Aplicación: Agenda de contactos

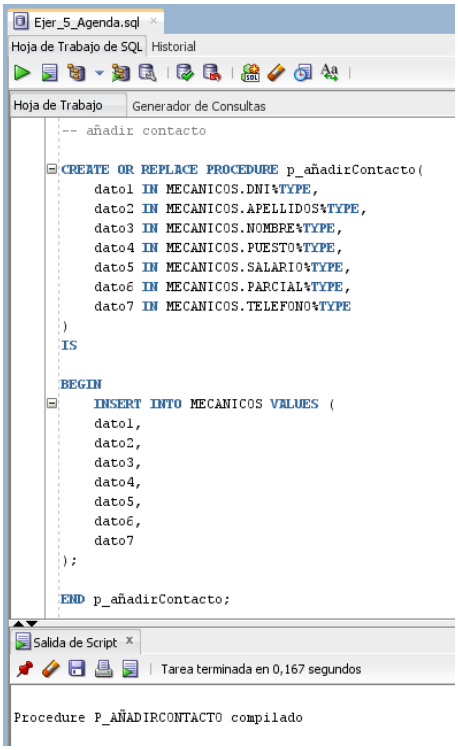
Utiliza tus conocimientos de programación PL/SQL para crear una agenda de contactos. Para ello puedes generarte tus tablas propias o utilizar las tablas de contactos de alguna base de datos que tengas (por ejemplo “taller”, para gestionar mecánicos). Para ello:

- Deberás escribir un bloque anónimo, que será el programa principal.
- Deberás pedir por teclado la opción elegida.
- En este programa principal se ofrecen varias acciones. Por ejemplo:
  - 1. Añadir contacto
  - 2. Editar teléfono
  - 3. Eliminar contacto
- Deberás escribir funciones o procedimientos (lo que tú estimes), para implementar dichas acciones
- Deberás ofrecer una respuesta (mediante DBMS_OUTPUT).

Ten en cuenta que dentro de cada acción elegida (añadir, editar, eliminar) vas a necesitar una información distinta (añadir contacto necesita todos los datos, editar necesita DNI y teléfono, eliminar necesita DNI).

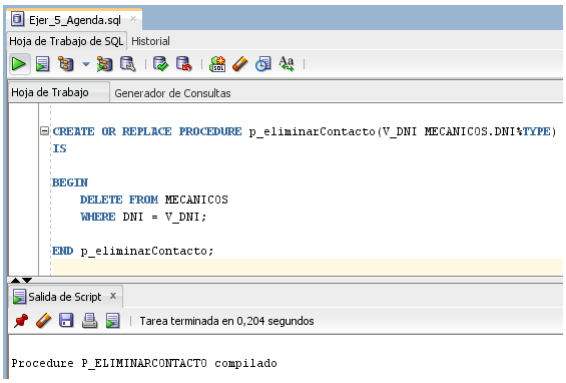
Procedimiento de añadir un contacto a la agenda

```
CREATE OR REPLACE PROCEDURE p_añadirContacto(
   dato1 IN MECANICOS.DNI%TYPE,
   dato2 IN MECANICOS.APELLIDOS%TYPE,
   dato3 IN MECANICOS.NOMBRE%TYPE,
   dato4 IN MECANICOS.PUESTO%TYPE,
   dato5 IN MECANICOS.SALARIO%TYPE,
   dato6 IN MECANICOS.PARCIAL%TYPE,
   dato7 IN MECANICOS.TELEFONO%TYPE
)
IS

BEGIN
   INSERT INTO MECANICOS VALUES (
   dato1,
   dato2,
   dato3,
   dato4,
   dato5,
   dato6,
   dato7
);

END p_añadirContacto;
```



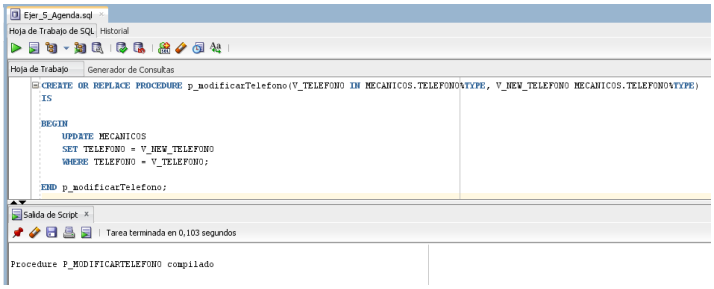
Procedimiento de eliminar un contacto de la agenda

```
CREATE OR REPLACE PROCEDURE p_eliminarContacto(V_DNI MECANICOS.DNI%TYPE)
IS

BEGIN
   DELETE FROM MECANICOS
   WHERE DNI = V_DNI;

END p_eliminarContacto;
```



Procedimiento de modificar un número de teléfono de la agenda

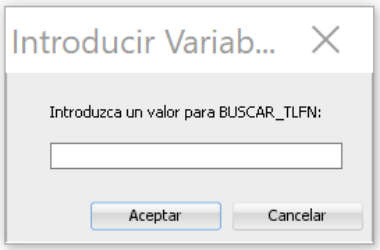
```
CREATE OR REPLACE PROCEDURE p_modificarTelefono(V_TELEFONO IN MECANICOS.TELEFONO%TYPE,
V_NEW_TELEFONO MECANICOS.TELEFONO%TYPE)
IS

BEGIN
   UPDATE MECANICOS
   SET TELEFONO = V_NEW_TELEFONO
   WHERE TELEFONO = V_TELEFONO;

END p_modificarTelefono;
```



Bloque anónimo

```
DECLARE
   V_OPCION NUMBER;
   V_DNI1 MECANICOS.DNI%TYPE;
   V_APELLIDOS1 MECANICOS.APELLIDOS%TYPE;
   V_NOMBRE1 MECANICOS.NOMBRE%TYPE;
   V_PUESTO1 MECANICOS.PUESTO%TYPE;
   V_SALARIO1 MECANICOS.SALARIO%TYPE;
   V_PARCIAL1 MECANICOS.PARCIAL%TYPE;
   V_TELEFONO1 MECANICOS.TELEFONO%TYPE;

   V_DNI MECANICOS.DNI%TYPE;
   V_TLFN MECANICOS.TELEFONO%TYPE;
   V_NEW_TLFN MECANICOS.TELEFONO%TYPE;

BEGIN
   OPCION := '&OPCION';

   CASE
      WHEN OPCION = '1' THEN
         dato1 := '&DNI';
         dato2 := '&APELLIDOS';
         dato3 := '&NOMBRE';
         dato4 := '&PUESTO';
         dato5 := '&SALARIO';
         dato6 := '&PARCIAL';
         dato7 := '&TELEFONO';
         p_añadirContacto(dato1, dato2, dato3, dato4, dato5, dato6, dato7);

      WHEN OPCION = '2' THEN
         V_TLFN := '&BUSCAR_TLFN';
         V_NEW_TLFN := '&NEW_TLFN';
         p_modificarTelefono(V_TLFN, V_NEW_TLFN);

      WHEN OPCION = '3' THEN
         V_DNI := '&DNI';
         p_eliminarContacto(V_DNI);

      END CASE;
END;
```

```
BEGIN
    OPCION := '&OPCION';

    CASE
        WHEN OPCION = '1' THEN
            dato1 := '&DNI';
            dato2 := '&APELLIDOS';
            dato3 := '&NOMBRE';
            dato4 := '&PUESTO';
            dato5 := '&SALARIO';
            dato6 := '&PARCIAL';
            dato7 := '&TELEFONO';
            p_añadirContacto(dato1, dato2, dato3, dato4, dato5, dato6, dato7);
```



El problema de esta actividad reside en que durante la ejecución del case del bloque anónimo, antes de finalizar, y aunque hayamos elegido una opción (en este caso la '1' por ejemplo) el bloque anónimo nos pide TODAS las variables antes de presentar el resultado final, es decir, el carácter ampersand nos obliga a introducir todos los datos de todas las variables que se pedirán, antes de que finalice la ejecución y se presente el resultado, lo que resulta ineficiente.