**Boucles :**

**1. Boucle WHILE :**

sql

CREATE OR REPLACE FUNCTION boucle\_while()

RETURNS VOID AS $$

DECLARE

i INT := 1;

BEGIN

WHILE i <= 5 LOOP

RAISE NOTICE 'Valeur de i : %', i;

i := i + 1;

END LOOP;

END;

$$ LANGUAGE plpgsql;

**2. Boucle FOR :**

sql

CREATE OR REPLACE FUNCTION boucle\_for()

RETURNS VOID AS $$

DECLARE

i INT;

BEGIN

FOR i IN 1..5 LOOP

RAISE NOTICE 'Valeur de i : %', i;

END LOOP;

END;

$$ LANGUAGE plpgsql;

**3. Boucle LOOP avec EXIT WHEN :**

sql

CREATE OR REPLACE FUNCTION boucle\_loop()

RETURNS VOID AS $$

DECLARE

i INT := 1;

BEGIN

LOOP

RAISE NOTICE 'Valeur de i : %', i;

EXIT WHEN i > 5;

i := i + 1;

END LOOP;

END;

$$ LANGUAGE plpgsql;

**Cases (Switch) :**

En PostgreSQL, il n'y a pas de construction directe pour les switch/case comme dans certains autres langages. Cependant, vous pouvez utiliser des blocs IF-ELSE ou une clause CASE pour simuler le comportement d'un switch/case.

**1. Bloc IF-ELSE :**

sql

CREATE OR REPLACE FUNCTION tester\_case(valeur INT)

RETURNS VOID AS $$

BEGIN

IF valeur = 1 THEN

RAISE NOTICE 'Cas 1';

ELSIF valeur = 2 THEN

RAISE NOTICE 'Cas 2';

ELSE

RAISE NOTICE 'Autre cas';

END IF;

END;

$$ LANGUAGE plpgsql;

**2. Clause CASE :**

sql

CREATE OR REPLACE FUNCTION tester\_case(valeur INT)

RETURNS VOID AS $$

BEGIN

CASE valeur

WHEN 1 THEN RAISE NOTICE 'Cas 1';

WHEN 2 THEN RAISE NOTICE 'Cas 2';

ELSE RAISE NOTICE 'Autre cas';

END CASE;

END;

$$ LANGUAGE plpgsql;

Pour utiliser ces exemples, appelez simplement la fonction tester\_case avec la valeur appropriée pour tester chaque cas :

sql

SELECT tester\_case(1); -- Résultat : Cas 1

SELECT tester\_case(2); -- Résultat : Cas 2

SELECT tester\_case(3); -- Résultat : Autre cas