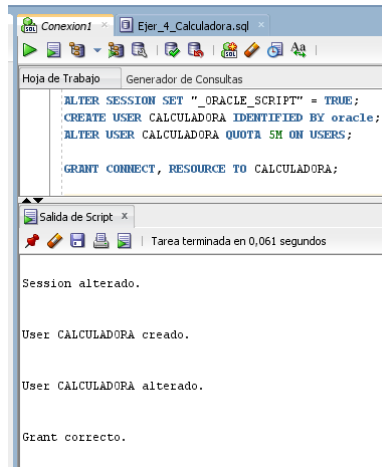


Ejer 4 Calculadora

1. Crear un programa a modo calculadora.



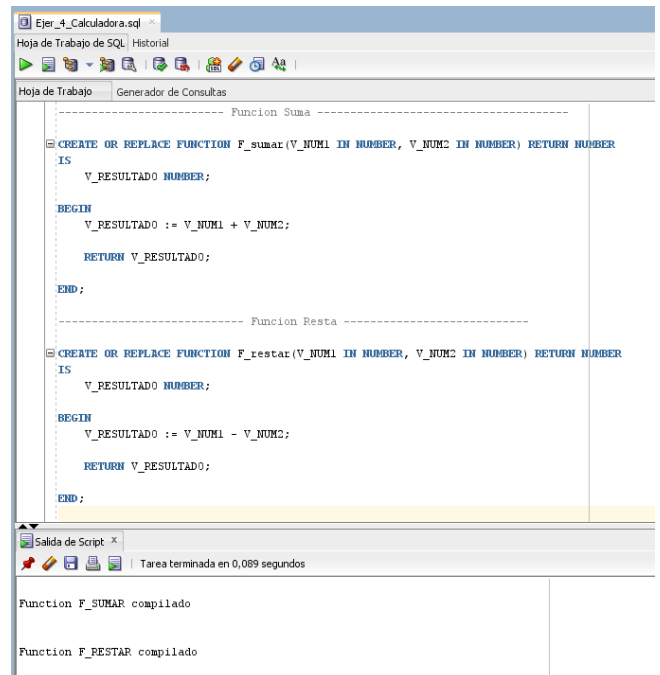
----- Funcion Suma -----

```
CREATE OR REPLACE FUNCTION F_sumar(V_NUM1 IN NUMBER, V_NUM2 IN NUMBER) RETURN
NUMBER
IS
    V_RESULTADO NUMBER;

BEGIN
    V_RESULTADO := V_NUM1 + V_NUM2;

    RETURN V_RESULTADO;

END;
```



----- Funcion Resta -----

```
CREATE OR REPLACE FUNCTION F_restar(V_NUM1 IN NUMBER, V_NUM2 IN NUMBER) RETURN
NUMBER
IS
    V_RESULTADO NUMBER;

BEGIN
    V_RESULTADO := V_NUM1 - V_NUM2;

    RETURN V_RESULTADO;

END;
```

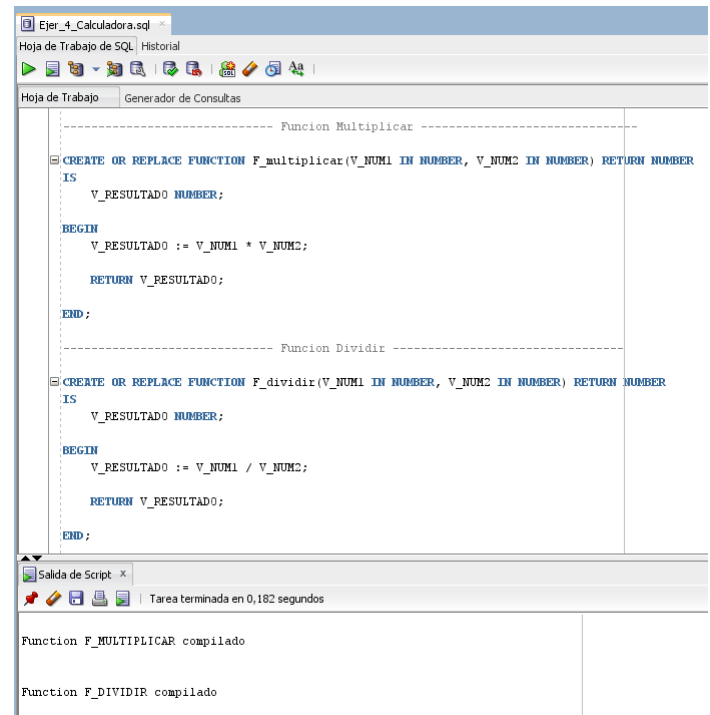
----- Funcion Multiplicar -----

```
CREATE OR REPLACE FUNCTION F_multiplicar(V_NUM1 IN NUMBER, V_NUM2 IN NUMBER) RETURN
NUMBER
IS
    V_RESULTADO NUMBER;

BEGIN
    V_RESULTADO := V_NUM1 * V_NUM2;

    RETURN V_RESULTADO;

END;
```



----- Funcion Dividir -----

```
CREATE OR REPLACE FUNCTION F_dividir(V_NUM1 IN NUMBER, V_NUM2 IN NUMBER) RETURN
NUMBER
IS
    V_RESULTADO NUMBER;

BEGIN
    V_RESULTADO := V_NUM1 / V_NUM2;

    RETURN V_RESULTADO;

END;
```

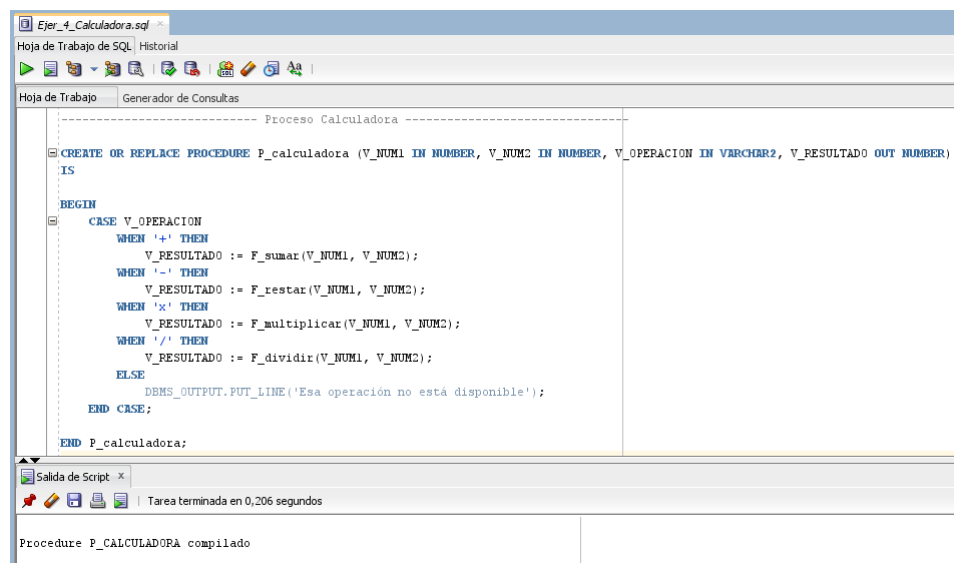
----- Proceso Calculadora -----

```

CREATE OR REPLACE PROCEDURE P_calculadora (V_NUM1 IN NUMBER, V_NUM2 IN NUMBER,
V_OPERACION IN VARCHAR2, V_RESULTADO OUT NUMBER)
IS
BEGIN
CASE V_OPERACION
WHEN '+' THEN
V_RESULTADO := F_sumar(V_NUM1, V_NUM2);
WHEN '-' THEN
V_RESULTADO := F_restar(V_NUM1, V_NUM2);
WHEN 'x' THEN
V_RESULTADO := F_multiplicar(V_NUM1, V_NUM2);
WHEN '/' THEN
V_RESULTADO := F_dividir(V_NUM1, V_NUM2);
ELSE
DBMS_OUTPUT.PUT_LINE('Esa operación no está disponible');
END CASE;

END P_calculadora;

```



----- Bloque Anónimo -----

DECLARE

```
V_NUMERO1 NUMBER;
V_NUMERO2 NUMBER;
V_OPERACION VARCHAR2(1);
V_SOLUCION NUMBER;
```

BEGIN

```
V_NUMERO1 := &NUMERO1;
V_NUMERO2 := &NUMERO2;
V_OPERACION := '&OPERACION';
```

```
P_calculadora(V_NUMERO1, V_NUMERO2, V_OPERACION, V_SOLUCION);
```

```
DBMS_OUTPUT.PUT_LINE('Operacion: ' || V_NUMERO1 || ' ' || V_OPERACION || ' ' || V_NUMERO2 || ' '
= ' || V_SOLUCION);
```

END;

```

----- Bloque Anónimo -----

DECLARE
  V_NUMERO1 NUMBER;
  V_NUMERO2 NUMBER;
  V_OPERACION VARCHAR2(1);
  V_SOLUCION NUMBER;

BEGIN
  V_NUMERO1 := &NUMERO1;
  V_NUMERO2 := &NUMERO2;
  V_OPERACION := '&OPERACION';

  P_calculadora(V_NUMERO1, V_NUMERO2, V_OPERACION, V_SOLUCION);

  DBMS_OUTPUT.PUT_LINE('Operacion: ' || V_NUMERO1 || ' ' || V_OPERACION || ' ' || V_NUMERO2 || ' ' = ' || V_SOLUCION);

END;

```

Salida de Script x

Tarea terminada en 5,182 segundos

```

DBMS_OUTPUT.PUT_LINE('Operacion: ' || V_NUMERO1 || ' ' || V_OPERACION || ' ' || V_NUMERO2 || ' ' = ' || V_SOLUCION);

END;
Procedimiento PL/SQL terminado correctamente.

```

Salida de DBMS

Tamaño de Buffer: 20000

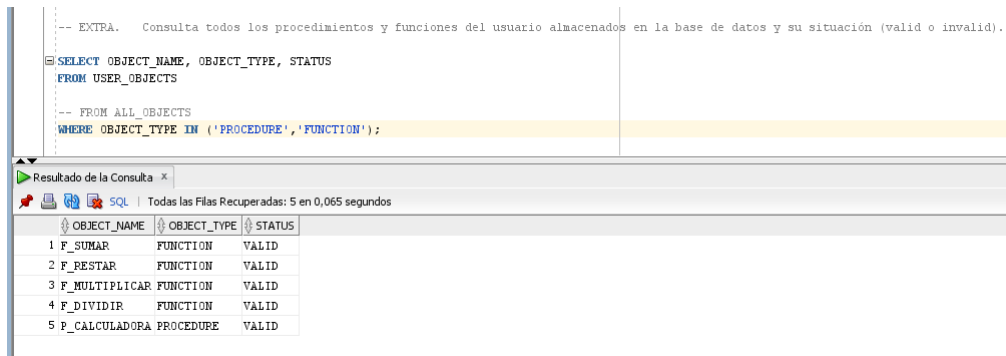
PL_SQL_Ejer_4_Calculadora x

Operacion: 1 + 2 = 3

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

```
SELECT OBJECT_NAME, OBJECT_TYPE, STATUS  
FROM USER_OBJECTS
```

```
-- FROM ALL_OBJECTS  
WHERE OBJECT_TYPE IN ('PROCEDURE','FUNCTION');
```



The screenshot shows a SQL query execution interface. The query is as follows:

```
-- EXTRA. Consulta todos los procedimientos y funciones del usuario almacenados en la base de datos y su situación (valid o invalid).  
SELECT OBJECT_NAME, OBJECT_TYPE, STATUS  
FROM USER_OBJECTS  
-- FROM ALL_OBJECTS  
WHERE OBJECT_TYPE IN ('PROCEDURE','FUNCTION');
```

The results are displayed in a table with the following columns: OBJECT_NAME, OBJECT_TYPE, and STATUS. The table contains 5 rows of data.

	OBJECT_NAME	OBJECT_TYPE	STATUS
1	F_SUMAR	FUNCTION	VALID
2	F_RESTAR	FUNCTION	VALID
3	F_MULTPLICAR	FUNCTION	VALID
4	F_DIVIDIR	FUNCTION	VALID
5	P_CALCULADORA	PROCEDURE	VALID