

POSTGRESQL - FUNCTIONS

http://www.tutorialspoint.com/postgresql/postgresql_functions.htm

Copyright © tutorialspoint.com

PostgreSQL **functions**, also known as Stored Procedures, allow you to carry out operations that would normally take several queries and round trips in a single function within the database. Functions allow database reuse as other applications can interact directly with your stored procedures instead of a middle-tier or duplicating code.

Functions can be created in language of your choice like SQL, PL/pgSQL, C, Python, etc.

Syntax

The basic syntax to create a function is as follows:

```
CREATE [OR REPLACE] FUNCTION function_name (arguments)
RETURNS return_datatype AS $variable_name$
DECLARE
    declaration;
    [...]
BEGIN
    < function_body >
    [...]
    RETURN { variable_name | value }
END; LANGUAGE plpgsql;
```

Where,

- **function-name** specifies the name of the function.
- [OR REPLACE] option allows modifying an existing function.
- The function must contain a **return** statement.
- **RETURN** clause specifies that data type you are going to return from the function. The **return_datatype** can be a base, composite, or domain type, or can reference the type of a table column.
- **function-body** contains the executable part.
- The AS keyword is used for creating a standalone function.
- **plpgsql** is the name of the language that the function is implemented in. Here, we use this option for PostgreSQL, it Can be SQL, C, internal, or the name of a user-defined procedural language. For backward compatibility, the name can be enclosed by single quotes.

Example

The following example illustrates creating and calling a standalone function. This function returns the total number of records in the COMPANY table. We will use the [COMPANY](#) table, which has the following records:

```
testdb# select * from COMPANY;
 id | name  | age | address  | salary
----+-----+----+-----+-----
  1 | Paul  | 32  | California | 20000
  2 | Allen | 25  | Texas     | 15000
  3 | Teddy | 23  | Norway    | 20000
  4 | Mark  | 25  | Rich-Mond | 65000
  5 | David | 27  | Texas     | 85000
  6 | Kim   | 22  | South-Hall | 45000
  7 | James | 24  | Houston   | 10000
(7 rows)
```

Function totalRecords() is as follows:

```
CREATE OR REPLACE FUNCTION totalRecords ()  
RETURNS integer AS $total$  
declare  
    total integer;  
BEGIN  
    SELECT count(*) into total FROM COMPANY;  
    RETURN total;  
END;  
$total$ LANGUAGE plpgsql;
```

When the above query is executed, the result would be:

```
testdb# CREATE FUNCTION
```

Now, let's execute a call to this function and check the records in the COMPANY table

```
testdb=# select totalRecords();
```

When the above query is executed, the result would be:

```
totalrecords  
-----  
              7  
(1 row)
```