Pgpymongo y Pgpycouch: extensiones de PostgreSQL para interactuar con MongoDB y CouchDB

Anthony R. Sotolongo León asotolongo@uci.cu

Resumen

El uso de las tecnologías de almacenamiento de tipo NoSQL se hace cada vez más frecuente en los sistemas actuales, la capacidad de extensibilidad que tiene el gestor PostgreSQL nos permite realizar funciones para interactuar con otros tipos de gestores como los son los NoSQL. En este trabajo se presentan dos extensiones de PostgreSQL para interactuar con las bases de datos NoSQL documentales MongoDB y CocuhDB escritas en plpython. Se muestran las funciones de cada una y se ejemplifica su utilización.

Las tendencias actuales de las tecnologías tienen en la mira a las bases de datos(BDs) NoSQL[i]. Sobre todo por su habilidad de libres de esquemas y horizontalmente escalables, pilares fundamentales en los sistemas dinámicos de la actualidad.

Algunos expertos[ii] en el tema la clasifican en:

Llave-valor (Redis, Voldemort**)**
Orientadas a documentos (MongoDB, CouchDB)
Por columnas (HBASE,Cassandra, Bigtable)
Grafos (Neo4j)

Varias empresas le han dado un uso bien importante tales como:

Además estas empresas utilizan en sus sistemas BDs relacionales. Es decir que la interacción entre las NoSQL y las relacionales es cada vez mayor.

En PostgreSQL se ha desarrollado la solución NoSQL llave-valor con el tipo de dato **Hstore**, el cual es el buen paso del gestor hacia la nueva tendencia del almacenamiento, además se le ha implementado varios conectores de datos externos (Foreing Data Wrapper por sus siglas en inglés) para acceder a Redis y CouchDB pero solo de lectura.

Se presenta extensiones de PostgreSQL capaz de interactuar con las bases de datos NoSQL orientadas a documentos Mongodb y Couchdb con nombres: **Pgpymongo y Pgpycouch.**

Tabla 1 – características generales de mongoDB y couchDB

Características /Gestor	mongoDB	CouchDB
Lenguaje	C++	Erlang
Objetos	Colecciones, documentos	Documentos, vistas
concurrencia	En el Lugar	MVCC
Soporte Map/reduce	sí	sí
Replicación	Maestro-Esclavo	Maestro - Maestro
Acceso	TCP/IP	HTTP
Almacenamiento de Archivos	GridFS	Attachments

Arquitectura de la extensión

La arquitectura de la extensión consiste en tener un esquema para cada base de datos documental y dentro de ellas funciones en PostgreSQL escritas en Plpython las cuales van a utilizar las bibliotecas de acceso a las distintas bases de datos NoSQL , pymongo para MongoDB y python-couchDB CouchDB.

Plpython

Plpython es un lenguaje procedural interno de PostgreSQL que permite que se escriba código de funciones en python y sea ejecutado desde PostgreSQL, es catalogado de desconfianza pues con él se puede acceder a recursos del servidor pero si se utiliza correctamente se puede lograr varias tareas, pues como se puede escribir código en python se logran utilizar sus funciones y bibliotecas del lenguaje, las cuales tienen una excelente documentación.

Funcionalidades de las extensiones.

Entre las funcionalidades de las extensiones están las siguientes:

- Gestión de una Base de datos (creación, eliminación, obtención).
- Gestión de documentos (creación, eliminación, modificación y obtención).
- Ejecución de funciones Map/Reduce.
- Obtención de la versión del servidor.

Además cada una posee varias funcionalidades específicas del gestor como son:

- CouchDB:
  - Compactación de documentos.
  - Configuración de la replicación.
  - Consultar una vista.
  - Obtener la versión de un documento.
  - Obtener un valor UUDI.
- MongoDB:
  - Gestión de colecciones (creación, eliminación y obtención).
  - Creación de índices.

Extensiones en PostgreSQL.

Las extensiones en PostgreSQL permiten agregar al gestor nuevas funcionalidades, suelen incluir múltiples objetos de SQL, por ejemplo, un nuevo tipo de datos o nuevas funciones.

Para definir una extensión se necesita por lo menos dos archivos, uno que sea un script que contiene los comandos SQL para crear objetos de la extensión, y un archivo de control que describe varias propiedades básicas de la extensión.

Para agregarle a PostgreSQL la extensión es solo agregarla al directorio *extension* y ejecutar el comando CREATE EXTESION nombre_de_extensión.

A continuación se muestra una relación de las funciones de cada extensión:

Funciones de Couchdb	**Descripción**
add_doc(text, text, text)	Agregar un documento
couchversion(text)	Obtener la versión del servidor
createdb(text, text)	Crear una base de datos en el servidor
db_compact(text, text)	Comprimir la base de datos
db_del_doc(text, text, text)	Eliminar un documento especifico
db_get_doc(text, text, text)	Obtener un document especifico
deletedb(text, text)	Eliminar una base de datos
get_doc_rev(text, text, text)	Obtener la versión especifica de un documento
get_query(text, text, text)	Ejecutar una consulta escrita en map/reduce
get_view(text, text, text)	Obtener los datos de una vista
getuudi(text)	Obtener un numero UUDI
replica(text, text, text, text)	Configurar una réplica del servidor
upd_doc(text, text, text, text)	Actualizar un documento

Funciones de MongoDB	**Descripción**
collectioncount(text, integer, text, text)	Cantidad de documentos de una colección
collectiondelete(text, integer, text, text, text)	Eliminar documentos de una colección
collectionfind(text, integer, text, text, text)	Buscar documentos en una colección
collectionfindone(text, integer, text, text, text)	Buscar un documentos específico en una colección
collectioninsert(text, integer, text, text, text)	Insertar un documento en una colección
collectionsimpleindex(text, integer, text, text, text, text)	Crear un índice en una colección
collectionupdate(text, integer, text, text, text, text)	Actualizar un documentos específicos en una colección
createcolleccion(text, integer, text, text)	Crear una colección
createdb(text, integer, text)	Crear una base de datos
deletecolleccion(text, integer, text, text)	Eliminar una colección
deletedb(text, integer, text)	Eliminar una base de datos
getcolleccions(text, integer, text)	Obtener las colecciones
getdatabases(text, integer)	Obtener las bases de datos
mongoversion(text, integer)	Versión del servidor de mongodb

Ejemplos:

Pgcouch.control

```
#pgpycouch extension
comment = 'extension para gestionar bases de datos couchdb desde postgresql'
default_version = '0.1'
relocatable = true
superuser = true
```

Pgmongo.control

```
#pgpymongo extension
comment = 'extension para gestionar bases de datos mongodb desde postgresql'
default_version = '0.1'
relocatable = true
superuser = true
```

“.SQL”

```
CREATE SCHEMA pgpycouch;
CREATE OR REPLACE FUNCTION pgpycouch.createdb(pserver text, pname text)  RETURNS text AS $$
import couchdb
#"http://ip:puerto"
servidor=couchdb.Server(str(pserver))
return servidor.create(pname)
$$  LANGUAGE plpythonu VOLATILE;


CREATE SCHEMA pgpymongo;
CREATE OR REPLACE FUNCTION pgpymongo.collectioninsert(pserver text, pport integer, pbdname text, pcollection text, pdoc text)  RETURNS text AS $BODY$
from pymongo import Connection
import json
#servidor puerto
servidor=Connection(pserver,pport)
bd =servidor[pbdname]
coll=bd[pcollection]
doc=json.loads(pdoc)
return coll.insert(doc)
$BODY$  LANGUAGE plpythonu VOLATILE;
```

Luego para utilizar las funciones escritas en plpython se ejecutan con la sentencia “Select”

Ejemplo de reación de una base de datos en couchdb:

select pgpycouch.createdb('http://localhost:5984', 'nueva')

Resultado- "<Database 'nueva'>"

Ejemplo creación de un documento en una base de datos Mongodb:

select pgpymongo.collectioninsert('localhost', 27017, 'nueva', 'micoleccion', '{"valor1":1,"valor2":"prueba"}')

Resultado- "5027cbf32229710460000002"

Concluyendo

Se evidenció la capacidad de extensibilidad que tiene PostgreSQL incluso a otros lenguajes diferentes al que fue creado, python en este caso, se mostraron dos extensiones pgpymongo y pgpycouch con funciones en plpython para interactuar con las bases de datos NoSQL documentales MongoDB y CouchDB respectivamente, presentando las funciones de cada una y con ejemplos que demuestran su utilización.

---

[i] Most Important Software Development Trends for 2012, as Voted by QCon London Attendees, Disponible: http://www.infoq.com/news/2012/03/top-technologies-qcon-london

ii Von der Weth. (2012) C.; Datta, A. Multiterm Keyword Search in NoSQL Systems, Internet Computing, IEEE Volume: 16, Issue: 1 Digital , pp.34-42.