

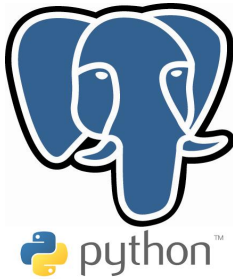
PyReplica

Sistema de replicación simple para
PostgreSQL
programado en
Python

Mariano Reingart (ArPUG/PyAr)



PostgreSQL



El Sistema de Gestión de Bases de Datos de código abierto más potente del mercado

- **Características Básicas:** transacciones, integridad referencial, disparadores, vistas, bloqueos, etc.
- **Características Avanzadas:** concurrencia MVCC, tipos avanzados y del usuario, reglas, herencia, etc.
- **Rendimiento:** similar al resto de los motores
- **Fiabilidad:** versiones estables
- **Soporte:** comunidad, listas de correo, etc.
- **Precio:** \$0.- (incluso usos comerciales) Licencia BSD

Replicación: Conceptos



Maestro/Esclavo:

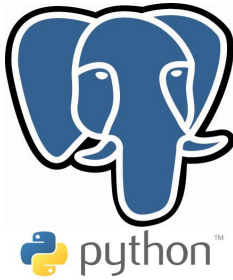
- Maestro: recibe consultas de lectura/escritura
- Esclavo: solo consultas de lectura
- Generalmente cambios asincrónicos (no simultaneo)

Multi-Maestro:

- Solo Maestros
- Generalmente con sincronismo entre servidores
- Sin sincronismo: resolución o prevención de conflictos

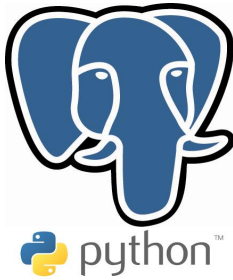
Motivación / Objetivos

PyReplica



- Fácil instalación (scripts, sin compilación)
- Fácil administración (sin comandos)
- Fácil adaptación (filtrar y/o transformar)
- Fácil mantenimiento (código KISS)
- Eficiencia (memoria, red, sin polling)
- Multiplataforma (Windows/Linux)

Alternativas: PgPool-II



Intermediario ente PostgreSQL y las aplicaciones:

- Pool de conexiones
- Replicación Sincrónica
- Balanceo de Carga
- Consultas paralelas

Posibles desventajas:

- Unico punto de falla
- No garantiza replicar funciones correctamente (SERIALs, randoms, CURRENT_TIMESTAMP, etc.)
- Soporte limitado de Autenticación, COPY, etc.
- Degenera o se detiene si se pierde el sincronismo

Alternativas: Slony-I



Replicador Maestro a Múltiples Esclavos mediante disparadores:

- Diseñado para datacenters y sitios de backups (todos los nodos estan generalmente disponibles)
- Esquema avanzado de configuración
- "Pequeño Lenguaje" de administración

Posibles desventajas:

- Instalación, configuración y administración compleja
- Solo un maestro (para escritura)
- Inviabile para conexiones inestables o configuraciones variables
- No replica DDL ni O/s ni SERIAL s

Alternativas: PgCluster / CyberCluster



Replicador Múltiples Maestro y Sincrónico (servidor "especial"):

- Funcionalidad Avanzada
- Replica funciones, LOs, SERIALs, etc.
- Sincronismo de alto rendimiento
- Balance de carga y monitoreo

Posibles desventajas:

- Requiere una versión "parcheada" de PostgreSQL
- Instalación y configuración compleja
- Puede requerir configuraciones avanzadas de hardware

Por que no usar...



PgPool SkyTools Slony-I PgReplicator PgCluster

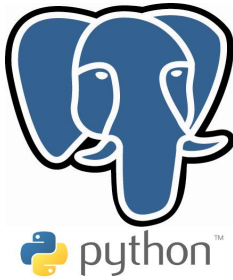
En General:

- Programados en C (compilación, bugs, etc.)
- Complejos (decenas de archivos y miles de líneas)
- Con juego de comandos propios (administración)
- Algnos no funcionan en Windows
- Inestables en condiciones extremas (pocos recursos)

Conclusión: más difíciles de usar y adaptar

Nota: Basado en experiencia y necesidades particulares

Características de PyReplica



- Asincrónico
- Maestro / Esclavo y Multimaestro limitado
- Replicación Condicional
- Detección de conflictos
- Notificaciones vía Email
- Monitoreo de las conexiones (KeepAlive)

- Conexiones directas a los backends
- Sin protocolos especiales (sql textual)
- Si herramientas externas (rsync)
- Protegido con transacciones en dos fases

Características de PyReplica



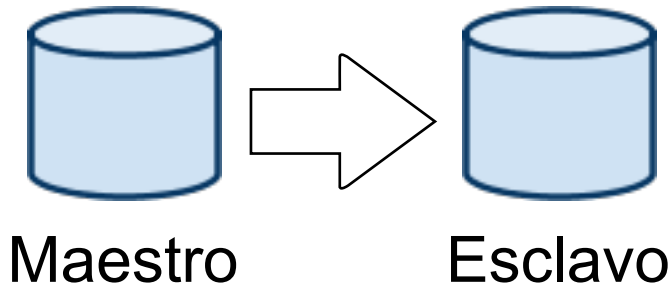
Al igual que slony y otros replicadores basados en disparadores, PyReplica no soporta:

- Replicación de DDL automática (pero puede usarse para propagar órdenes DDL a varios servidores)
- Replicación de LOs (se puede usar ByteA)
- Replicación de secuencias
- Replicación sincrónica (simultanea)
- Resolución de conflictos (se pueden evitar con reglas/disparadores y/o detectarlos)

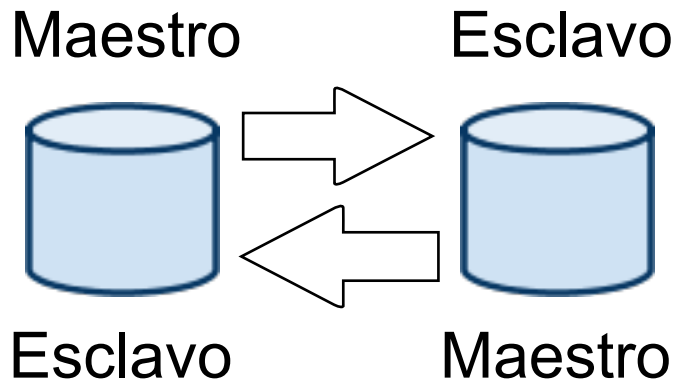
Si soporta:

- Replicación de valores devueltos por funciones de fecha, aleatorias, secuencias, etc.

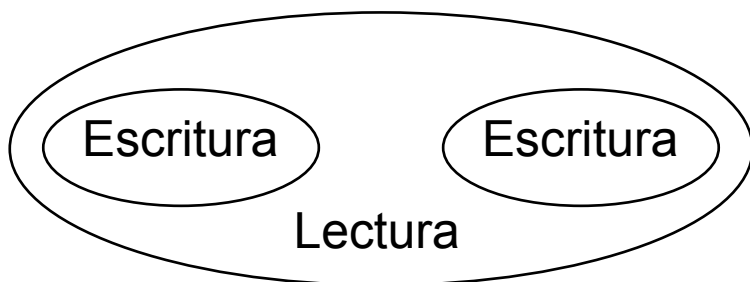
Usos de PyReplica



- Maestro/Esclavo: esclavo de solo lectura
 - Respaldo / Failover
 - Balanceo de carga
 - Datawarehouse / Consolidación



- Multimaestro: ambos son maestro y esclavo (lectura/escritura)
 - Redundancia / Alta Disponibilidad
 - Servidores Remotos (ej. ptos. venta)
 - Servidores Móviles (ej. vendedores)
 - Requiere particionado lógico para evitar conflictos de escritura



Programación de PyReplica



- Lenguaje Python:
 - Código compacto, simple y claro
 - Baterías incluidas: email, hilos, configuración,...
- Simple de programar:
 - 3 archivos fuente ppales., 500 líneas en total aprox.
 - <150 líneas para el disparador y replicador
 - Aprox. 1/8 del tamaño de Slony-I (disparador)
 - Desarrollo en Windows y Linux
 - Sin compilación
 - Conjunto de pruebas automatizadas

Estructura de PyReplica



py_log_trigger:

- Disparador de registro
- Detecta y almacena sentencias de replicación

pyreplica.py:

- Cliente de replicación.
- Ejecuta las sentencias de replicación

daemon.py:

- Demonio residente (configuracion, hilos, etc.)

Disparador de Registro py_log_replica



- Conversión Datos Python •• PostgreSQL
- Detecta cambios y genera SQL:
 - Inserciones (INSERT)
 - Modificaciones (UPDATE y SELECT)
 - Eliminaciones (DELETE)
- Verifica condiciones
- Almacena el SQL en el registro (replica_log)
- Notifica a las replicas

Replicador (cliente y demonio)



- Se conecta a ambas bases de datos
- Escucha notificaciones
- Ejecuta las sentencias SQL registradas

- Maneja las transacciones
- Maneja los hilos (por cada maestro/esclavo)
- Envía email en caso de conflictos
- Registra archivos de logs para seguimiento

Instalación PyReplica (1/3)



- Dependencias: python, plpython, psycopg2

```
apt-get install postgresql-plpython-8.3 python2.5  
python-psycopg2
```

- Obtener/Instalar una copia del PyReplica

```
svn co http://www.nsis.com.ar/svn/pyreplica  
/usr/local/pyreplica
```

Instalación PyReplica (2/3)



- Copiar base al esclavo (dump/restore)

```
createdb somedb -h remote
```

```
pg_dump somedb | psql somedb -h remote
```

- Instalar Replicación en el Maestro:

```
psql somedb -U someuser < master-install.sql
```

Instalación PyReplica (3/3)



Configurar e Iniciar demonio:

- Copiar y editar archivo de configuración

```
cp /usr/local/pyreplica/sample.conf  
  /etc/pyreplica/somedb.conf  
vi /etc/pyreplica/somedb.conf
```

- Iniciar el demonio:

```
/usr/local/pyreplica/daemon.py start
```

Archivo de Configuración



```
[MAIN]
NAME=mydb
# master:
DSN0=dbname=somedb user=someuser password=secret host=remote
# slave:
DSN1=dbname=somedb user=someuser password=secret host=localhost
.
[SMTP]
SERVER=somehost.somewhere.com
START_SUBJECT=[PyReplica] Starting mydb replication
STOP_SUBJECT=[PyReplica] Stopping mydb replication
ERROR_SUBJECT=[PyReplica] Starting mydb replication (ERROR)
WARNING_SUBJECT=[Replica] WARNING on mydb replication
FROM_ADDR=no-reply@somewhere.com
TO_ADDRS=dba@somewhere.com
```

Replicación Condicional



py_log_filter:

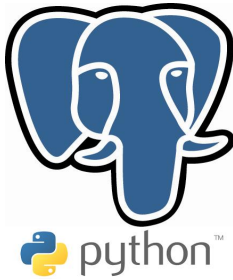
- relname: nombre de la tabla a filtrar
- event: evento INSERT, DELETE o UPDATE
- condition: expresión python verdadero/falso

- diccionarios new y old para comparaciones

Ej:

- `new["campo1"].startswith("Algo")`
- `old["campo2"]==123`

Rendimiento



- Tiempos aproximados (100.000 iteraciones):

```
INSERT INTO test (t,n,f,b) VALUES (random()::text, random(), now(), True);
```

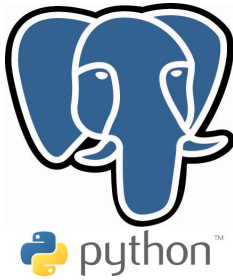
- Sin disparador: 3,5 s
- Disparador de Slony-I (C): 8,8 s
- Disparador prueba (plsql): 11,5 s
- Disparador de PyReplica (plpythonu): 15,1 s

Temas pendientes



- Evitar o Resolver conflictos (multimaestro):
 - Eliminar intervención manual
 - Evitar bloqueos innecesarios
- Mejorar instalación:
 - Paquetes (linux). Autoinstalables (windows)
- Soportar al demonio en windows:
 - Corregir LISTEN en psycopg2 (o usar otro)
 - Ajustar llamadas fork, signal, etc.
- Mejorar demonio y extender pruebas
- Optimizar con txid_snapshot (8.3+)

En testing: PyReplica-Admin



Administrador "visual" de PyReplica:

- Facilita instalación automatizada
- Programado en Python y PythonCard (WX)
- Multiplataforma Windows, Linux, Mac
- Funcionalidad de Respaldo y Restauración
- Monitorización de estado

Yanier (Cuba)

En desarrollo: ALERCE



Replicador Sincrónico para Python:

- Conector compatible DbApi (simil Sequoia)
- Base de datos "Virtual" transparente para la aplicación
- Esquema Maestro - múltiples Esclavos
- Failover simple - cambiar a cualquier maestro (sincrónico!)
- Commit en dos fases para sincronizar bases
- pg_log_replica (disparador) para facilitar el failback (recuperación de esclavos caidos)

¡Gracias!



Direcciones útiles:

- PyReplica:
 - <http://pgfoundry.org/projects/pyreplica/>
- ArPUG: PostgreSQL Argentina
 - www.arpug.com.ar
- PyAr: Python Argentina
 - www.python.org.ar

Copyright 2009 Mariano Reingart - Licencia GPLv3.0

Propaganda...



PyCon Argentina 2009

La primera conferencia en español, conectando a la comunidad Python.

Buenos Aires, 4 y 5 de Septiembre.

<http://ar.pycon.org/>

Los esperamos en PyCon 2009 Argentina:

- La primer conferencia en español del lenguaje Python y temas relacionado
- Septiembre 4 y 5 - Ciudad de Buenos Aires
- Abierto a las comunidades del software libre

ar.pycon.org