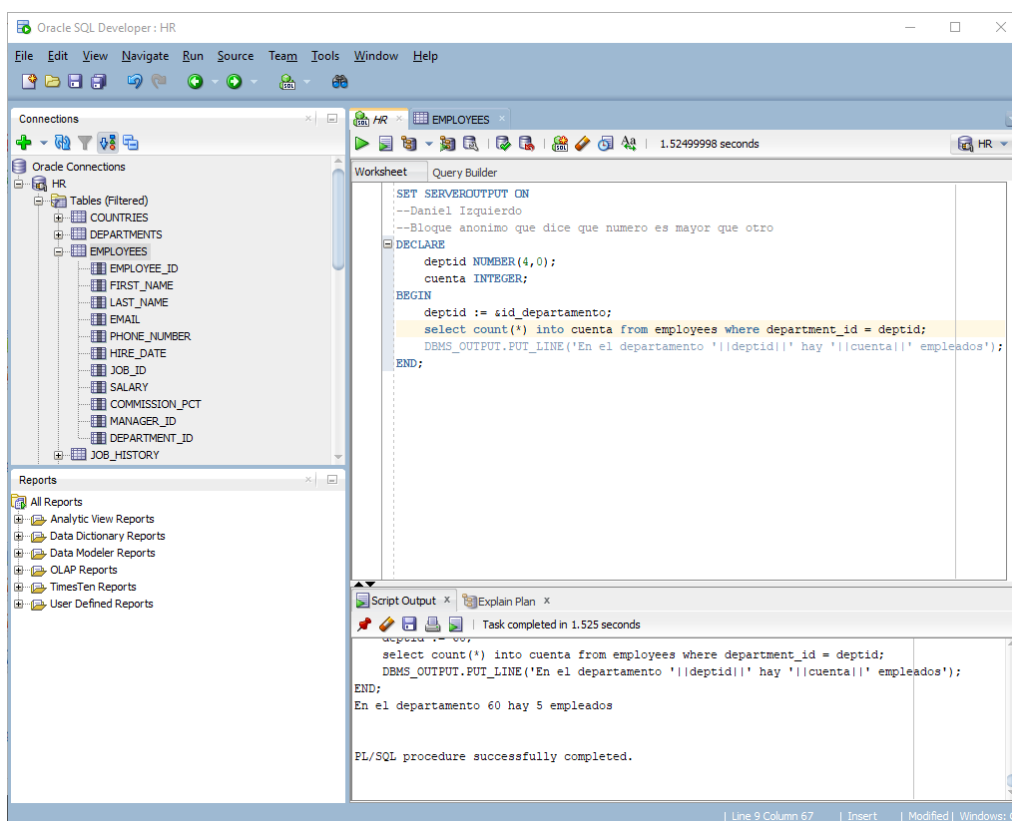


Nombre completo del alumno: Daniel Izquierdo Bonilla

3. Hacer un bloque anónimo que pida un numero entero y diga si es par o impar

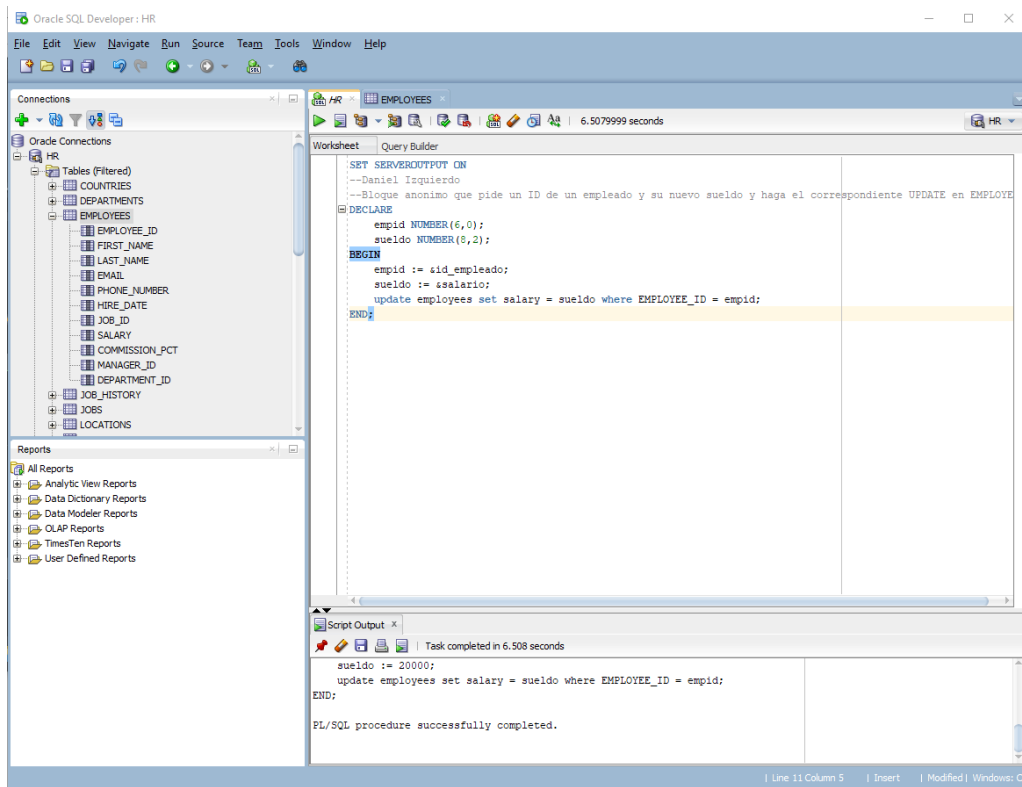


4. Hacer un bloque anónimo que cuente los empleados de un determinado departamento (del cual se pasa su ID)

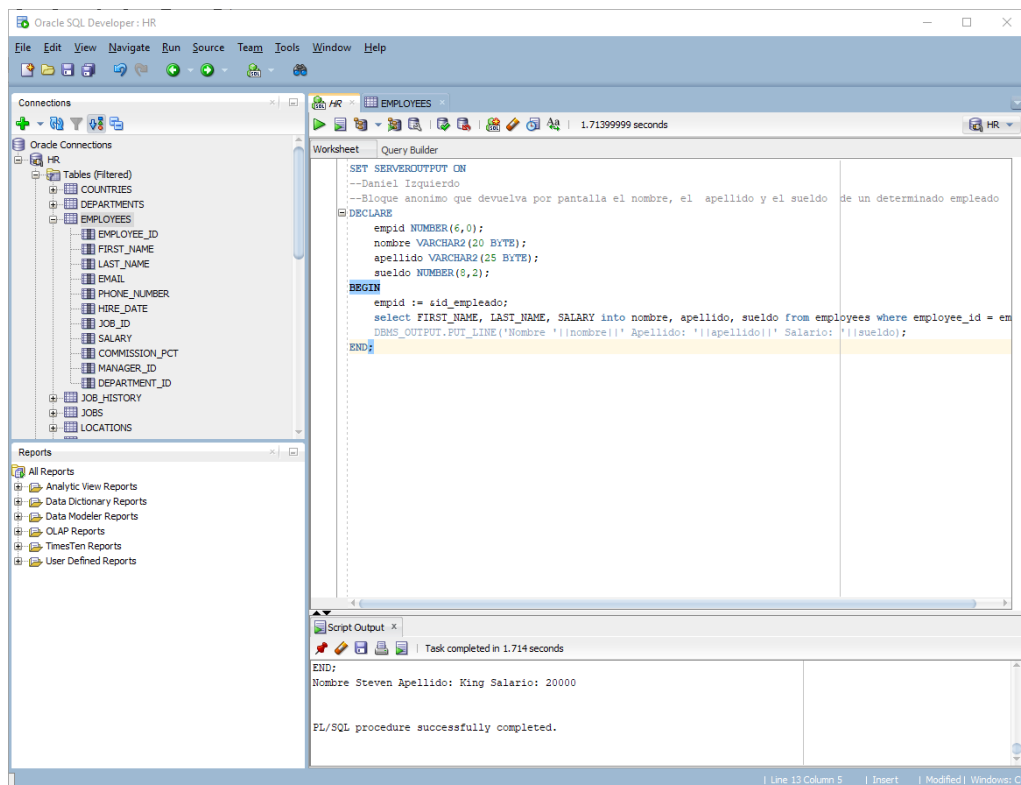


Nombre completo del alumno: Daniel Izquierdo Bonilla

5. Hacer un bloque anónimo que pida un ID de un empleado y su nuevo sueldo y haga el correspondiente UPDATE en EMPLOYEES



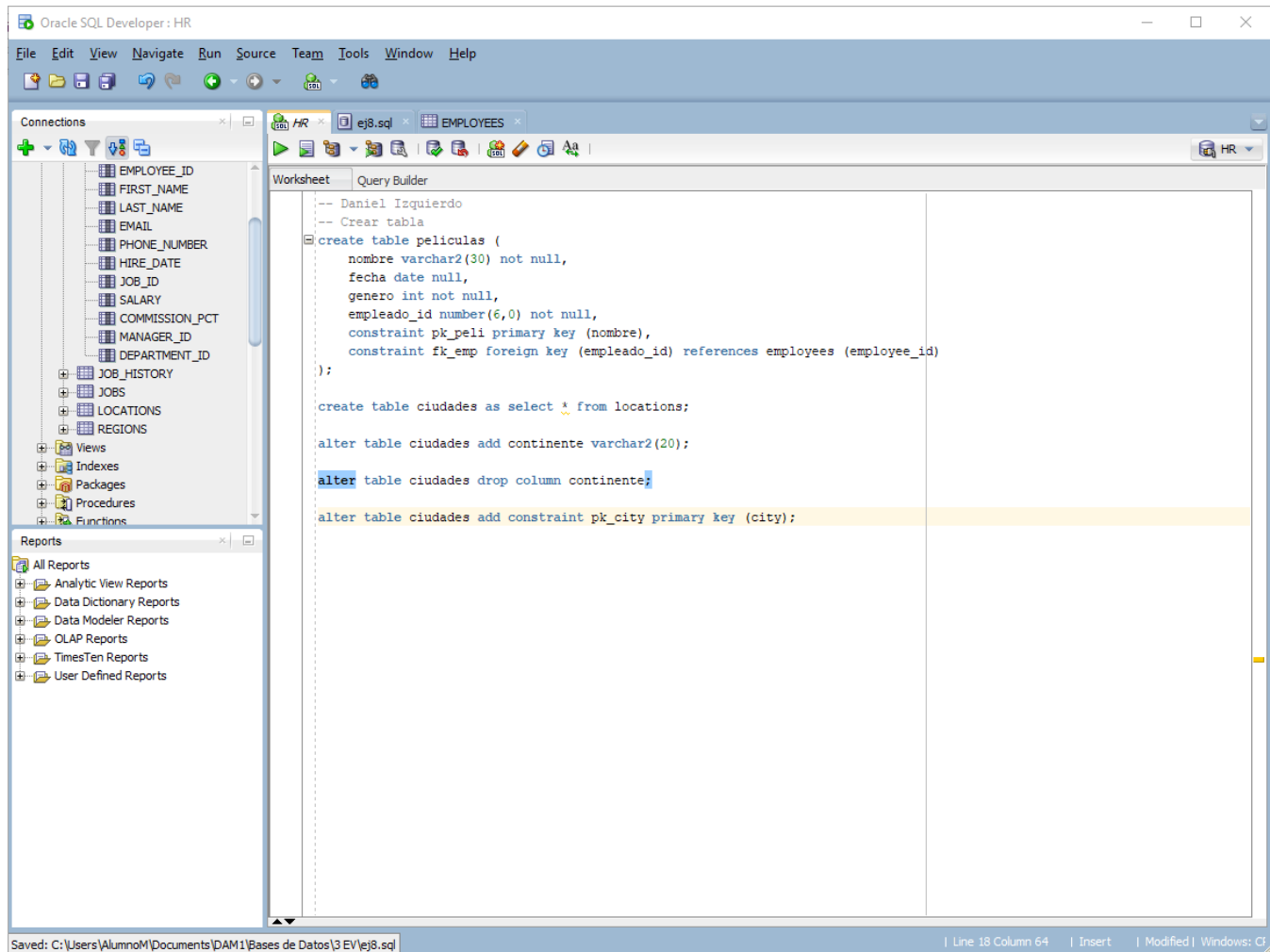
6. Hacer un bloque anónimo que devuelva por pantalla el nombre, el apellido y el sueldo de un determinado empleado del cual se pasa su ID



EJERCICIOS DE EXCEPCIONES

7. Ejercicio repaso de DDL:

- CREAR UNA TABLA
- CREAR UNA TABLA USANDO UNA SUBCONSULTA
- AÑADIR (O BORRAR) UNA COLUMNA (sobre una de las tablas que acabamos de crear)
- AÑADIR INDICES, CLAVES FORÁNEAS ETC (sobre una de las tablas que acabamos de crear)

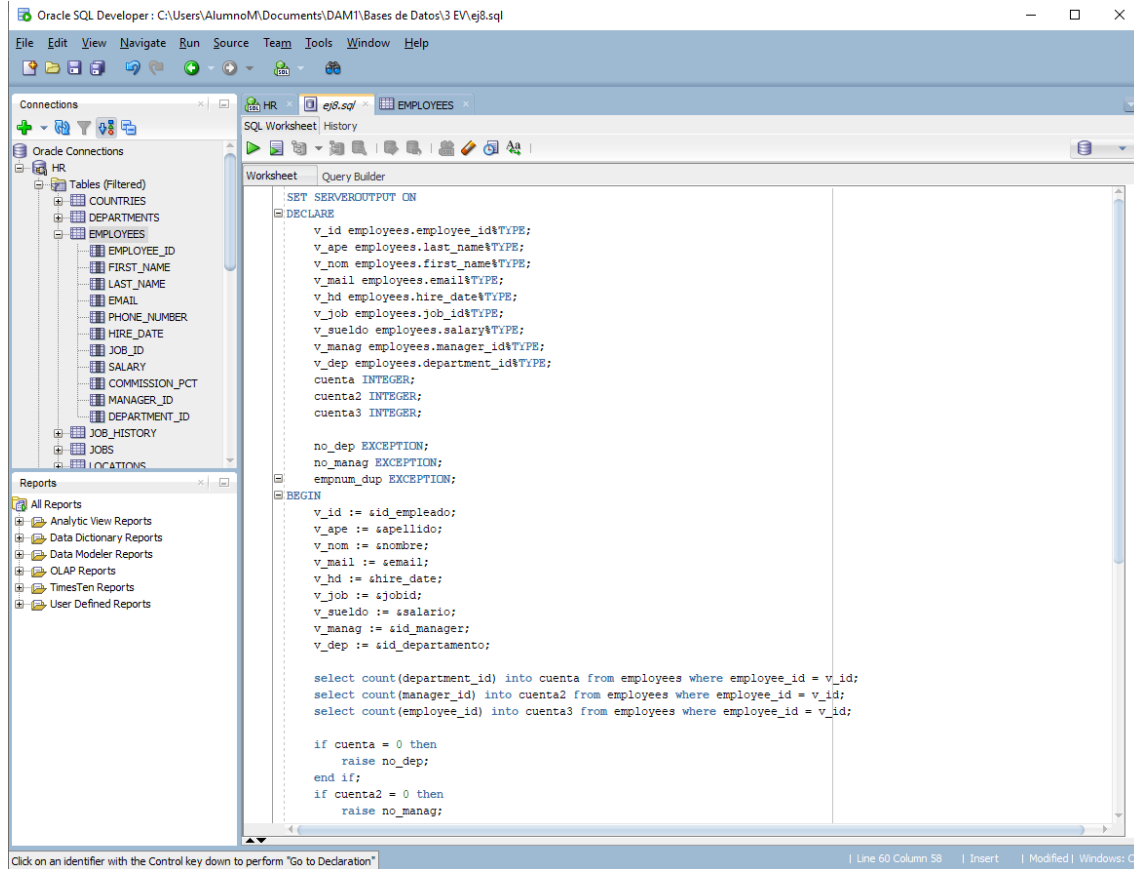


8. Ejercicio de EXCEPCIONES:

Escribe un bloque anónimo que reciba todos los datos de un nuevo empleado y procese la transacción de alta, gestionando posibles errores. El procedimiento deberá gestionar en concreto los siguientes puntos:

- no_existe_departamento.
- no_existe_director.
- numero_empleado_duplicado.
- Salario nulo: con RAISE_APPLICATION_ERROR.
- Otros posibles errores de Oracle visualizando código de error y el mensaje de error.

Nombre completo del alumno: Daniel Izquierdo Bonilla



Oracle SQL Developer: C:\Users\AlumnoM\Documents\DA11\Bases de Datos\3 EV\ej8.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

Oracle Connections

HR

Tables (Filtered)

COUNTRIES

DEPARTMENTS

EMPLOYEES

EMPLOYEE_ID

FIRST_NAME

LAST_NAME

EMAIL

PHONE_NUMBER

HIRE_DATE

JOB_ID

SALARY

COMMISSION_PCT

MANAGER_ID

DEPARTMENT_ID

JOB_HISTORY

JOBS

LOCATIONS

Reports

All Reports

Analytic View Reports

Data Dictionary Reports

Data Modeler Reports

OLAP Reports

TimesTen Reports

User Defined Reports

SQL Worksheet: History

Worksheet

Query Builder

```
SET SERVEROUTPUT ON
DECLARE
    v_id employees.employee_id%TYPE;
    v_ape employees.last_name%TYPE;
    v_nom employees.first_name%TYPE;
    v_mail employees.email%TYPE;
    v_hd employees.hire_date%TYPE;
    v_job employees.job_id%TYPE;
    v_sueldo employees.salary%TYPE;
    v_manag employees.manager_id%TYPE;
    v_dep employees.department_id%TYPE;
    cuenta INTEGER;
    cuenta2 INTEGER;
    cuenta3 INTEGER;

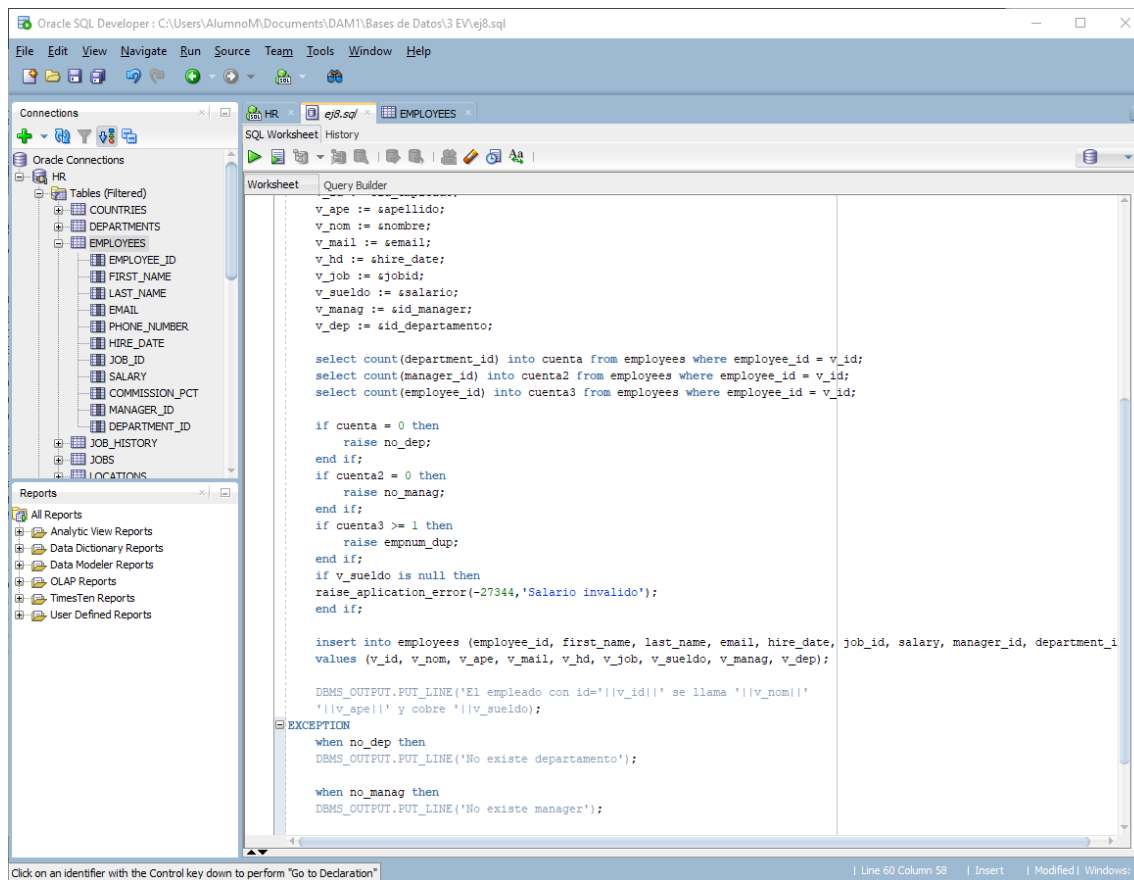
    no_dep EXCEPTION;
    no_manag EXCEPTION;
    empnum_dup EXCEPTION;

BEGIN
    v_id := sid_empleado;
    v_ape := sapellido;
    v_nom := snombre;
    v_mail := semail;
    v_hd := shire_date;
    v_job := sjobid;
    v_sueldo := ssalario;
    v_manag := sid_manager;
    v_dep := sid_departamento;

    select count(department_id) into cuenta from employees where employee_id = v_id;
    select count(manager_id) into cuenta2 from employees where employee_id = v_id;
    select count(employee_id) into cuenta3 from employees where employee_id = v_id;

    if cuenta = 0 then
        raise no_dep;
    end if;
    if cuenta2 = 0 then
        raise no_manag;
    end if;
```

Click on an identifier with the Control key down to perform "Go to Declaration" | Line 60 Column 58 | Insert | Modified | Windows: C



Oracle SQL Developer: C:\Users\AlumnoM\Documents\DA11\Bases de Datos\3 EV\ej8.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

Oracle Connections

HR

Tables (Filtered)

COUNTRIES

DEPARTMENTS

EMPLOYEES

EMPLOYEE_ID

FIRST_NAME

LAST_NAME

EMAIL

PHONE_NUMBER

HIRE_DATE

JOB_ID

SALARY

COMMISSION_PCT

MANAGER_ID

DEPARTMENT_ID

JOB_HISTORY

JOBS

LOCATIONS

Reports

All Reports

Analytic View Reports

Data Dictionary Reports

Data Modeler Reports

OLAP Reports

TimesTen Reports

User Defined Reports

SQL Worksheet: History

Worksheet

Query Builder

```
v_ape := sapellido;
v_nom := snombre;
v_mail := semail;
v_hd := shire_date;
v_job := sjobid;
v_sueldo := ssalario;
v_manag := sid_manager;
v_dep := sid_departamento;

select count(department_id) into cuenta from employees where employee_id = v_id;
select count(manager_id) into cuenta2 from employees where employee_id = v_id;
select count(employee_id) into cuenta3 from employees where employee_id = v_id;

if cuenta = 0 then
    raise no_dep;
end if;
if cuenta2 = 0 then
    raise no_manag;
end if;
if cuenta3 >= 1 then
    raise empnum_dup;
end if;
if v_sueldo is null then
    raise_application_error(-27344, 'Salario invalido');
end if;

insert into employees (employee_id, first_name, last_name, email, hire_date, job_id, salary, manager_id, department_id)
values (v_id, v_nom, v_ape, v_mail, v_hd, v_job, v_sueldo, v_manag, v_dep);

DBMS_OUTPUT.PUT_LINE('El empleado con id='||v_id||' se llama '||v_nom||'
'||v_ape||' y cobra '||v_sueldo);

EXCEPTION
    when no_dep then
        DBMS_OUTPUT.PUT_LINE('No existe departamento');

    when no_manag then
        DBMS_OUTPUT.PUT_LINE('No existe manager');
```

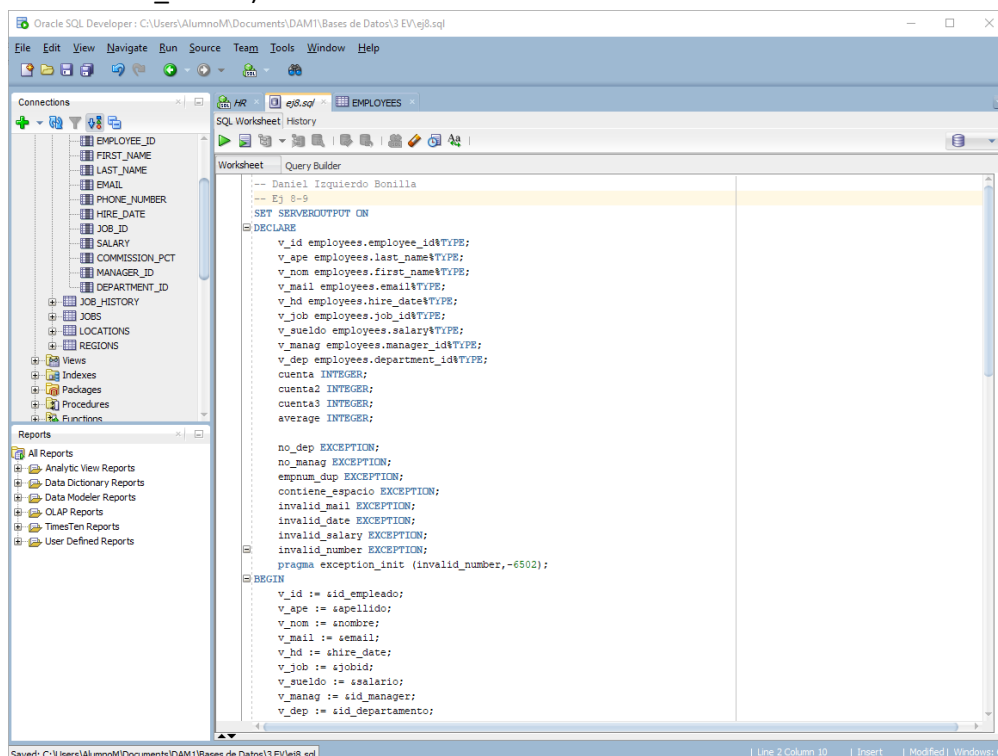
Click on an identifier with the Control key down to perform "Go to Declaration" | Line 60 Column 58 | Insert | Modified | Windows: C

Nombre completo del alumno: Daniel Izquierdo Bonilla

9. Ejercicio de EXCEPCIONES:

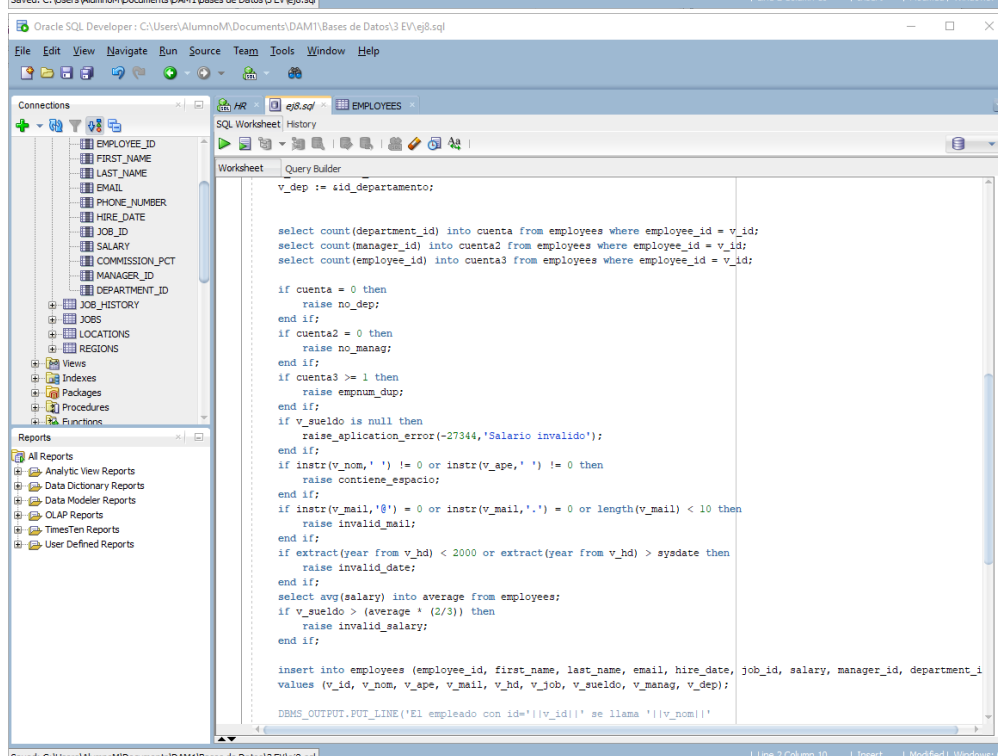
Añadir al bloque anónimo del ejercicio 8 las siguientes características:

- Nueva excepción de “nombre o apellido inválido” si lo que se introduce contiene espacios
- Nueva excepción de “email inválido” si el email que se introduce no contiene la @, el punto y al menos 10 caracteres de largo.
- Nueva excepción de “fecha de contratación inválida si es anterior al año 2000 o se trata de una fecha futura
- Nueva excepción de “salario excesivo” si lo que se introduce es mayor que los dos tercios de la media de todos los empleados de la compañía
- (Tratad de usar en algún momento todos los tipos excepciones, incluyendo PRAGMA EXCEPTION_INIT y RAISE_APPLICATION_ERROR)



```
-- Daniel Izquierdo Bonilla
-- Ej 9-9
SET SERVEROUTPUT ON
DECLARE
  v_id employees.employee_id%TYPE;
  v_ape employees.last_name%TYPE;
  v_nom employees.first_name%TYPE;
  v_mail employees.email%TYPE;
  v_hd employees.hire_date%TYPE;
  v_job employees.job_id%TYPE;
  v_sueldo employees.salary%TYPE;
  v_manag employees.manager_id%TYPE;
  v_dep employees.department_id%TYPE;
  cuenta INTEGER;
  cuenta2 INTEGER;
  cuenta3 INTEGER;
  average INTEGER;

  no_dep EXCEPTION;
  no_manag EXCEPTION;
  empnum_dup EXCEPTION;
  contiene_espacio EXCEPTION;
  invalid_mail EXCEPTION;
  invalid_date EXCEPTION;
  invalid_salary EXCEPTION;
  invalid_number EXCEPTION;
  pragma exception_init (invalid_number, -6502);
BEGIN
  v_id := sid_empleado;
  v_ape := apellido;
  v_nom := nombre;
  v_mail := email;
  v_hd := shire_date;
  v_job := sjobid;
  v_sueldo := ssalario;
  v_manag := sid_manag;
  v_dep := sid_departamento;
```



```
v_dep := sid_departamento;

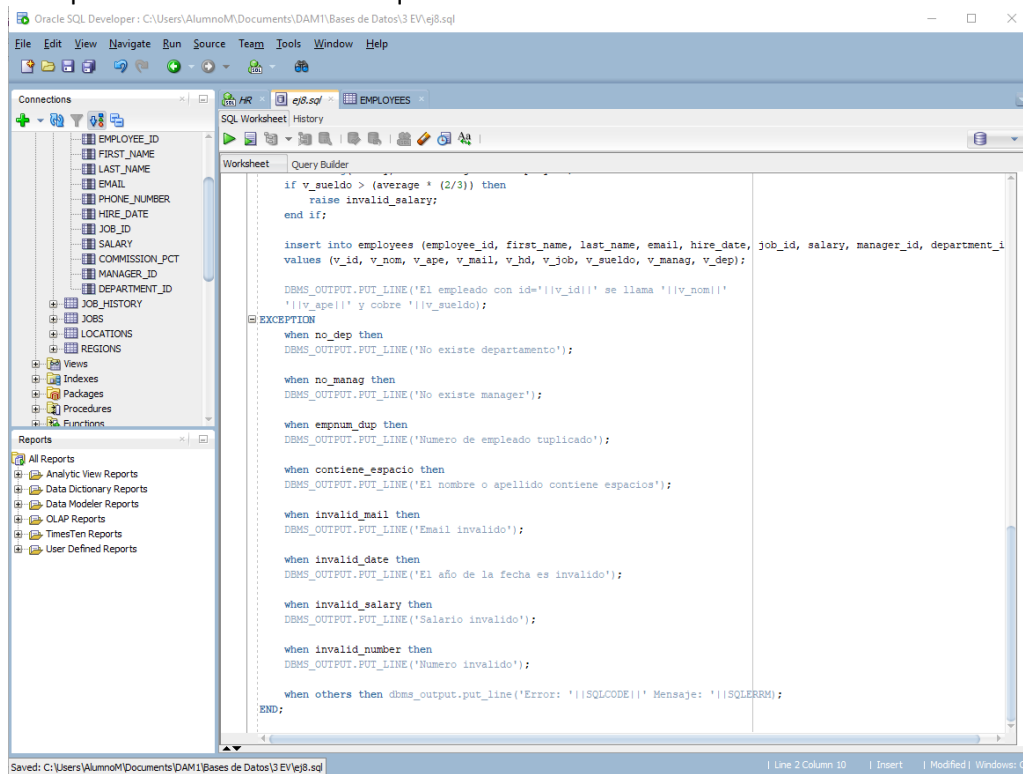
select count(department_id) into cuenta from employees where employee_id = v_id;
select count(manager_id) into cuenta2 from employees where employee_id = v_id;
select count(employee_id) into cuenta3 from employees where employee_id = v_id;

if cuenta = 0 then
  raise no_dep;
end if;
if cuenta2 = 0 then
  raise no_manag;
end if;
if cuenta3 >= 1 then
  raise empnum_dup;
end if;
if v_sueldo is null then
  raise_application_error(-27344, 'Salario invalido');
end if;
if instr(v_nom, ' ') != 0 or instr(v_ape, ' ') != 0 then
  raise contiene_espacio;
end if;
if instr(v_mail, '@') = 0 or instr(v_mail, '.') = 0 or length(v_mail) < 10 then
  raise invalid_mail;
end if;
if extract(year from v_hd) < 2000 or extract(year from v_hd) > sysdate then
  raise invalid_date;
end if;
select avg(salary) into average from employees;
if v_sueldo > (average * (2/3)) then
  raise invalid_salary;
end if;

insert into employees (employee_id, first_name, last_name, email, hire_date, job_id, salary, manager_id, department_id)
values (v_id, v_nom, v_ape, v_mail, v_hd, v_job, v_sueldo, v_manag, v_dep);

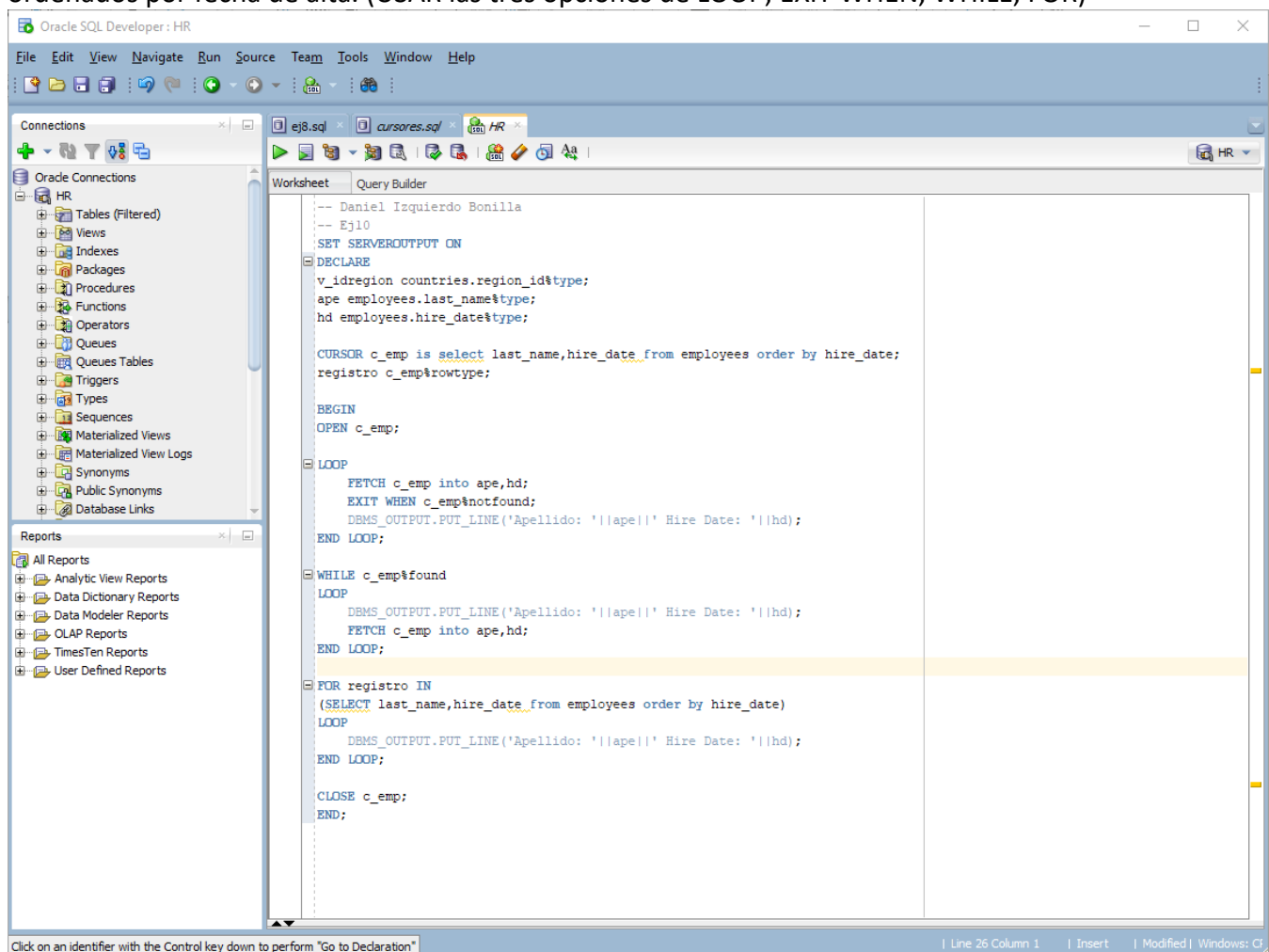
DBMS_OUTPUT.PUT_LINE('El empleado con id'||v_id||' se llama '||v_nom||'
```

Nombre completo del alumno: Daniel Izquierdo Bonilla



EJERCICIOS DE CURSORES

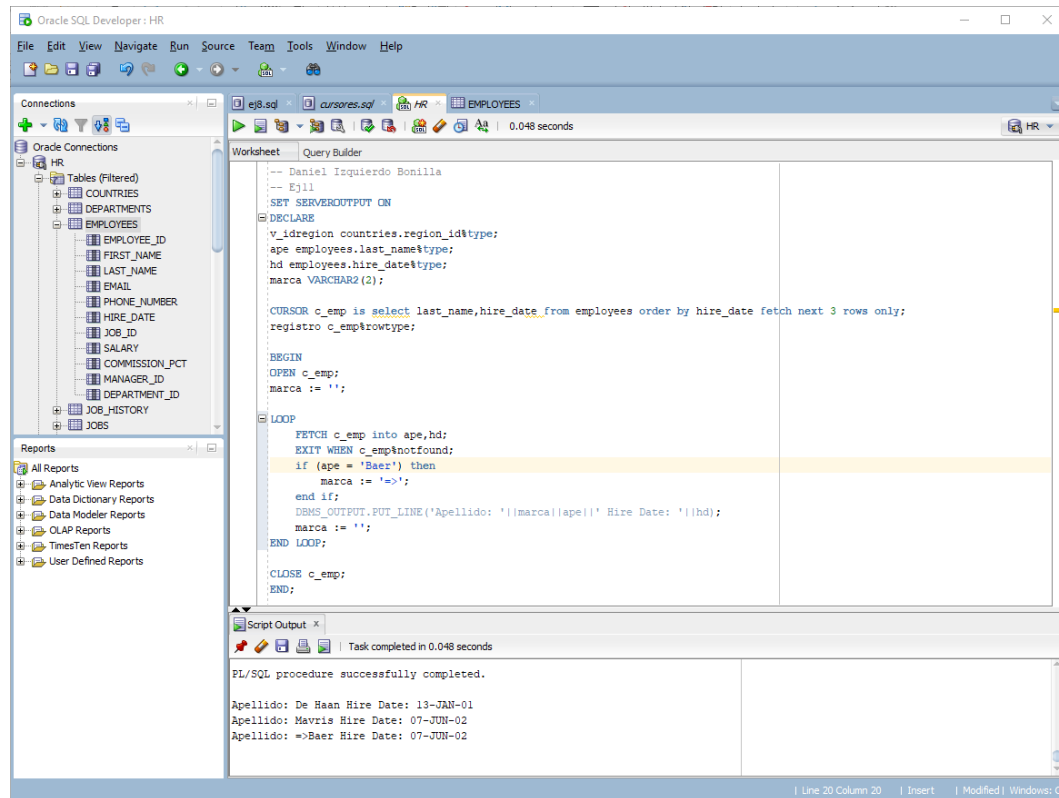
10. Escribiremos un bloque PL/SQL que visualice el apellido y la fecha de alta de todos los empleados ordenados por fecha de alta. (USAR las tres opciones de LOOP; EXIT WHEN, WHILE, FOR)



Nombre completo del alumno: Daniel Izquierdo Bonilla

11. En el ejercicio anterior,

- ponerle una marca (por ejemplo →) a un empleado que tenga un determinado apellido o departamento.
- listar solo los tres primeros



```
-- Daniel Izquierdo Bonilla
-- Ej11
SET SERVEROUTPUT ON
DECLARE
  v_idregion countries.region_id%type;
  ape employees.last_name%type;
  hd employees.hire_date%type;
  marca VARCHAR2(2);

  CURSOR c_emp is select last_name,hire_date from employees order by hire_date fetch next 3 rows only;
  registro c_emp%rowtype;

BEGIN
  OPEN c_emp;
  marca := '';

  LOOP
    FETCH c_emp into ape,hd;
    EXIT WHEN c_emp%notfound;
    if (ape = 'Baer') then
      marca := '=>';
    end if;
    DBMS_OUTPUT.PUT_LINE('Apellido: '||marca||ape||' Hire Date: '||hd);
    marca := '';
  END LOOP;

  CLOSE c_emp;
END;
```

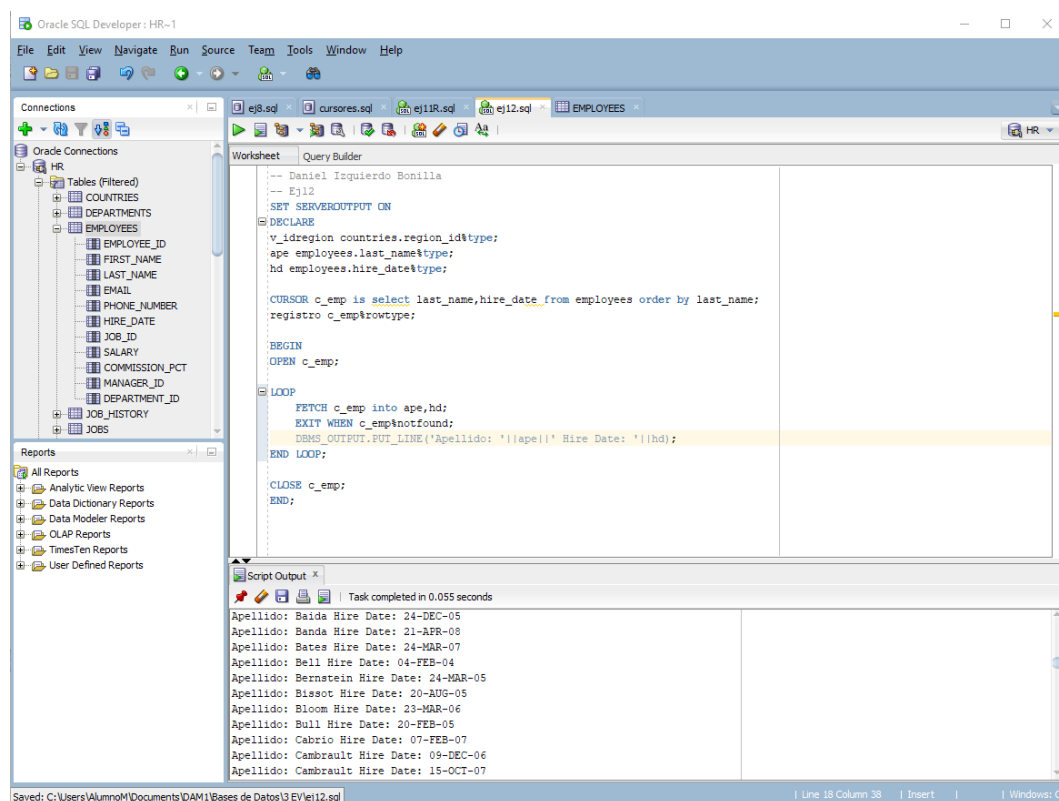
Script Output

Task completed in 0.048 seconds

PL/SQL procedure successfully completed.

Apellido: De Haan Hire Date: 13-JAN-01
Apellido: Mavris Hire Date: 07-JUN-02
Apellido: =>Baer Hire Date: 07-JUN-02

12. Desarrollar un programa que visualice el apellido y la fecha de alta de todos los empleados ordenados por apellido.



```
-- Daniel Izquierdo Bonilla
-- Ej12
SET SERVEROUTPUT ON
DECLARE
  v_idregion countries.region_id%type;
  ape employees.last_name%type;
  hd employees.hire_date%type;

  CURSOR c_emp is select last_name,hire_date from employees order by last_name;
  registro c_emp%rowtype;

BEGIN
  OPEN c_emp;

  LOOP
    FETCH c_emp into ape,hd;
    EXIT WHEN c_emp%notfound;
    DBMS_OUTPUT.PUT_LINE('Apellido: '||ape||' Hire Date: '||hd);
  END LOOP;

  CLOSE c_emp;
END;
```

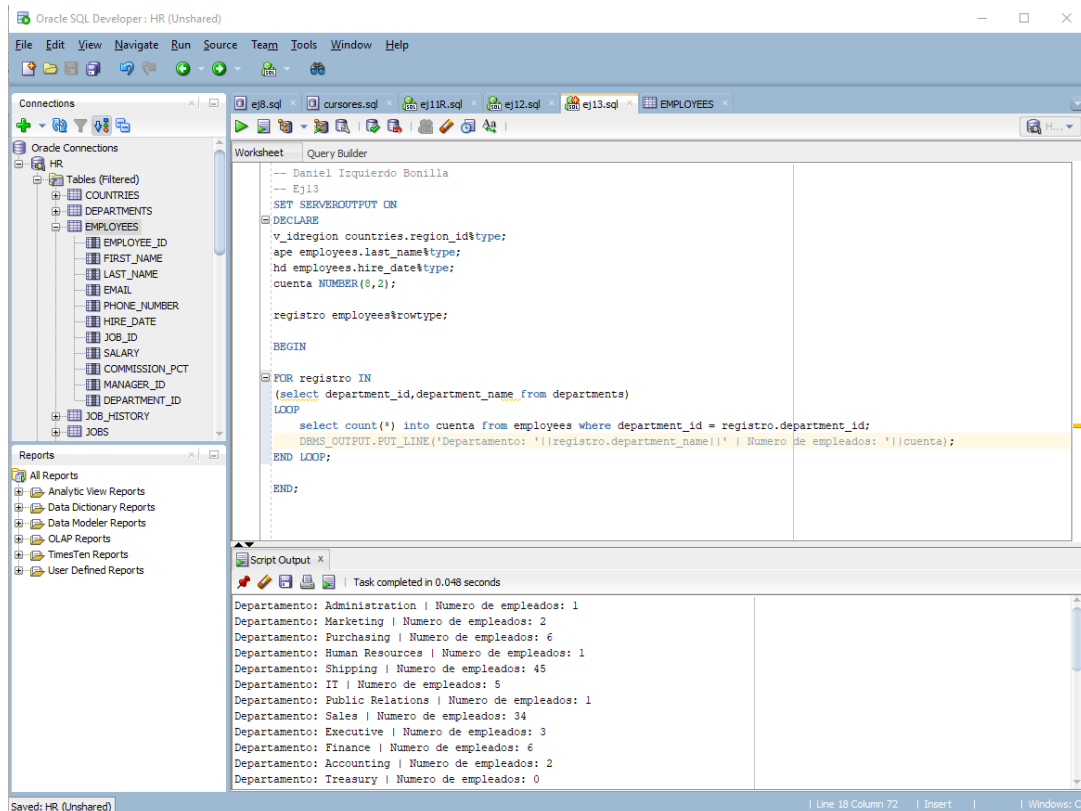
Script Output

Task completed in 0.055 seconds

Apellido: Baida Hire Date: 24-DEC-05
Apellido: Banda Hire Date: 21-APR-08
Apellido: Bates Hire Date: 24-MAR-07
Apellido: Bell Hire Date: 04-FEB-04
Apellido: Bernstein Hire Date: 24-MAR-05
Apellido: Bisot Hire Date: 20-AUG-05
Apellido: Bloom Hire Date: 23-MAR-06
Apellido: Bull Hire Date: 20-FEB-05
Apellido: Cabrio Hire Date: 07-FEB-07
Apellido: Cambrault Hire Date: 09-DEC-06
Apellido: Cambrault Hire Date: 15-OCT-07

Nombre completo del alumno: Daniel Izquierdo Bonilla

13. Codificar un programa que muestre el nombre de cada departamento y el número de empleados que tiene. (USAR FOR...LOOP)y que muestre también departamentos sin empleados (0 empleados)



```
-- Daniel Izquierdo Bonilla
-- Ej13
SET SERVEROUTPUT ON

DECLARE
  v_idregion countries.region_id%type;
  ape employees.last_name%type;
  hd employees.hire_date%type;
  cuenta NUMBER(8,2);

  registro employees%rowtype;

BEGIN

  FOR registro IN
    (select department_id, department_name from departments)
  LOOP
    select count(*) into cuenta from employees where department_id = registro.department_id;
    DBMS_OUTPUT.PUT_LINE('Departamento: '||registro.department_name||' | Numero de empleados: '||cuenta);
  END LOOP;

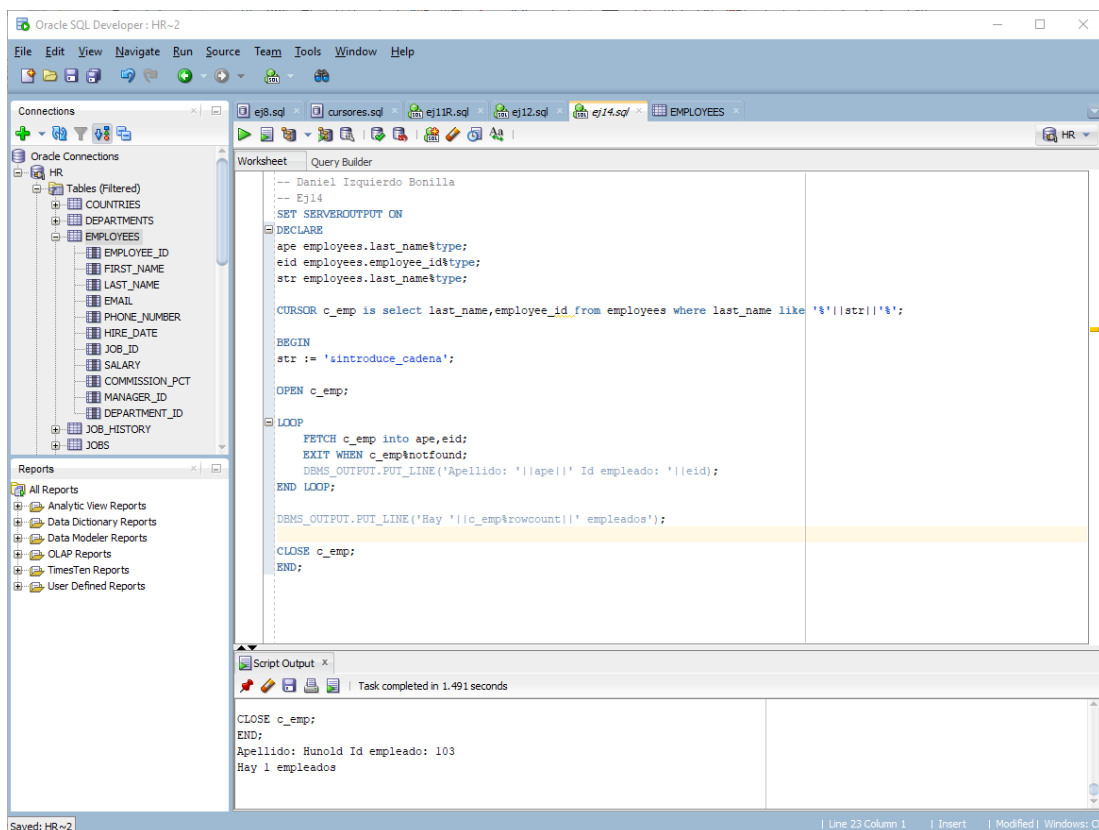
END;
```

Script Output x

Task completed in 0.048 seconds

| | |
|--------------------------------|-------------------------|
| Departamento: Administration | Numero de empleados: 1 |
| Departamento: Marketing | Numero de empleados: 2 |
| Departamento: Purchasing | Numero de empleados: 6 |
| Departamento: Human Resources | Numero de empleados: 1 |
| Departamento: Shipping | Numero de empleados: 45 |
| Departamento: IT | Numero de empleados: 5 |
| Departamento: Public Relations | Numero de empleados: 1 |
| Departamento: Sales | Numero de empleados: 34 |
| Departamento: Executive | Numero de empleados: 3 |
| Departamento: Finance | Numero de empleados: 6 |
| Departamento: Accounting | Numero de empleados: 2 |
| Departamento: Treasury | Numero de empleados: 0 |

14. Escribir un programa que reciba una cadena y visualice el apellido y el número de empleado de todos los empleados cuyo apellido contenga la cadena especificada. Al finalizar visualizar el número de empleados mostrados. (Con OPEN, FETCH, CLOSE...)



```
-- Daniel Izquierdo Bonilla
-- Ej14
SET SERVEROUTPUT ON

DECLARE
  ape employees.last_name%type;
  eid employees.employee_id%type;
  str employees.last_name%type;

  CURSOR c_emp is select last_name, employee_id from employees where last_name like '%'||str||'%';

BEGIN
  str := 'sintroduce_cadena';

  OPEN c_emp;

  LOOP
    FETCH c_emp into ape, eid;
    EXIT WHEN c_emp%notfound;
    DBMS_OUTPUT.PUT_LINE('Apellido: '||ape||' Id empleado: '||eid);
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('Hay '||c_emp%rowcount||' empleados');

  CLOSE c_emp;
END;
```

Script Output x

Task completed in 1.491 seconds

```
CLOSE c_emp;
END;
Apellido: Humold Id empleado: 103
Hay 1 empleados
```

Nombre completo del alumno: Daniel Izquierdo Bonilla

15. Escribir un programa que visualice el apellido y el salario de los cinco empleados que tienen el salario más alto. (Con OPEN, FETCH, CLOSE...)

The screenshot shows the Oracle SQL Developer interface. The 'Connections' pane on the left shows the 'HR' schema with tables like EMPLOYEES, DEPARTMENTS, and JOBS. The 'Worksheet' pane contains the following PL/SQL code:

```
-- Daniel Izquierdo Bonilla
-- Ej15
SET SERVEROUTPUT ON

DECLARE
ape employees.last_name%type;
salario employees.salary%type;

CURSOR c_emp is select last_name,salary from employees order by salary desc fetch next 5 rows only;

BEGIN

OPEN c_emp;

LOOP

    FETCH c_emp into ape,salario;
    EXIT WHEN c_emp%notfound;
    DBMS_OUTPUT.PUT_LINE('Apellido: '||ape||' Salario: '||salario);
END LOOP;

CLOSE c_emp;
END;
```

The 'Script Output' pane at the bottom shows the results of the execution:

```
Task completed in 0.057 seconds

Apellido: King Salario: 24000
Apellido: Kochhar Salario: 17000
Apellido: De Haan Salario: 17000
Apellido: Russell Salario: 14000
```

16. Codificar un programa que visualice los dos empleados que ganan menos de cada oficina

The screenshot shows the Oracle SQL Developer interface. The 'Connections' pane on the left shows the 'HR' schema with tables like EMPLOYEES, DEPARTMENTS, and JOBS. The 'Worksheet' pane contains the following PL/SQL code:

```
-- Daniel Izquierdo Bonilla
-- Ej16
SET SERVEROUTPUT ON

DECLARE
ape employees.last_name%type;
salario employees.salary%type;

CURSOR c_emp (jid jobs.job_id%type) is select last_name,salary from employees where job_id = jid order by salary desc fetch next 2 rows only;

BEGIN

FOR registro IN
(SELECT job_title, job_id from jobs)
LOOP

    OPEN c_emp(registro.job_id);
    DBMS_OUTPUT.PUT_LINE('Oficio: '||registro.job_title);

    FETCH c_emp into ape,salario;
    WHILE c_emp%notfound
    LOOP
        DBMS_OUTPUT.PUT_LINE('Apellido: '||ape||' Salario: '||salario);
        FETCH c_emp into ape,salario;
    END LOOP;
END LOOP;

END;
```

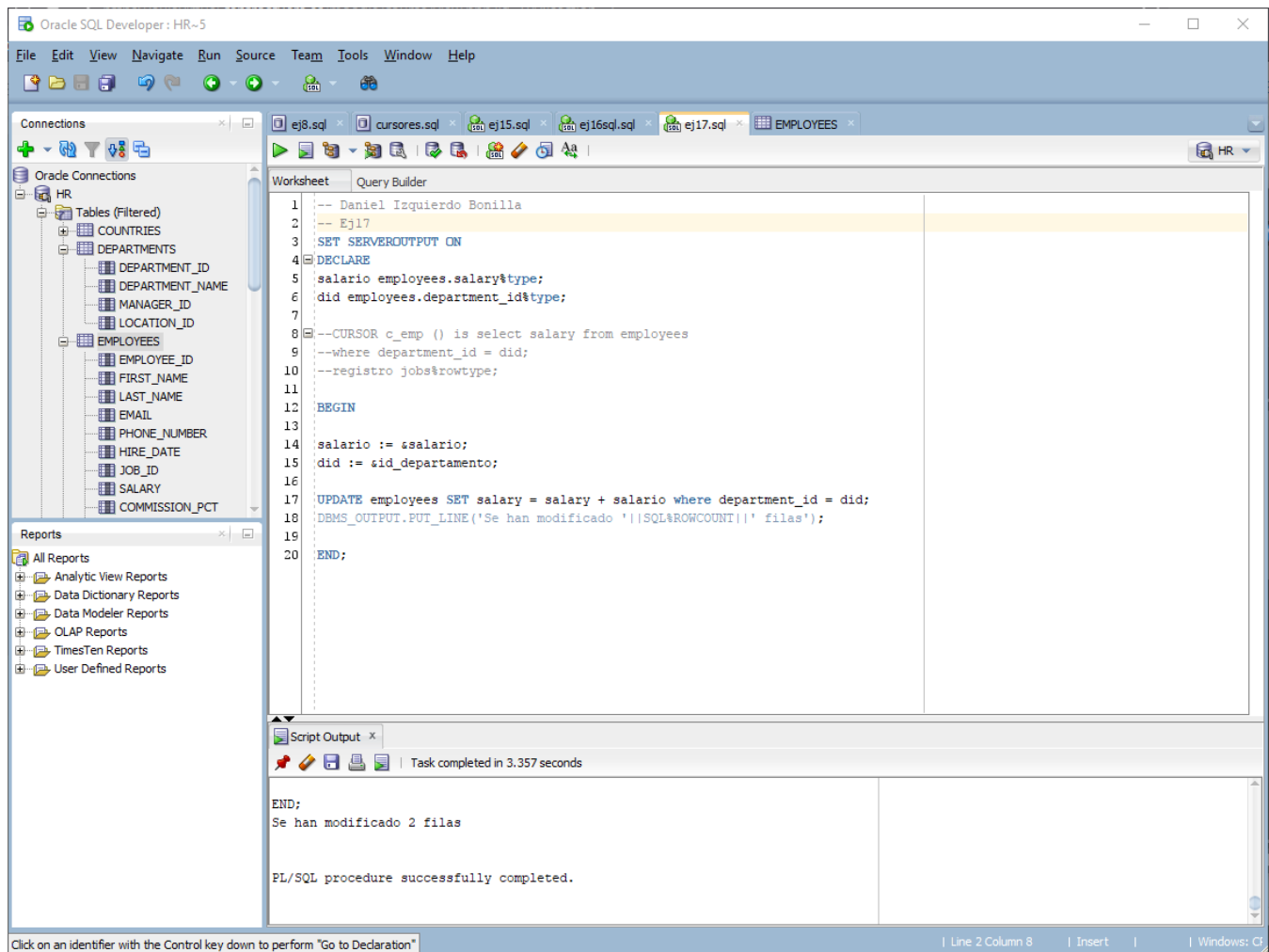
The 'Script Output' pane at the bottom shows the results of the execution:

```
Task completed in 0.058 seconds

Oficio: Stock Manager
Apellido: Fripp Salario: 8200
Apellido: Weiss Salario: 8000
Oficio: Stock Clerk
Apellido: Ludwig Salario: 3600
Apellido: Raju Salario: 3500
Oficio: Shipping Clerk
```

Nombre completo del alumno: Daniel Izquierdo Bonilla

17. Escribe un programa que incremente el salario de los empleados de un determinado departamento que se pasará como primer parámetro. El incremento será una cantidad en euros que se pasará como segundo parámetro en la llamada. El programa deberá informar del número de filas afectadas por la actualización.



```
1  -- Daniel Izquierdo Bonilla
2  -- Ej17
3  SET SERVEROUTPUT ON
4  DECLARE
5  salario employees.salary%type;
6  did employees.department_id%type;
7
8  --CURSOR c_emp () is select salary from employees
9  --where department_id = did;
10 --registro jobs$rowtype;
11
12 BEGIN
13
14 salario := &salario;
15 did := &sid_departamento;
16
17 UPDATE employees SET salary = salary + salario where department_id = did;
18 DBMS_OUTPUT.PUT_LINE('Se han modificado '||SQL%ROWCOUNT||' filas');
19
20 END;
```

Script Output

Task completed in 3.357 seconds

END;
Se han modificado 2 filas

PL/SQL procedure successfully completed.

Nombre completo del alumno: Daniel Izquierdo Bonilla

EJERCICIOS DE FUNCIONES Y PROCEDIMIENTOS (1ª PARTE)

Nombre completo del alumno: Daniel Izquierdo Bonilla

EJERCICIOS DE FUNCIONES Y PROCEDIMIENTOS (2ª PARTE)

Nombre completo del alumno: Daniel Izquierdo Bonilla

EJERCICIOS DE TRIGGERS