Práctica 27

Funciones en psql

Objetivo

En esta práctica se analizan los diferentes tipos de funciones que se pueden programar en PostgreSQL, así como la forma de crear un sistema de reglas.

Introducción

El SMBD PostgreSQL proporciona tres tipos de funciones:

- Funciones de lenguaje de consultas (funciones escritas en SQL)
- Funciones de lenguaje procedural (funciones escritas en lenguaje PLSQL)
- Funciones de lenguaje de programación (funciones escritas en un lenguaje de programación compilado como C)

Equipo necesario

Una computadora con sistema operativo ***Windows*** que cuente con ***psql***.

Metodología

1. Ejecute la herramienta ***SQL Shell (psql)***.
2. Deje el usuario ***postgres*** y proporcione la contraseña definida en la primera práctica.
3. Cambie a la base de datos Biblioteca:

```
postgres=# \c Biblioteca;
```

4. En este primer ejemplo se crea una función que toma un argumento de tipo texto, que corresponde al nombre de una editorial, y devuelve un resultado de un tipo definido por el usuario, que es la dirección de ésta:

```
Biblioteca=# CREATE FUNCTION dir(text)
Biblioteca-# RETURNS direc
Biblioteca-# AS 'SELECT "Dirección" FROM Autores WHERE
Nombre=$1'
```

```
Biblioteca-# language 'sql';
```

5. Para probar esta función, ejecute la siguiente llamada mediante una sentencia SELECT:

```
Biblioteca=# SELECT dir('Homero Simpson');
```

6. Obtenga una imagen completa de la pantalla, mostrando la llamada a la función ***dir***, consérvela como evidencia.

7. Ahora se va a crear una función que calcule la edad de una persona, mediante la diferencia entre dos fechas:

```
Biblioteca=# CREATE FUNCTION edad(fecha1 DATE, fecha2 DATE)
Biblioteca-# RETURNS DOUBLE PRECISION
Biblioteca-# AS 'SELECT (extract(year from fecha2)-extract(year
from fecha1))
Biblioteca'# CASE WHEN (extract(DOY from fecha2)<extract(DOY
from fecha1))
Biblioteca'# THEN 1 ELSE 0 END'
Biblioteca-# language 'sql';
```

8. Después de haber escrito la función anterior, podrá usarla para calcular la edad que tiene actualmente el jugador César Saúl "el chino" Herrera, que nació el 3 de diciembre de 2000, escriba la llamada a la función ***edad***, empleando el formato internacional de fechas ***ISO***:

```
Biblioteca=# SELECT edad('2000-12-03', CURRENT_DATE);
```

9. Se obtiene el mismo resultado si escribe la fecha en el formato habitual:

```
Biblioteca=# SELECT edad('30/12/2000', CURRENT_DATE);
```

10. También puede calcular la edad que tenía "o rei" Pelé, al ganar Brasil su tercer campeonato mundial de Futbol, si nació el 23 de octubre de 1940 y la final del Mundial de México se jugó el 21 de junio de 1970 en el estadio Azteca venciendo Brasil a Italia por 4 – 1.

```
Biblioteca=# SELECT edad('23-10-1940', '1970/06/21') as "Edad
que tenía Pelé";
```

11. Obtenga una imagen completa de la pantalla, mostrando la creación y las tres llamadas de la función ***edad***, consérvela como evidencia.

12. A veces es necesario contar con una función que permita convertir una fecha a un formato de texto, por lo que se pide que escriba la siguiente función:

```
Biblioteca=# CREATE FUNCTION fecha(f DATE) RETURNS text
Biblioteca-# AS 'SELECT TO_CHAR(f, ''FMDay DD" de "FMMonth" de
"YYYY'')'
Biblioteca-# language 'sql';
```

13. (el prefijo FM elimina espacios en blanco innecesarios en los nombres del día y del mes):

14. Como ejemplo deberá convertir a texto la fecha de hoy:

```
Biblioteca=# SELECT fecha(current_date);
```

15. Se presenta el inconveniente que los nombres del día y el del mes aparecen en inglés.

16. Obtenga una imagen completa de la pantalla, mostrando la creación y la llamada de la función ***fecha***, consérvela como evidencia.

17. Ejecute la siguiente consulta:

```
Biblioteca=# SHOW lc_time;
```

18. Aunque el formato de las fechas corresponda a México, ***Spanish_Mexico.1252***, se mostrarán los nombres de los meses y días en inglés.

19. A continuación, se creará una función en lenguaje procedural (*plpgsql*) para implementar el cálculo de la función factorial de un entero, como se muestra a continuación:

```
Biblioteca=# CREATE OR REPLACE FUNCTION Factorial(N NUMERIC)
Biblioteca-# RETURNS NUMERIC
Biblioteca-# AS $$
Biblioteca$# DECLARE
Biblioteca$# FACT NUMERIC;
Biblioteca$# I NUMERIC;
Biblioteca$# BEGIN
Biblioteca$# FACT:=1;
Biblioteca$# FOR I IN 1 .. N LOOP
Biblioteca$# FACT:=FACT*I;
Biblioteca$# END LOOP;
Biblioteca$# RETURN FACT;
Biblioteca$# END; $$
Biblioteca-# LANGUAGE plpgsql;
```

20. Ahora ejecútelo de la siguiente manera:

```
Biblioteca=# SELECT Factorial(5::NUMERIC);
```

21. Observe que puede calcular la función factorial de números más grandes:

```
Biblioteca=# SELECT Factorial(65::NUMERIC);
```

22. Obtenga una imagen completa de la pantalla, mostrando la creación y las dos llamadas de la función ***factorial***, consérvela como evidencia.

23. Defina una función que calcule la sumatoria de los cuadrados de los primeros N números naturales.

24. Ejecute la función para N=10.

25. Obtenga una imagen completa de la pantalla, mostrando la creación y ejecución de esta función, consérvela como evidencia.

26. Cierre la consola del ***psql***.
27. Fin de la Práctica

Evidencias

El alumno deberá enviar al instructor las evidencias requeridas durante la realización de la práctica.

Sugerencias didácticas

El instructor deberá atender a los alumnos que tengan dificultades en la interpretación y la realización de las instrucciones de la práctica.

Resultados

Se aprendió a crear funciones utilizando la herramienta ***PostgreSQL***.

Bibliografía

- http://es.tldp.org/Postgresql-es/web/navegable/user/sql-createfunction.html