**PL/pgSQL**

PL/pgSQL is influenced by Oracle SQL. It allows for loops, conditionals, functions, data types and much more.

CREATE OR REPLACE FUNCTION func\_name(parameter par\_type) RETURNS ret\_type AS

$body$

BEGIN

--statements

END

$body$

LANGUAGE plpqsql

**Get Product Price by Name**

SELECT item.price

FROM item

NATURAL JOIN product

WHERE product.name = 'Grandview';

CREATE OR REPLACE FUNCTION fn\_get\_price\_product\_name(prod\_name varchar)

RETURNS numeric AS

$body$

BEGIN

RETURN item.price

FROM item

NATURAL JOIN product

WHERE product.name = prod\_name;

END

$body$

LANGUAGE plpgsql

SELECT fn\_get\_price\_product\_name('Grandview');

**Using Variables in Functions**

--Create variables in functions

CREATE OR REPLACE FUNCTION fn\_get\_sum(val1 int, val2 int)

RETURNS int AS

$body$

--Put variables here

DECLARE

ans int;

BEGIN

ans := val1 + val2;

RETURN ans;

END;

$body$

LANGUAGE plpgsql

SELECT fn\_get\_sum(4,5);

**Assign Variable Value with a Query**

--Get random number and assign it to a variable

CREATE OR REPLACE FUNCTION fn\_get\_random\_number(min\_val int, max\_val int)

RETURNS int AS

$body$

--Put variables here

DECLARE

rand int;

BEGIN

SELECT random()\*(max\_val - min\_val) + min\_val INTO rand;

RETURN rand;

END;

$body$

LANGUAGE plpgsql

SELECT fn\_get\_random\_number(1, 5);

**Store Rows in Variables & Concat**

--Get random sales person name

CREATE OR REPLACE FUNCTION fn\_get\_random\_salesperson()

RETURNS varchar AS

$body$

--Put variables here

DECLARE

rand int;

--Use record to store row data

emp record;

BEGIN

--Generate random number

SELECT random()\*(5 - 1) + 1 INTO rand;

--Get row data for a random sales person and store in emp

SELECT \*

FROM sales\_person

INTO emp

WHERE id = rand;

--Concat the first and last name and return it

RETURN CONCAT(emp.first\_name, ' ', emp.last\_name);

END;

$body$

LANGUAGE plpgsql

SELECT fn\_get\_random\_salesperson();

**IN INOUT and OUT**

--These can be used to except and return multiple values without return

CREATE OR REPLACE FUNCTION fn\_get\_sum\_2(IN v1 int, IN v2 int, OUT ans int) AS

$body$

BEGIN

ans := v1 + v2;

END;

$body$

LANGUAGE plpgsql

SELECT fn\_get\_sum\_2(4,5);

**Using Multiple Outs**

-- Get a customer born in given month

CREATE OR REPLACE FUNCTION fn\_get\_cust\_birthday(IN the\_month int, OUT bd\_month int, OUT bd\_day int, OUT f\_name varchar, OUT l\_name varchar) AS

$body$

BEGIN

SELECT EXTRACT(MONTH FROM birth\_date), EXTRACT(DAY FROM birth\_date),

first\_name, last\_name

INTO bd\_month, bd\_day, f\_name, l\_name

FROM customer

WHERE EXTRACT(MONTH FROM birth\_date) = the\_month

LIMIT 1;

END;

$body$

LANGUAGE plpgsql

SELECT fn\_get\_cust\_birthday(12);