**Centro de Estudios Superiores AFUERA**

**Curso 2021-2022**

**BASES DE DATOS**

**Cuadernillo de prácticas de PL/SQL**

**1ª PARTE**

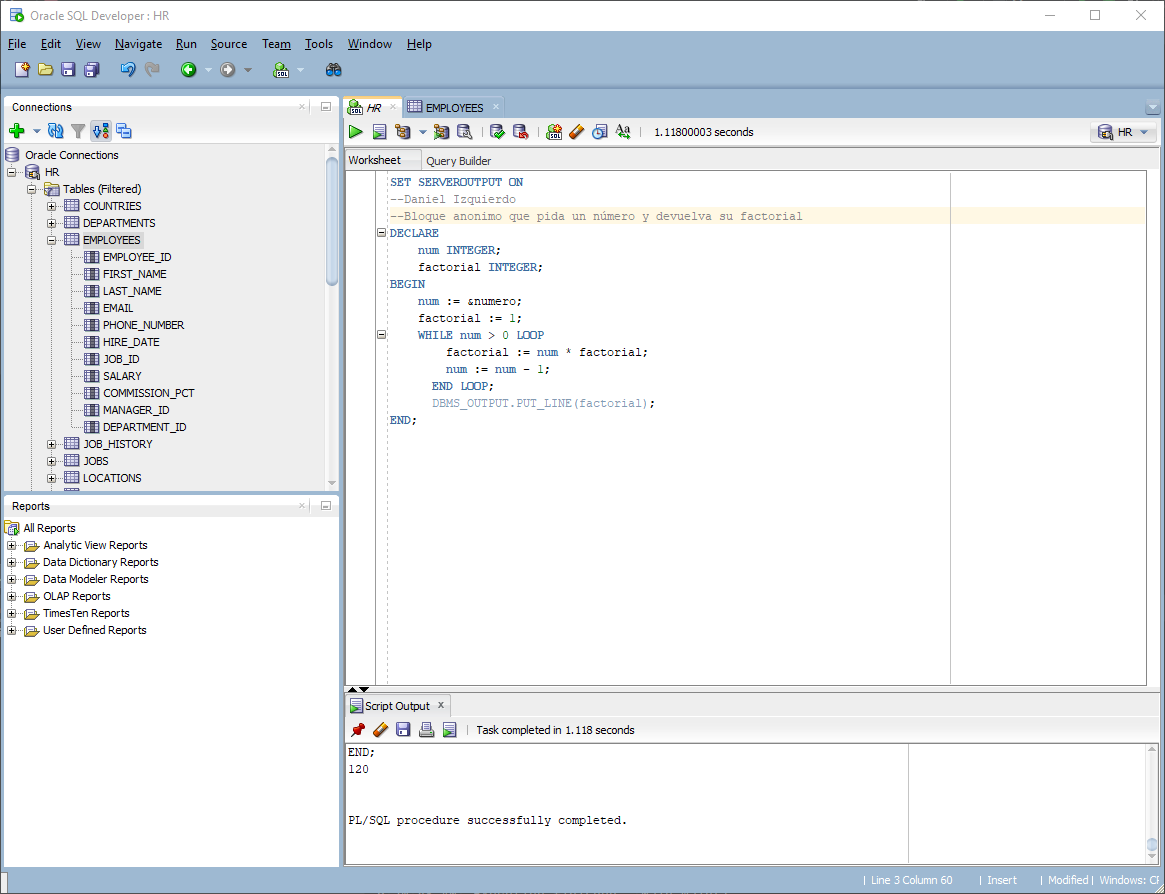
**Instrucciones:**

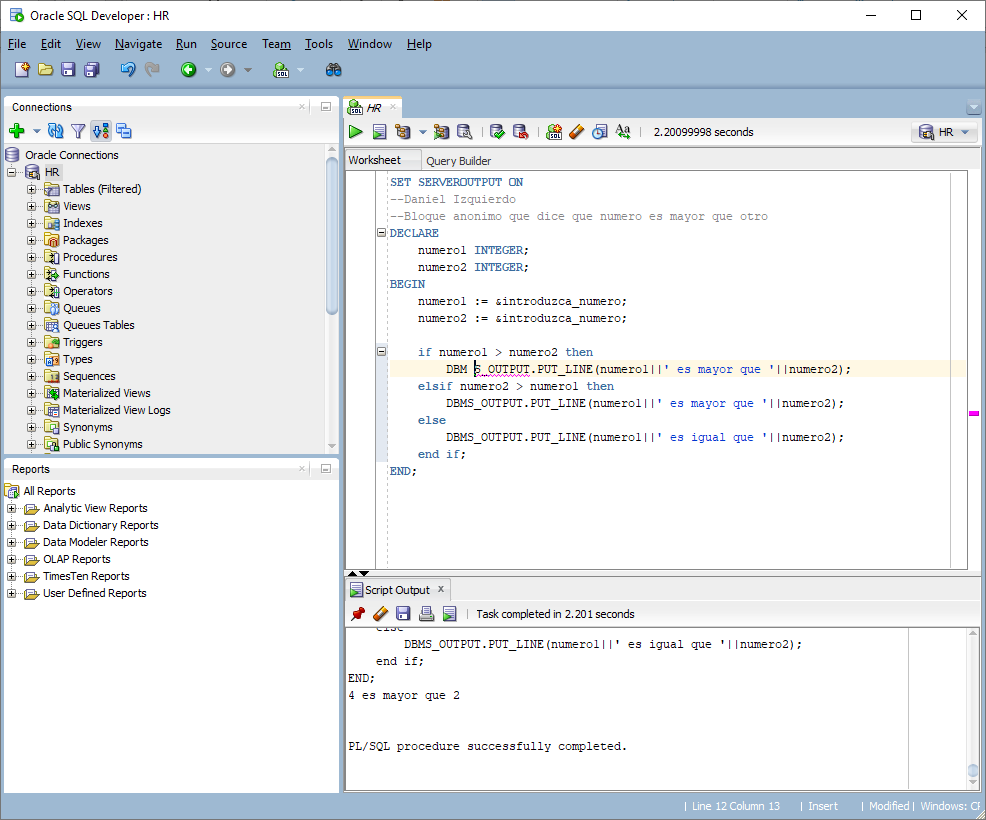
En este documento Word los alumnos irán realizando los ejercicios de Base de Datos y se usará como documento para las tareas y entregas parciales según se vayan completando cada uno de los temas. El documento se salvará con el nombre PLSQL\_1erApellido\_Nombre\_A (formato pdf o docx u odt)

NOTA: Los 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.

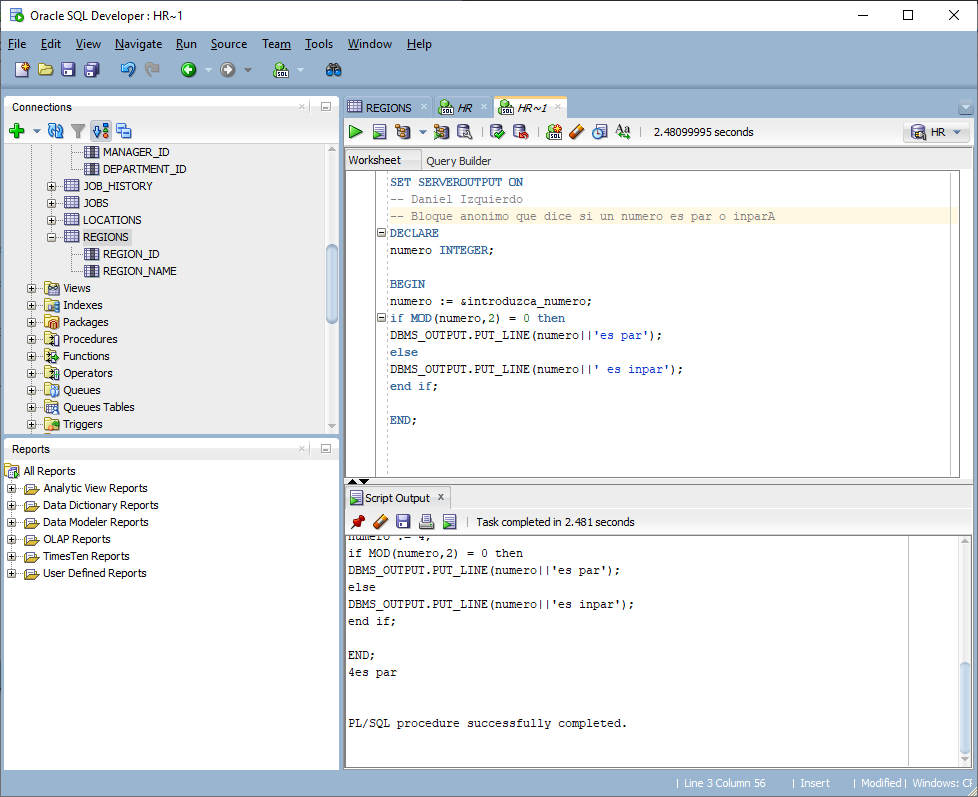
**EJERCICIOS DE BLOQUES ANÓNIMOS**

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

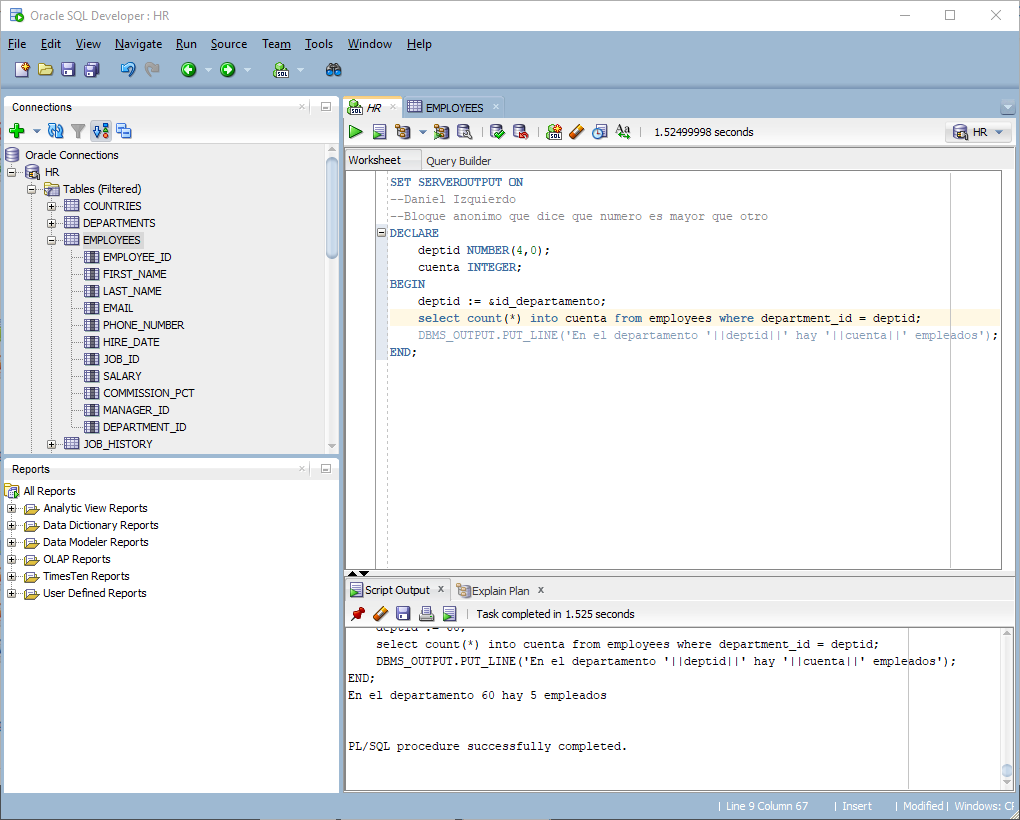
2. Hacer un bloque anónimo que pida dos números y cual es mayor que el otro



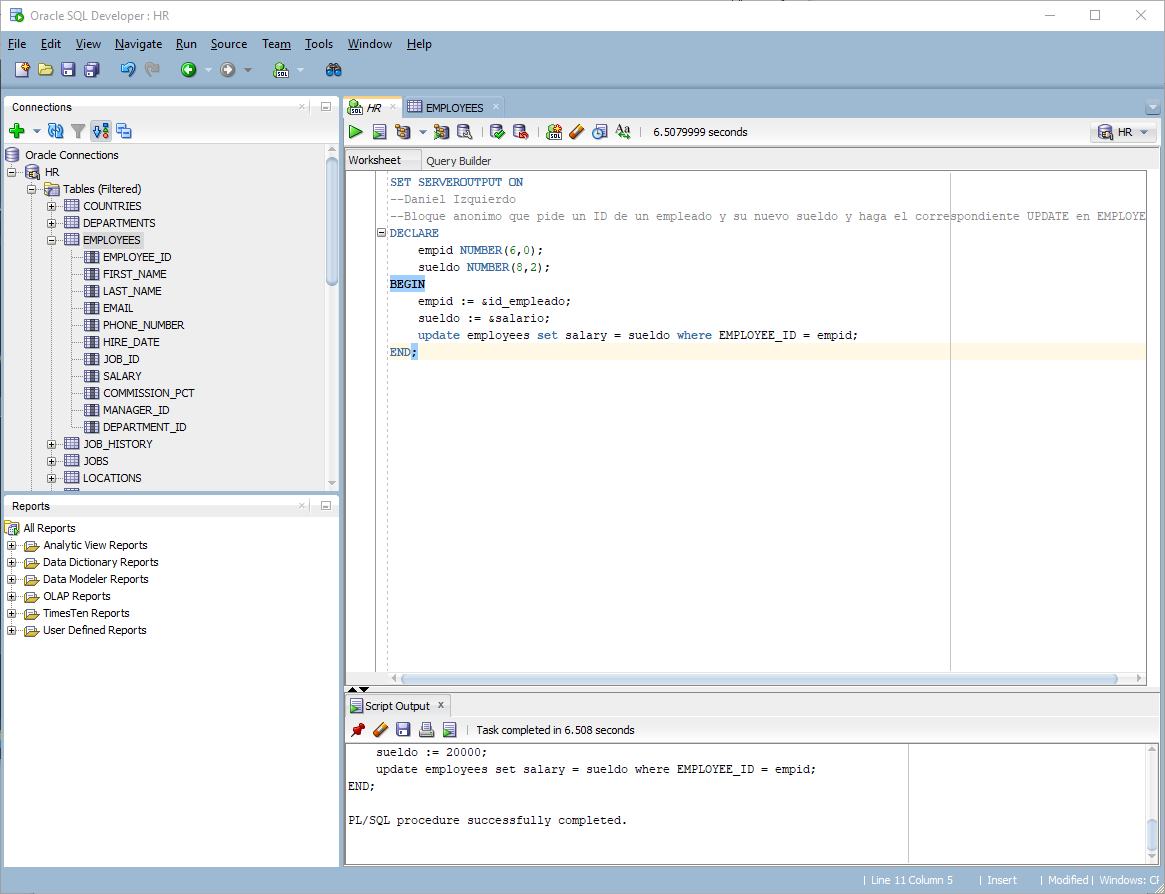
3. Hacer un bloque anónimo que pida un numero entero y diga si es par o impar



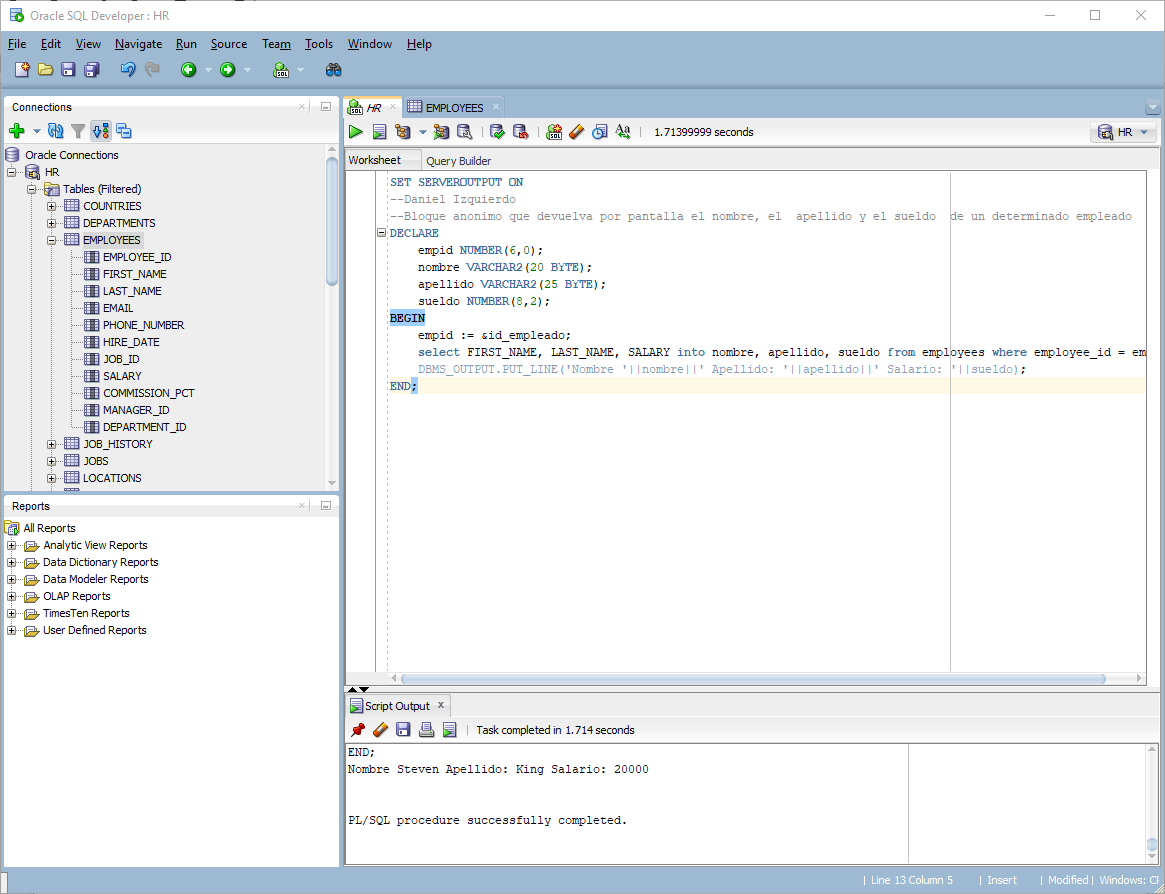
4. Hacer un bloque anónimo que cuente los empleados de un determinado departamento (del cual se pasa su ID)



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



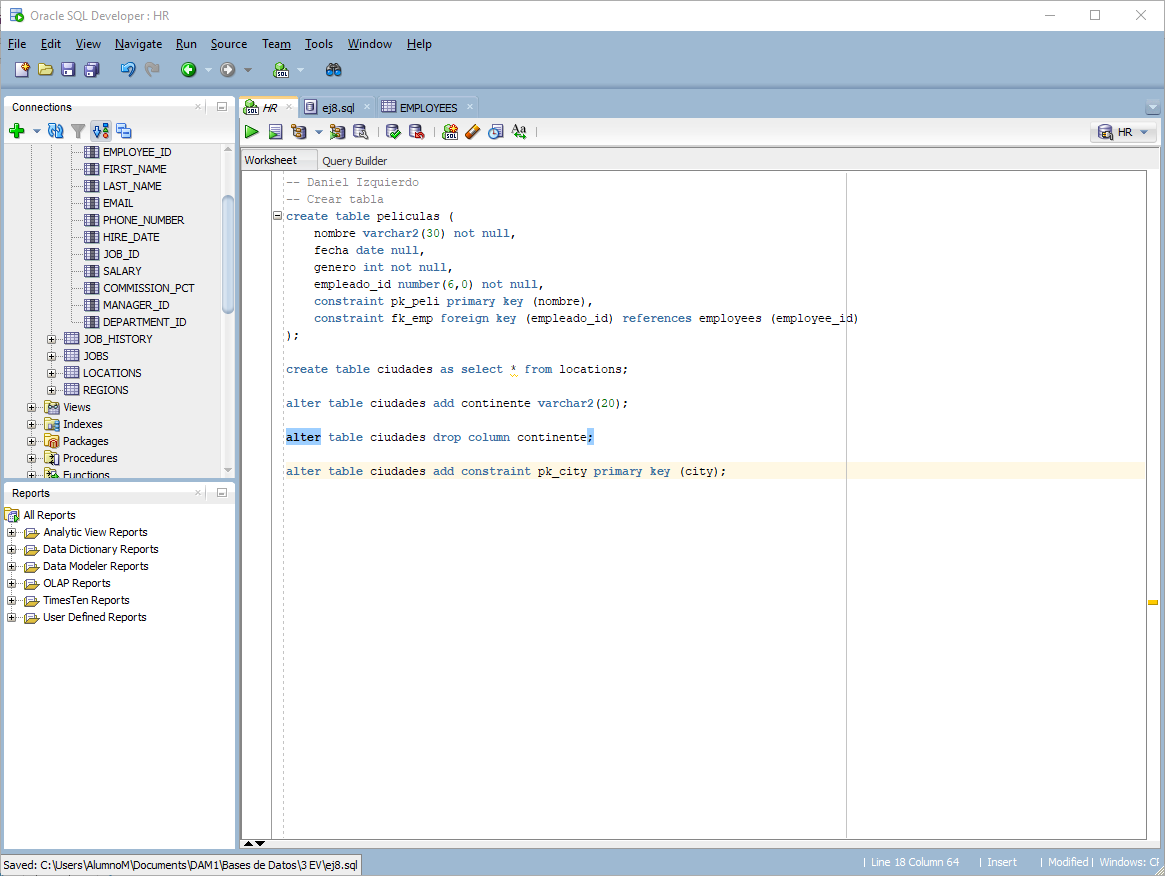
6. Hacer un bloque anónimo que devuelva por pantalla el nombre, el apellido y el sueldo de un determinado empleado del cual se pasa su ID



**EJERCICIOS DE EXCEPCIONES**

7. Ejercicio repaso de DDL:

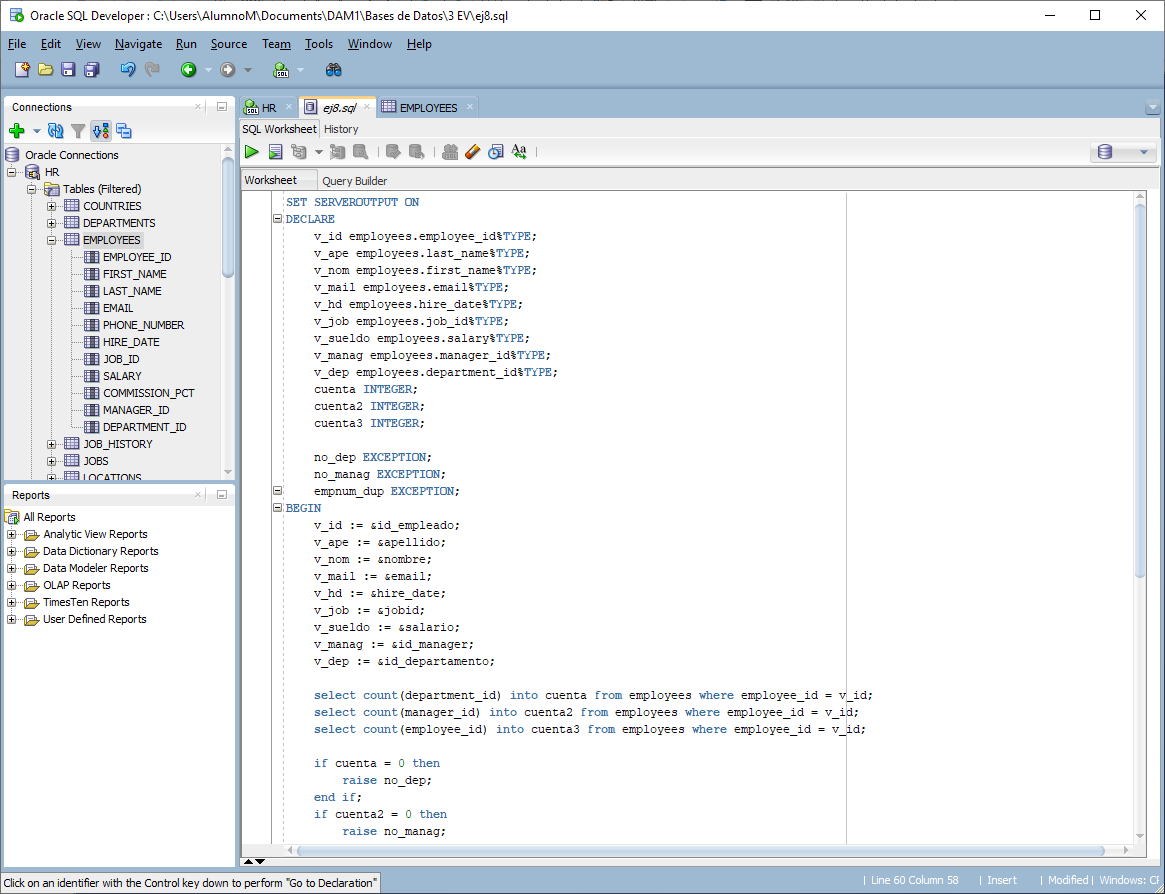
* CREAR UNA TABLA
* CREAR UNA TABLA USANDO UNA SUBCONSULTA
* AÑADIR (O BORRAR) UNA COLUMNA (sobre una de las tablas que acabamos de crear)
* AÑADIR INDICES, CLAVES FORÁNEAS ETC (sobre una de las tablas que acabamos de crear)

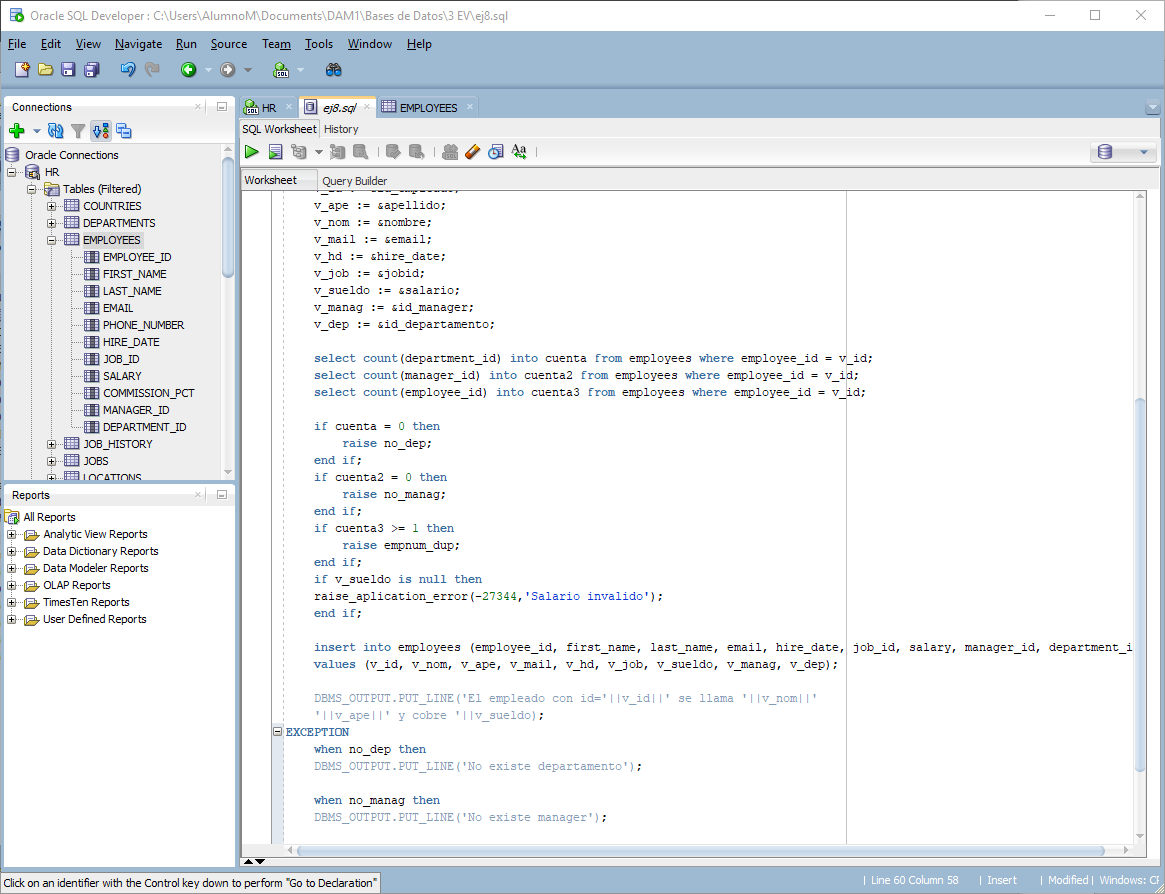


8. Ejercicio de EXCEPCIONES:

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.

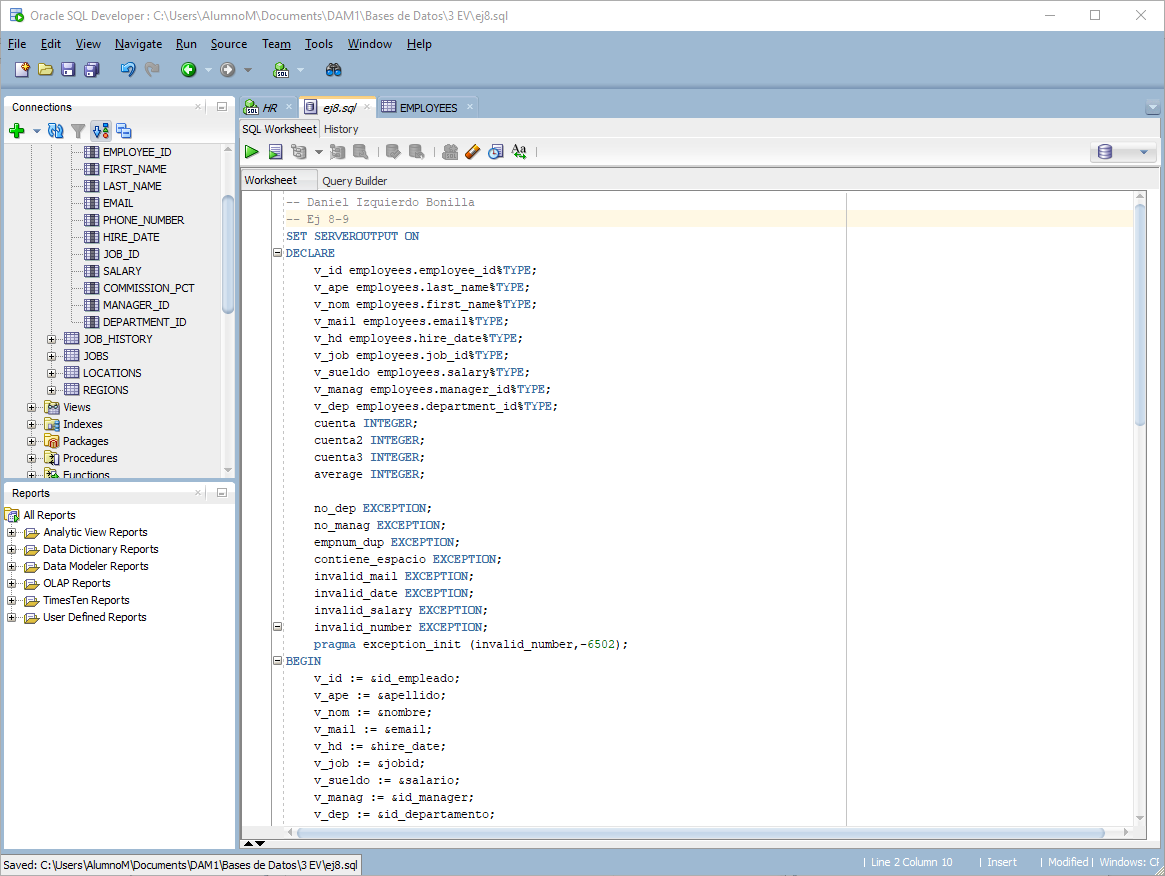


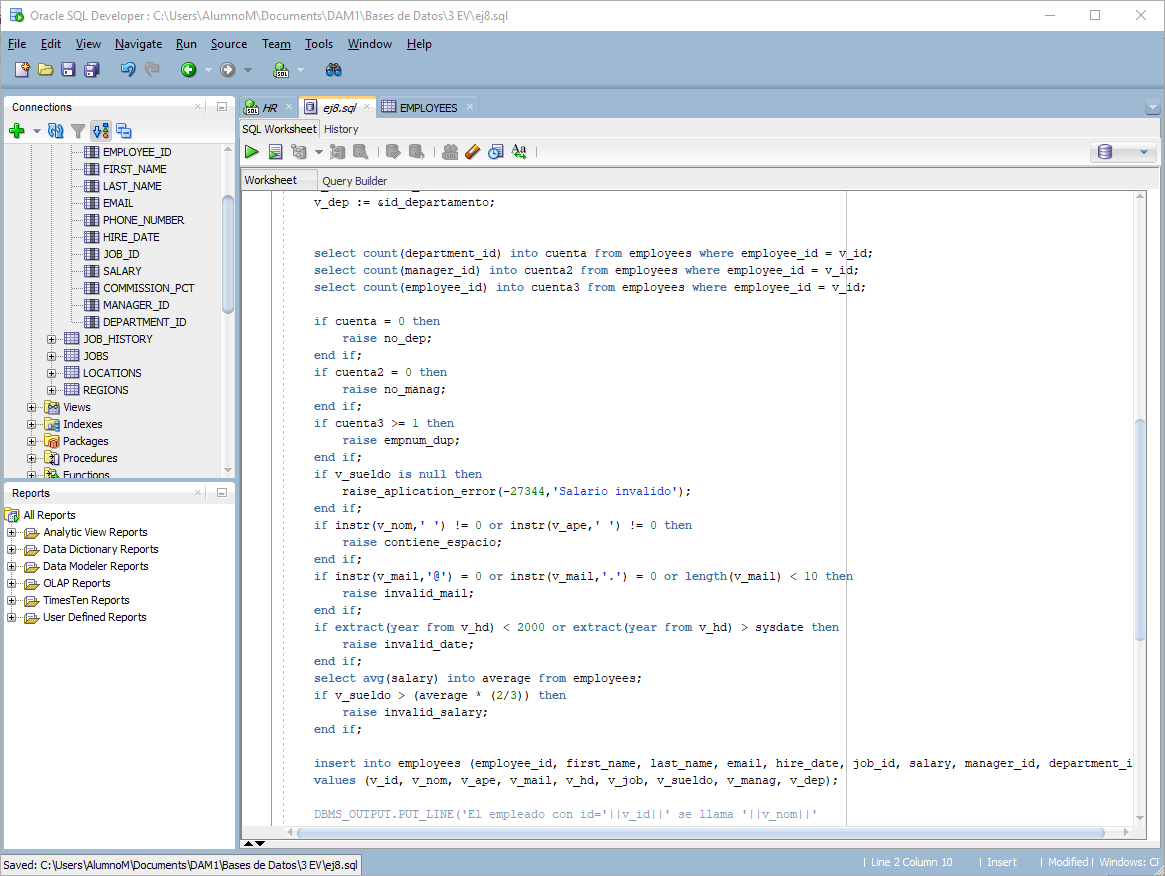


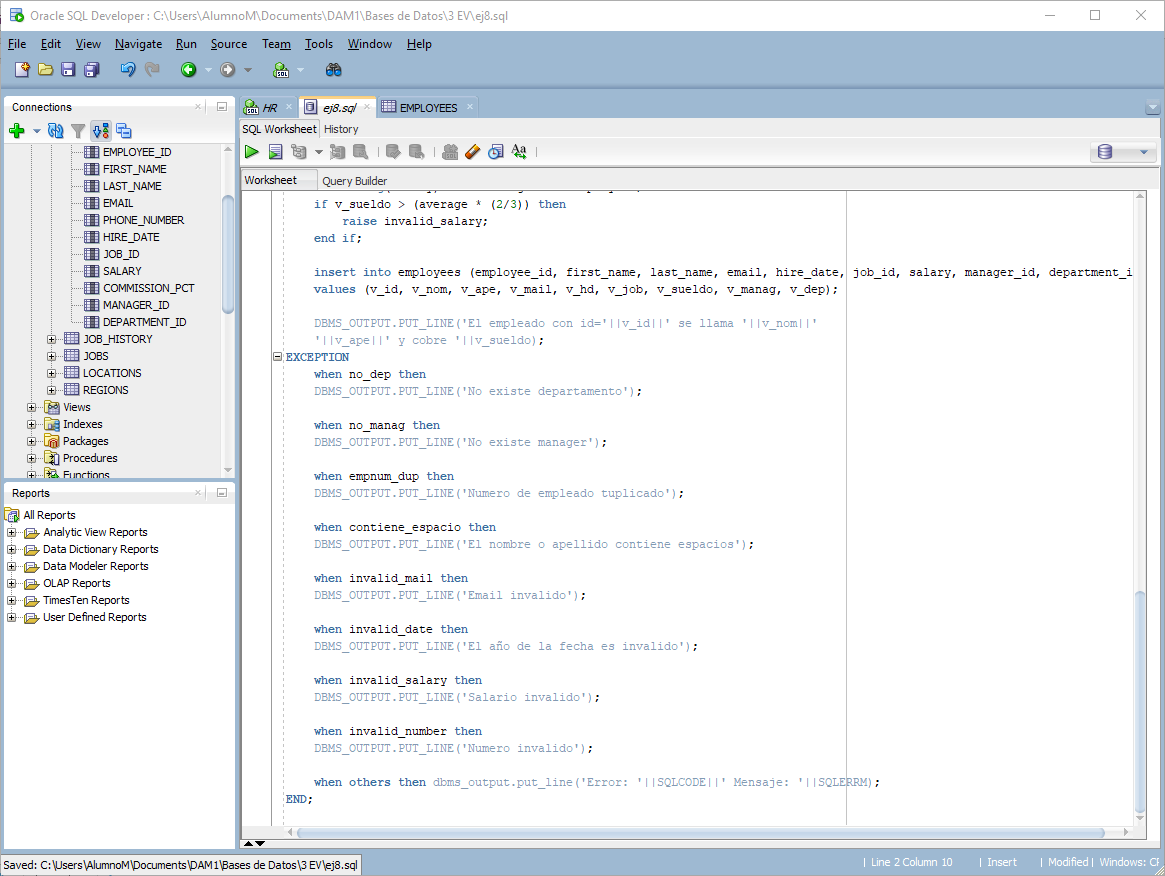
9. Ejercicio de EXCEPCIONES:

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)

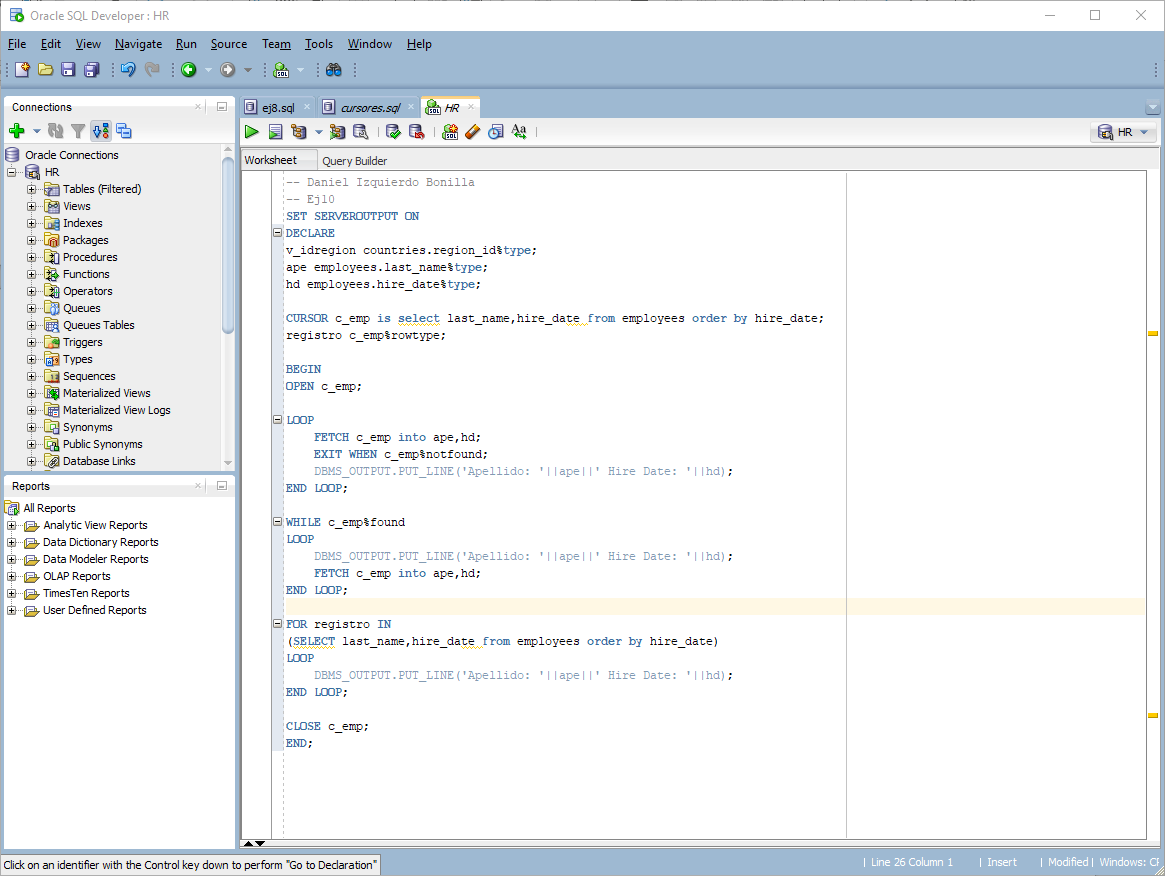






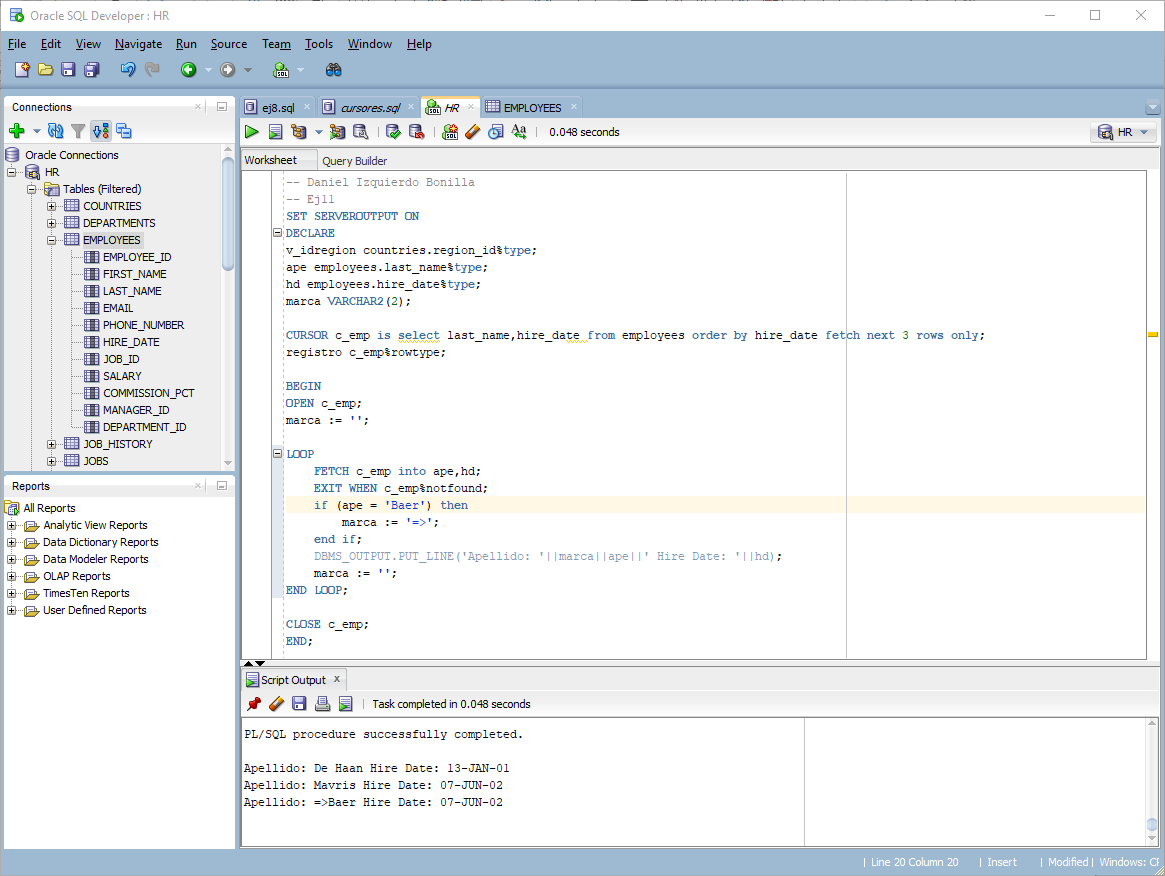
**EJERCICIOS DE CURSORES**

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)

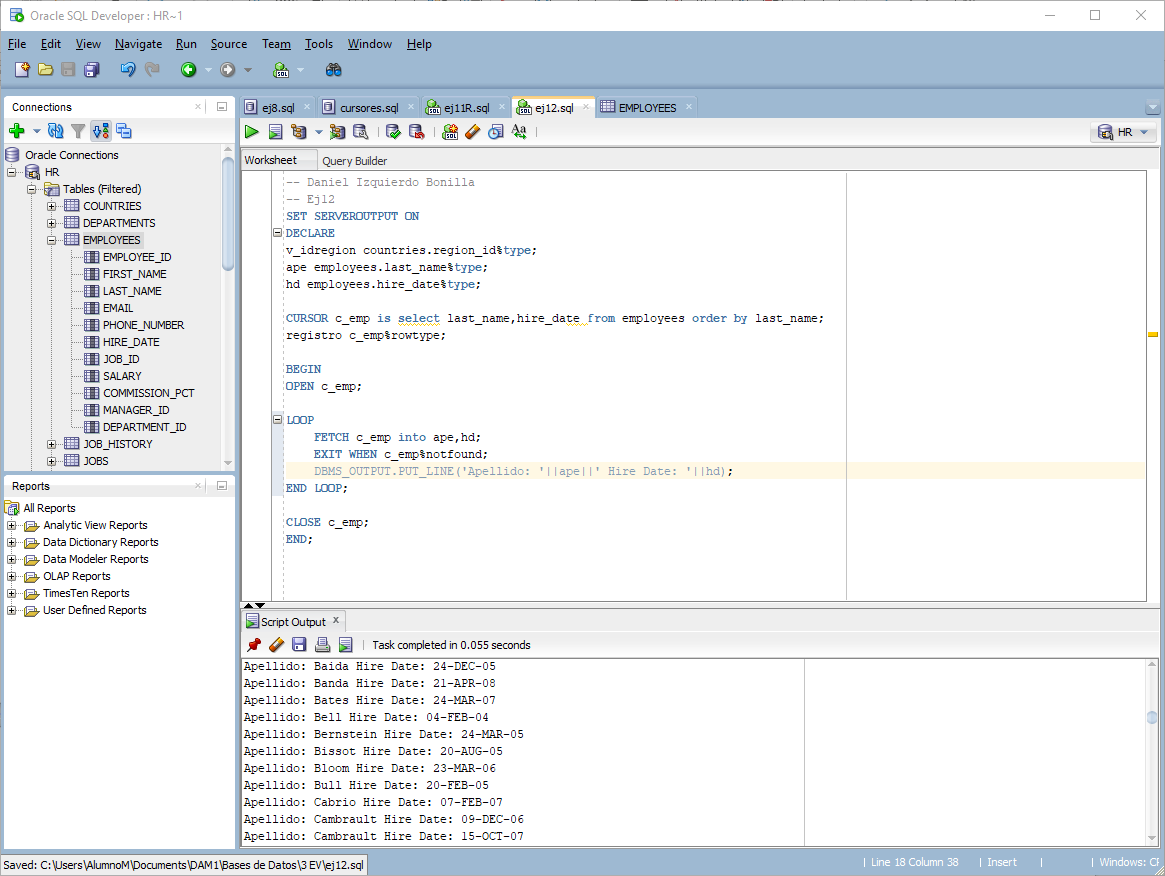


11. En el ejercicio anterior,

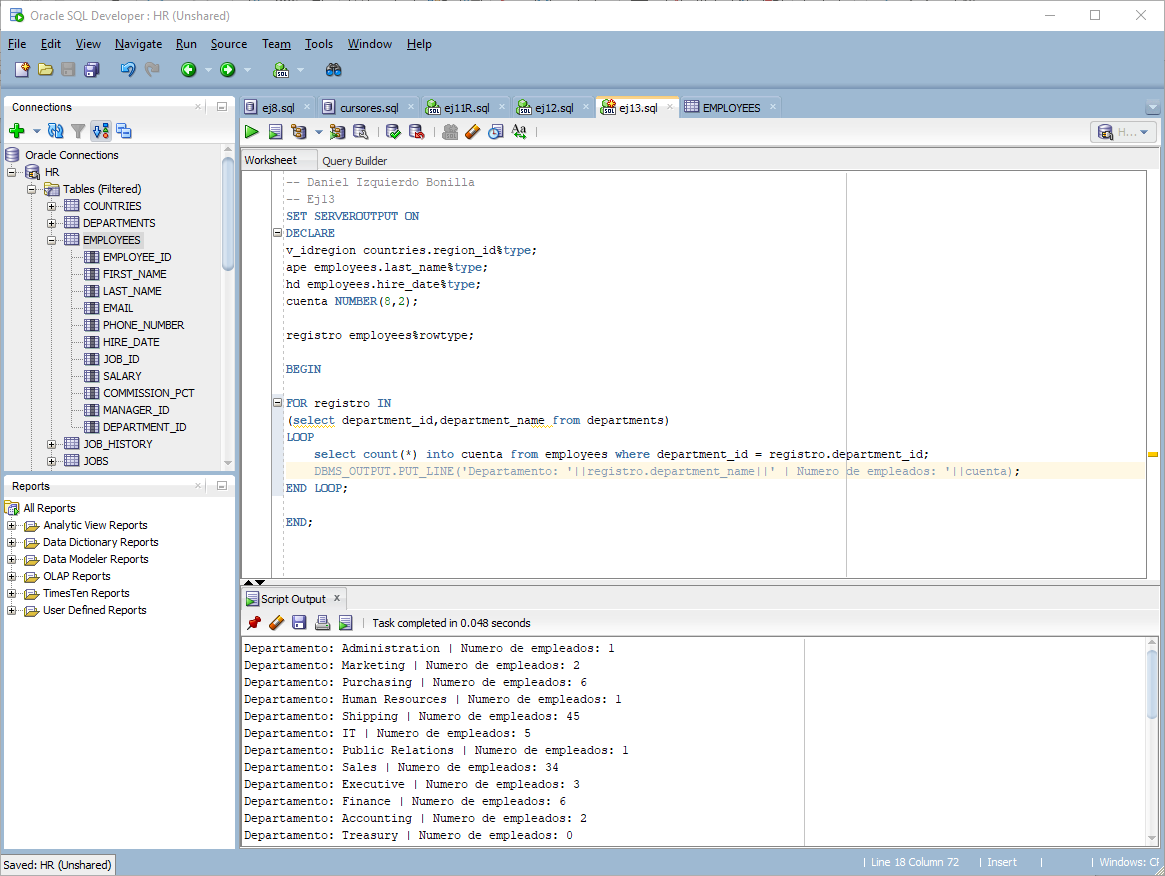
* ponerle una marca (por ejemplo ) a un empleado que tenga un determinado apellido o departamento.
* listar solo los tres primeros



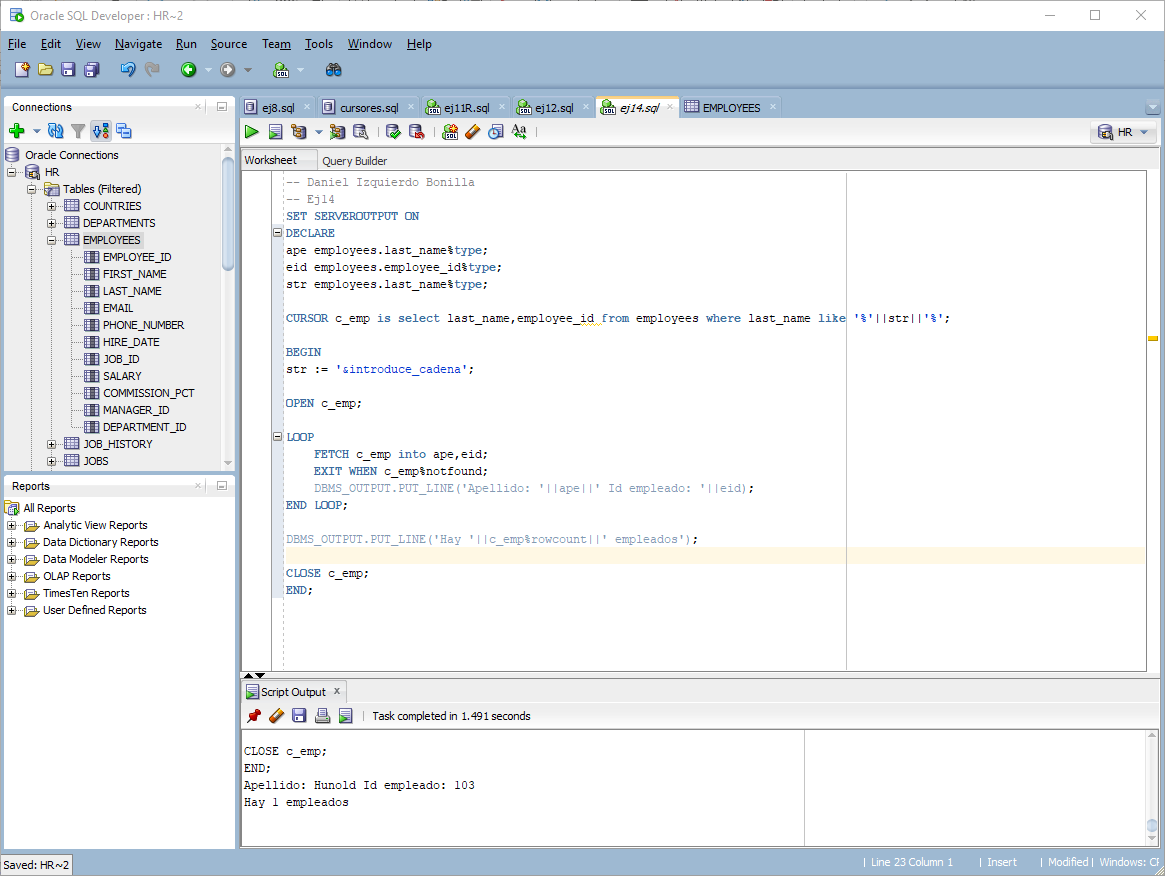
12. Desarrollar un programa que visualice el apellido y la fecha de alta de todos los empleados ordenados por apellido.



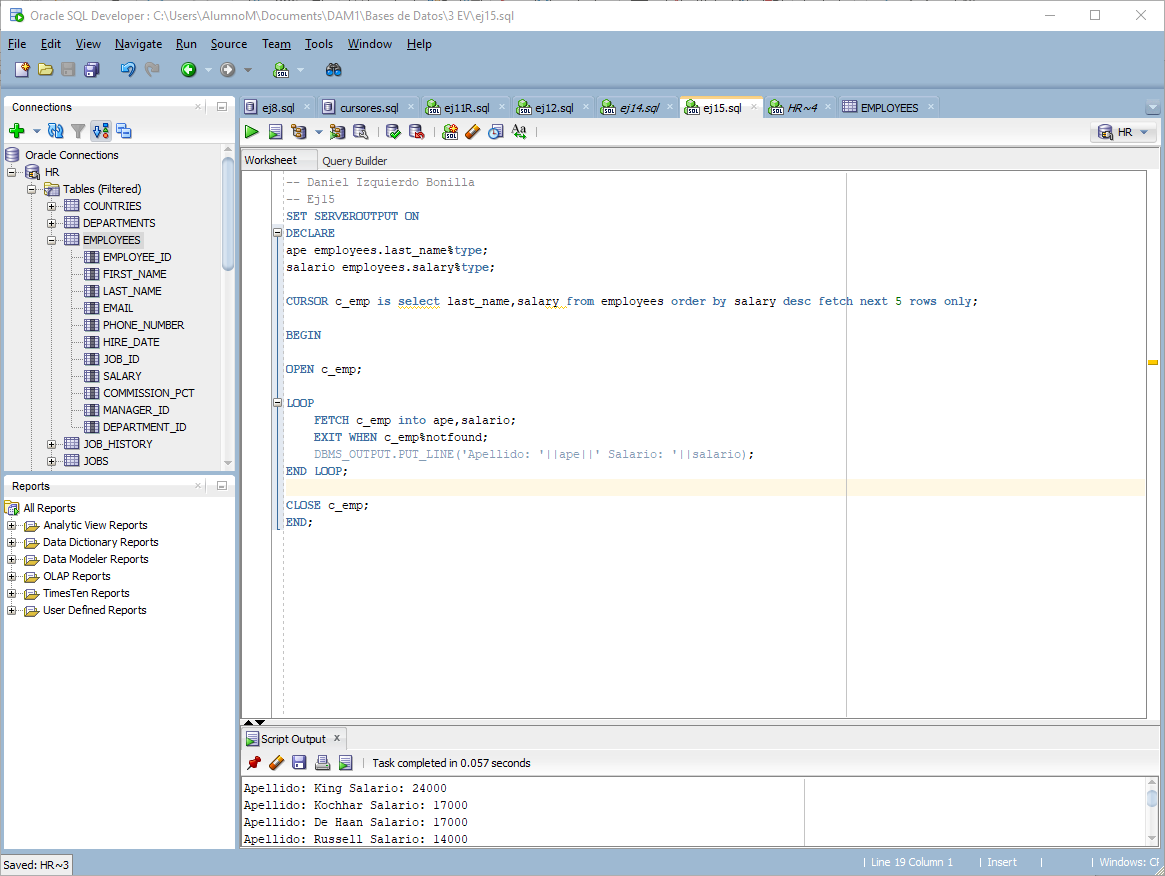
13. Codificar un programa que muestre el nombre de cada departamento y el número de empleados que tiene. (USAR FOR…LOOP)y que muestre también departamentos sin empleados (0 empleados)



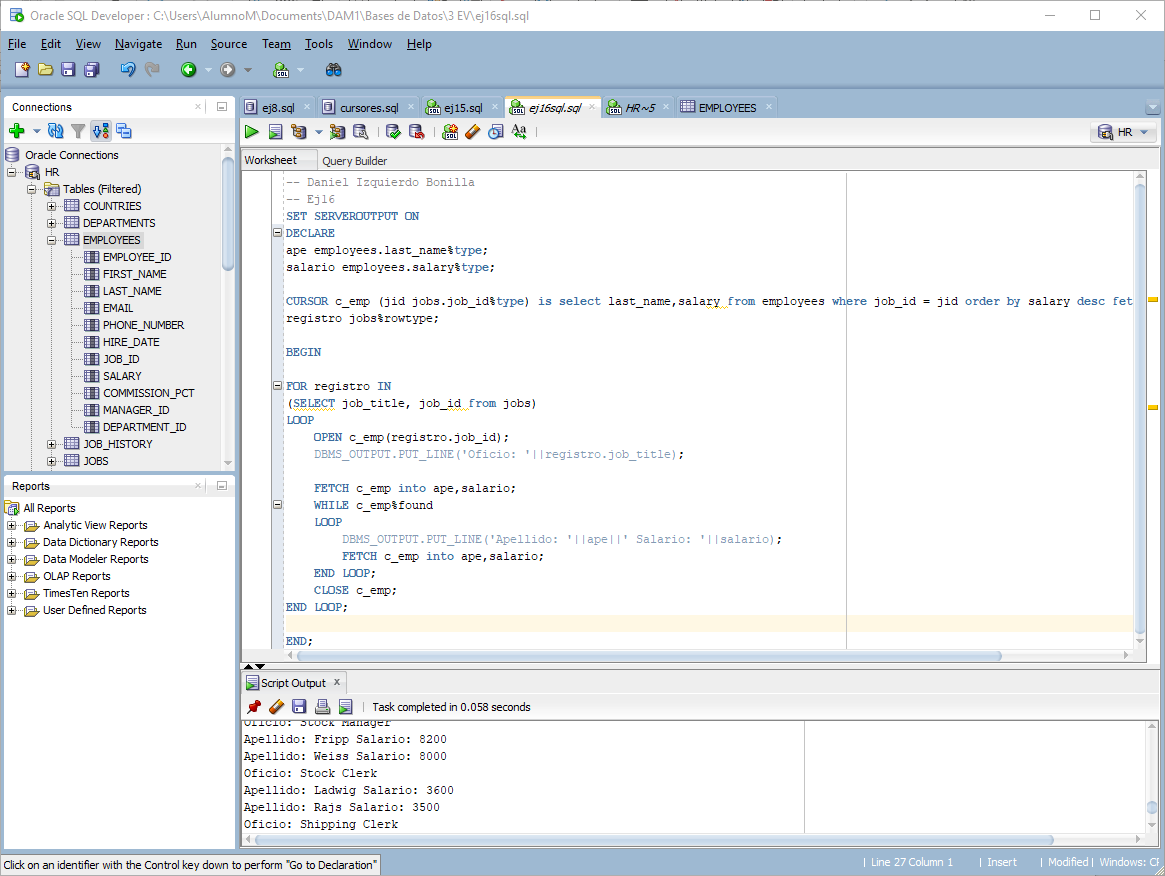
14. Escribir un programa que reciba una cadena y visualice el apellido y el número de empleado de todos los empleados cuyo apellido contenga la cadena especificada. Al finalizar visualizar el número de empleados mostrados. (Con OPEN, FETCH, CLOSE…)



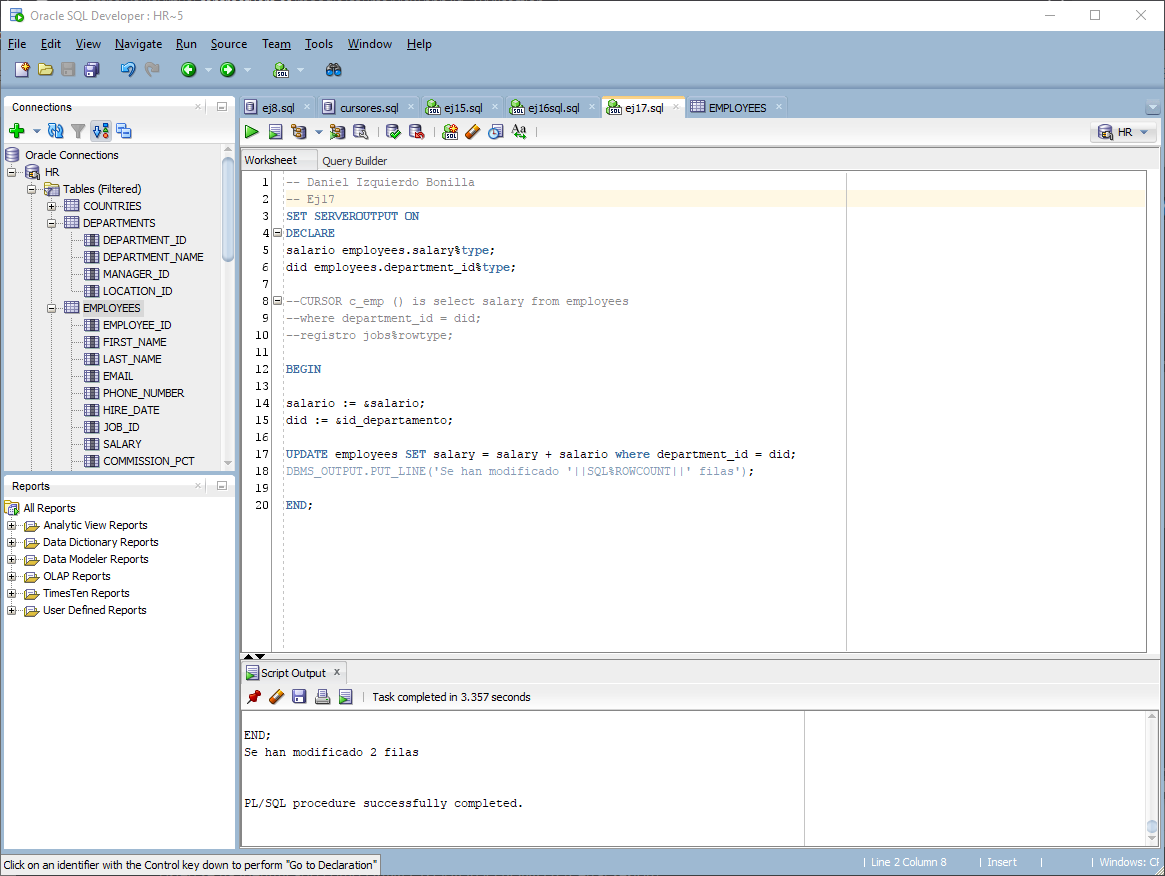
15. Escribir un programa que visualice el apellido y el salario de los cinco empleados que tienen el salario más alto. (Con OPEN, FETCH, CLOSE…)



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

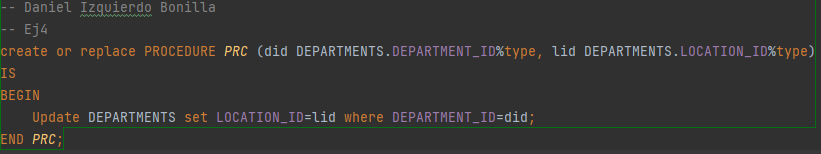


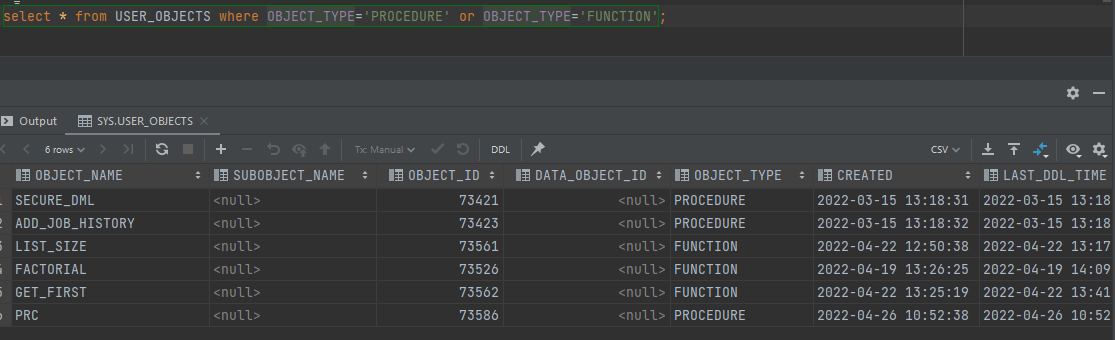
17. Escribe un programa que incremente el salario de los empleados de un determinado departamento que se pasará como primer parámetro. El incremento será una cantidad en euros que se pasará como segundo parámetro en la llamada. El programa deberá informar del número de filas afectadas por la actualización.



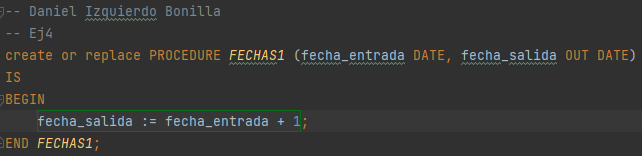
**EJERCICIOS DE FUNCIONES Y PROCEDIMIENTOS**

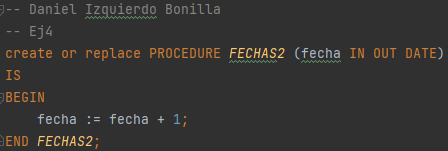
4) Escribe un procedimiento que modifique la localidad de un dpto. (con dos parámetros: el número del departamento, id,  y la nueva localidad, id)

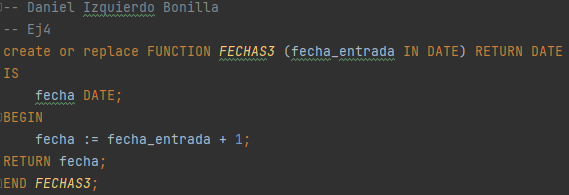
5) Visualiza todos los procedimientos y funciones del usuario almacenados en la base de datos y su situación (valid o invalid)..



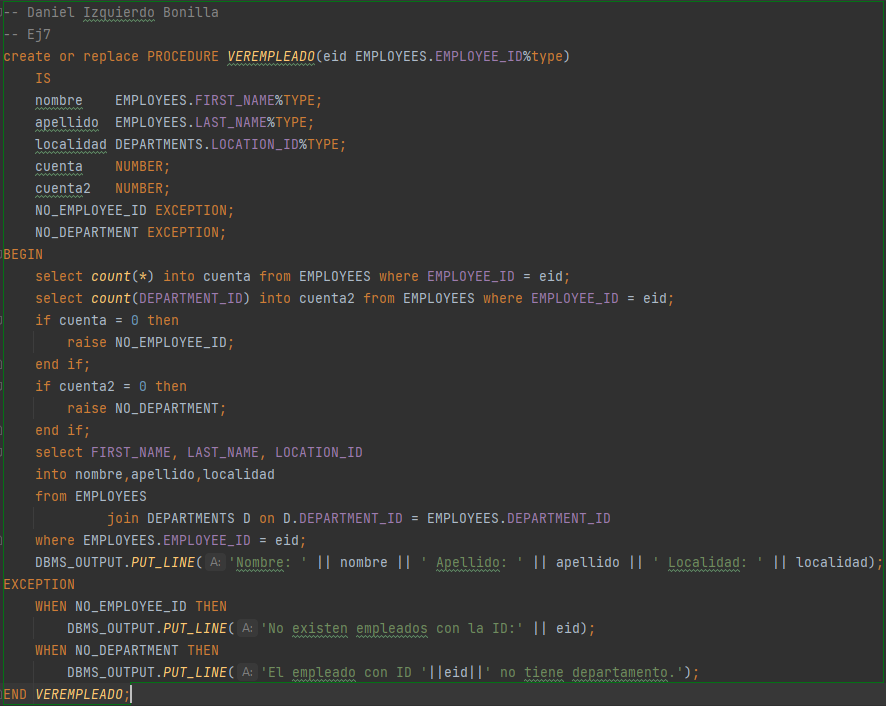
6) Hacer una procedimiento que tenga dos parámetros tipo fecha uno de entrada y otro de salida. El de salida se hará la fecha del día siguiente a la entrada. Hacer lo mismo con un solo parámetro IN OUT. Hacer lo mismo con una función con un parámetro IN



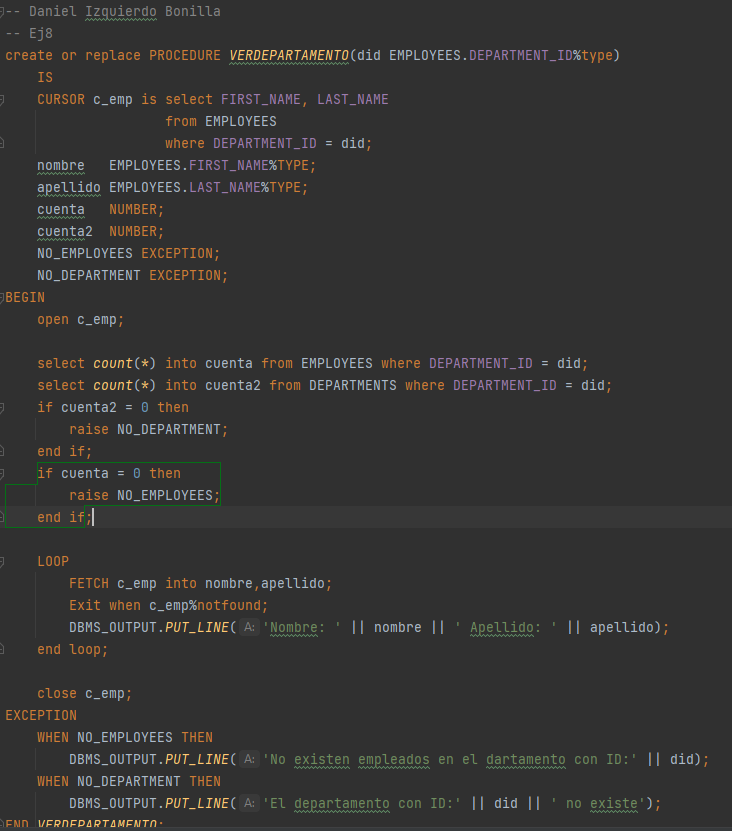




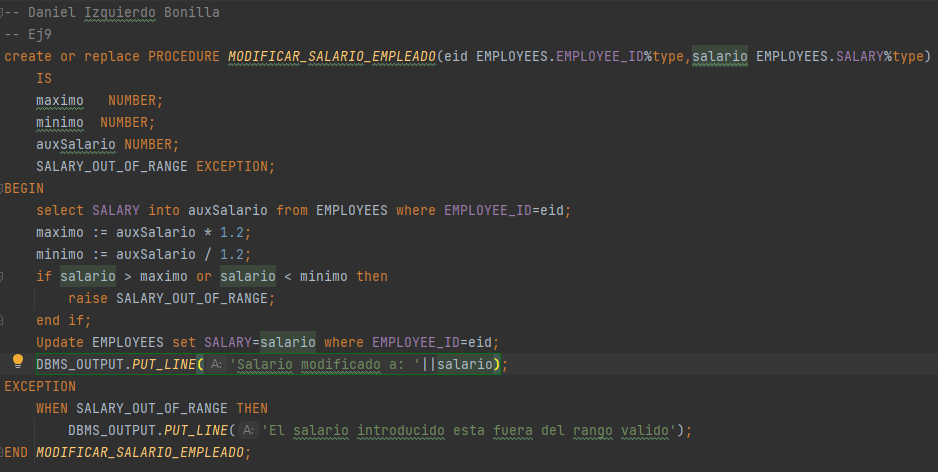
7) Crear un procedimiento llamado VEREMPLEADO que muestre el nombre, apellido y localidad del empleado cuyo ID se pasa como parámetro. (Tratar la excepción si no existe ese ID o si el empleado no tiene departamento)



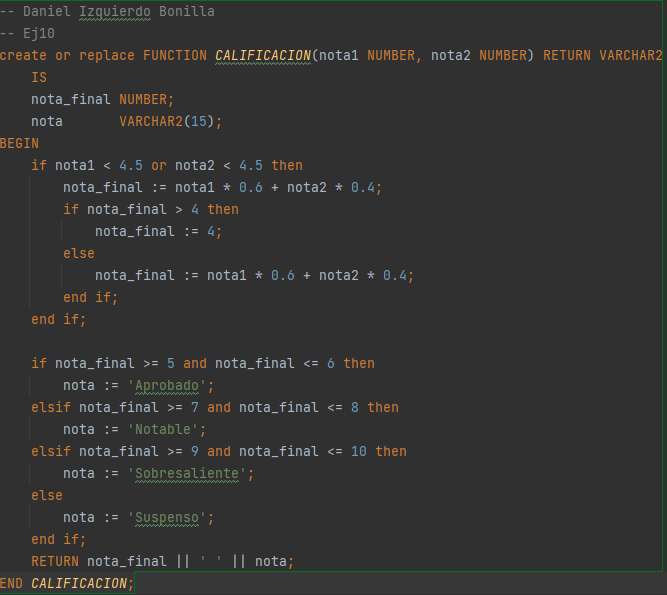
8) Crear un procedimiento que se llame VERDEPARTAMENTO que liste los empleados (nombre y apellido) de un determinado departamento cuyo ID se pasa como parámetro (Usar un cursor explícito y tratar la excepción se que no exista ese departamento o bien no tenga empleados)



9) Crear el procedimiento MODIFICAR\_SALARIO\_EMPLEADO que modifique el salario de un empleado pasándole su ID y el nuevo salario. El procedimiento comprobará que la variación de precio no supere el 20% (arriba o abajo) y lanzará una excepción en caso contrario y no efectuará la modificación. (También lanza excepción si no existe el ID)



10) Crea una función CALIFICACION que devuelva una nota final en número y letra (“Suspenso” “Aprobado”, “Notable” o “Sobresaliente”) en función de dos notas númericas de tal manera que la primera nota cuente el 60% y la segunda un 40% y que si cualquiera de las dos se suspende (<4,5) la nota final no puede ser mayor que 4.



**EJERCICIOS DE TRIGGERS**