



Unidad 2: Funciones y Procedimientos (Parte 4)

Dr. Rogelio Mamani Ramos, Ph.D.

Abril, 2025



2.2.8. Condicionales

- La sentencia *if* tiene una estructura amigable:

```
IF condición THEN  
    -- Bloque de código por verdad  
END IF;
```



2.2.8. Condicionales

- También se puede trabajar elsif y else:

```
IF condición1 THEN
    -- por verdad
ELSIF condición2 THEN
    -- condicion1 falso y verdad condicion 2
ELSE
    -- si las condiciones anteriores son FALSE
END IF;
```



2.2.8. Condicionales

- En el ejemplo: Verifique si el numero ingresado es par o impar

```
CREATE OR REPLACE FUNCTION verifica_par_impar(val INTEGER) RETURNS TEXT AS
$$
BEGIN
    IF val % 2 = 0 THEN
        RETURN 'numero' || val || ' es par.';
    ELSE
        RETURN 'numero ' || val || ' es impar.';
    END IF;
END;
$$
LANGUAGE plpgsql;
select verifica_par_impar(5)
```



2.2.9. bucle while

- En el ejemplo: Mostrar la serie de los n primeros números naturales, se observa la aplicación del *while*

```
CREATE OR REPLACE FUNCTION serie_entero(n INTEGER) RETURNS TEXT AS
$$
DECLARE
    c INTEGER := 1;
    res TEXT := '';
BEGIN
    WHILE c <= n LOOP
        res := res || c;
        IF c < n THEN
            res := res || ', ';
        END IF;
        c := c + 1;
    END LOOP;
    RETURN res;
END;
$$
LANGUAGE plpgsql;
```



2.2.9. bucle for

- En el ejemplo: Mostrar la serie de los n primeros números naturales, se observa la aplicación del *for*

```
CREATE OR REPLACE FUNCTION serie_enterodos(n INTEGER) RETURNS TEXT LANGUAGE plpgsql AS
$$
DECLARE
    i INTEGER;
    res TEXT := '';
BEGIN
    FOR i IN 1..n LOOP
        res := res || i;
        IF i < n THEN
            res := res || ', ';
        END IF;
    END LOOP;
    RETURN res;
END;
$$;

SELECT serie_enterodos(3);
```