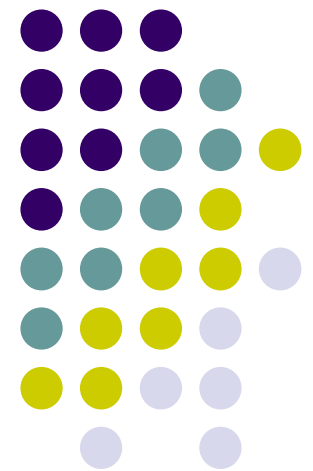
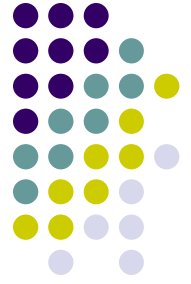


PyReplica

Sistema de replicación simple para
PostgreSQL
programado en
Python



Mariano Reingart



Motivación / Objetivos

- 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)



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)
- Algunos no funcionan en Windows (o difícil instalación)
- Inestables en condiciones extremas (pocos recursos)

Conclusión: más difíciles de usar y adaptar

Nota: Basado en experiencia y necesidades particulares

Características de PyReplica

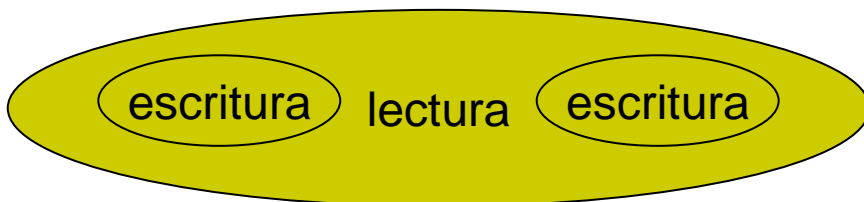
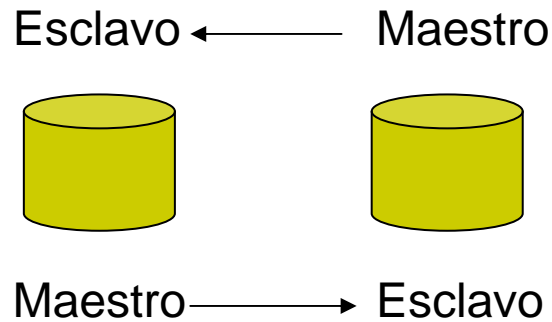
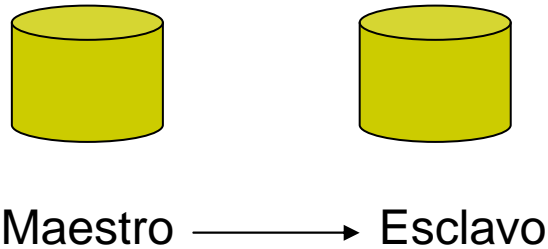


- Asincrónico
- Maestro / Esclavo y Multimaestro limitado
- Detección de conflictos
- Notificaciones vía Email
- Monitoreo de las conexiones (KeepAlive)

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



Usos



- **Maestro/Esclavo: esclavo de solo lectura**
 - Respaldo
 - Balanceo
 - Datawarehouse
- **Multimaestro: ambos son maestro y esclavo (lectura/escritura)**
 - Servidores Remotos (ej. puntos venta)
 - Servidores Móviles (ej. vendedores)
 - Respaldo

Requiere particionado lógico para evitar conflictos de escritura



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:

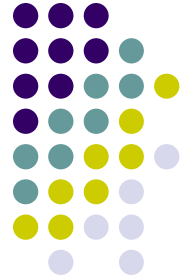
- Demonio residente (configuración, hilos, etc.)



Instalación

- **Dependencias: plpython, python y psycopg2**
`apt-get install postgresql-plpython-8.1 python2.4 python-psycopg2`
- **Obtener/Instalar una copia del PyReplica**
`svn co http://www.nsis.com.ar/svn/pyreplica /usr/local/pyreplica`
- **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`
- **Configurar e Iniciar demonio**
`cp /usr/local/pyreplica/sample.conf /etc/pyreplica/somedb.conf`
`vi /etc/pyreplica/somedb.conf`
`/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
```



Programación

- Lenguaje Python:
 - Código compacto, simple y claro
 - Baterías incluidas: email, webserver, 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



Disparador de Registro

- Conversión Datos Python \leftrightarrow PostgreSQL
- Detecta cambios y genera SQL:
 - Inserciones (INSERT)
 - Modificaciones (UPDATE y SELECT)
 - Eliminaciones (DELETE)
- 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

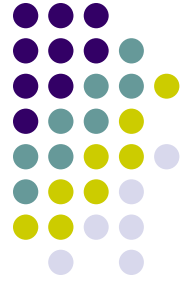


Rendimiento

- 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 bloqueos/intervención manual al producirse un error
 - Modo sincrónico (After Triggers sin demonio?)
- Mejorar instalación:
 - Paquetes (linux). Autoinstalables (windows). Mac?
- Soportar al demonio en windows:
 - Corregir LISTEN en psycopg2 (o usar otro driver)
 - Ajustar llamadas fork, signal, etc. (demonio→servicio)
- Mejorar demonio y extender pruebas
- Futuro (8.3+): usar `txid_current_snapshot()`
 - Agradecimiento a Marko Kreen (skytools) por este último consejo y haber revisado las primeras versiones y comentado sus errores