



# ¿Open Solaris y Postgresql?

Por lo menos, tenés que escucharlos...

**Emanuel Calvo Franco @ AOSUG**



*“Postgresql y Solaris conforman una plataforma de bajo costo para escenarios críticos de mediana a gran escala.”*

# ¿ Que es Postgresql ?

- ♦ Base de Datos Objeto Relacional.
- ♦ Arquitectura Cliente -Servidor.
- ♦ Estable, capaz de correr con escasos recursos, segura y extensible.



# Características de Postgresql

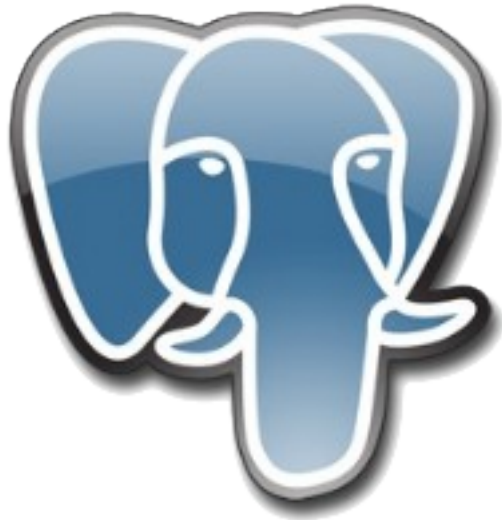
- ♦ MVCC.
- ♦ PITR
- ♦ ACID compliant.
- ♦ Mejorada para Solaris.
- ♦ Standard SQL:2003
- ♦ Funciones criptográficas avanzadas.
- ♦ Diversos lenguajes procedurales.



# Solaris + Pgsql

- ◆ Soporte Dtrace.
- ◆ Escalabilidad para multinucleos y procesadores.
- ◆ Solaris Cluster support and the PostgreSQL HA Agent
- ◆ Solaris Service Manager integration.



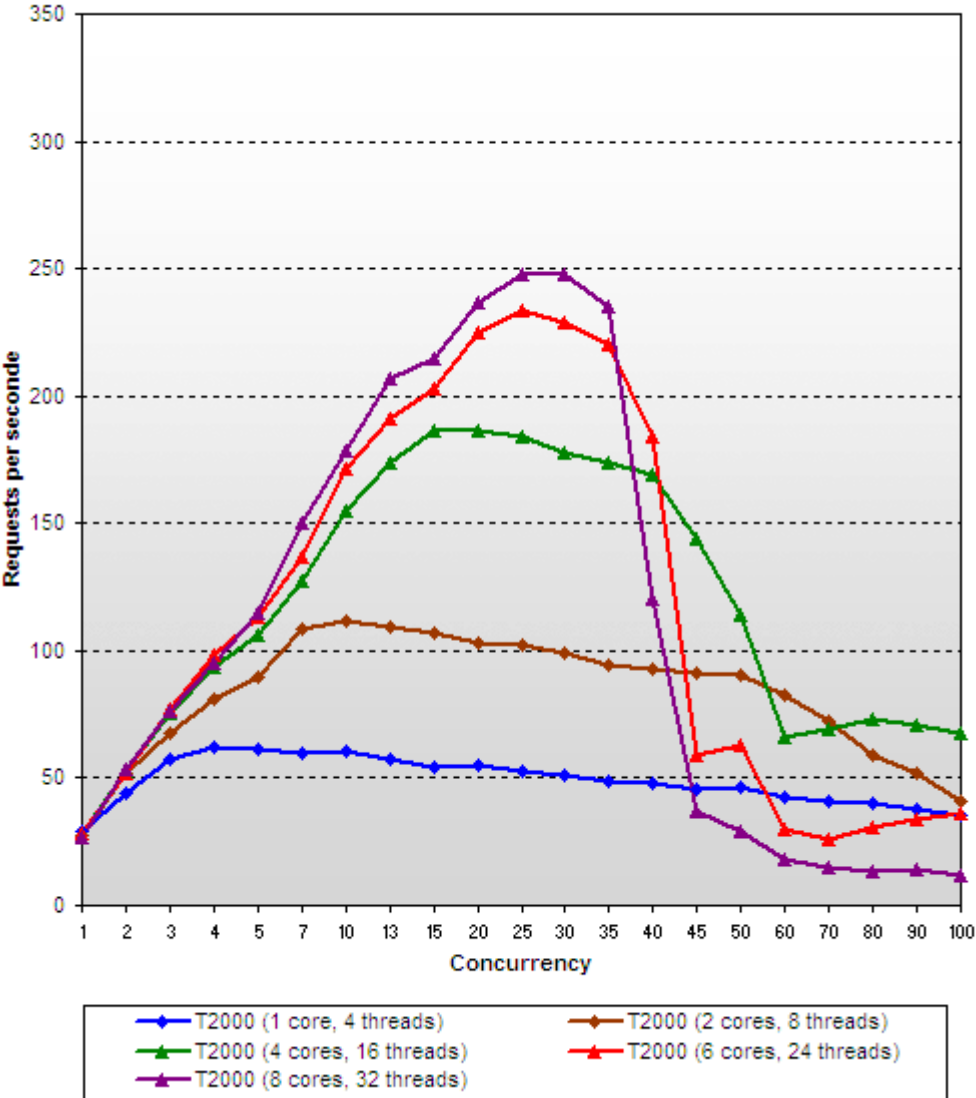


# Benchmarks

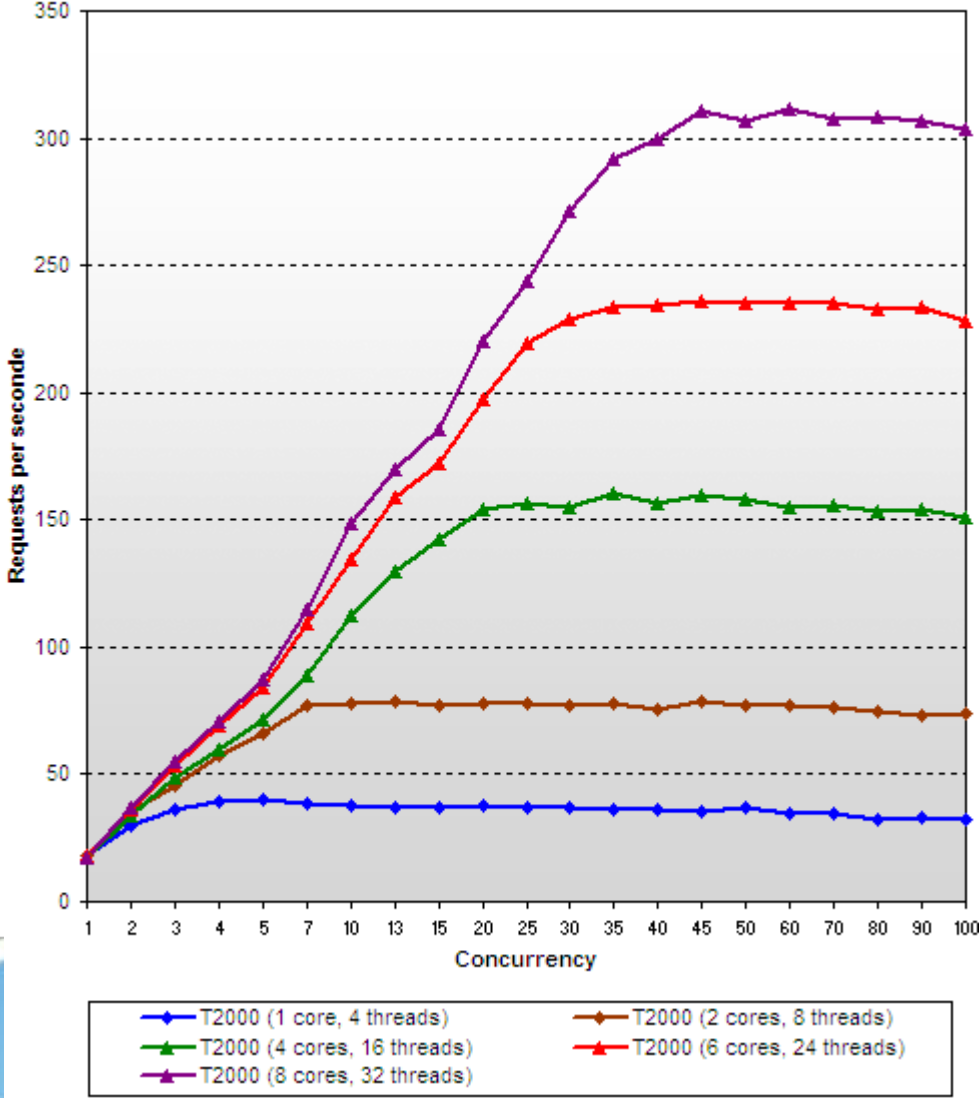


# Benchmarks

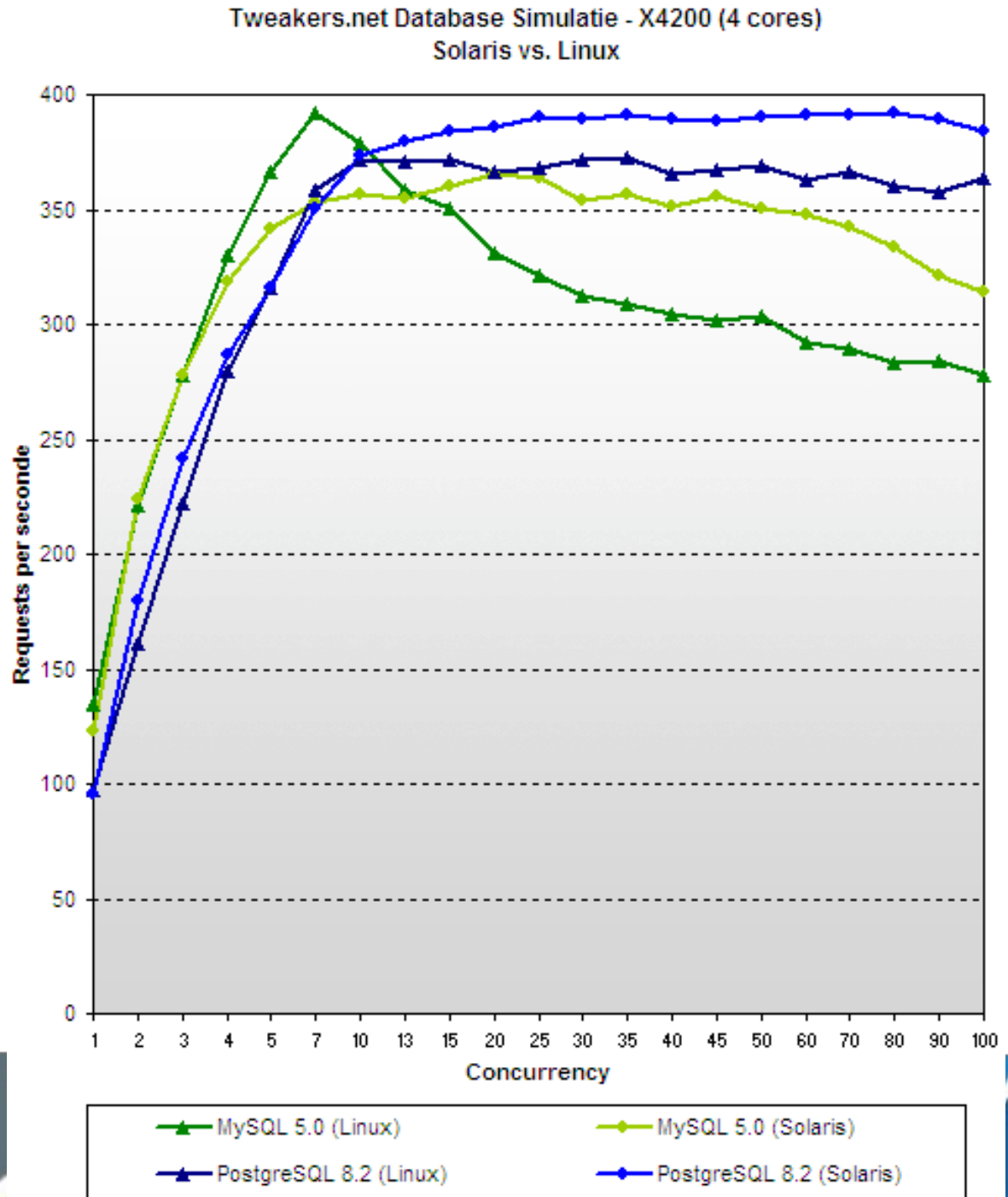
Tweakers.net Database Simulatie - MySQL 5.0.20a schaalgedrag

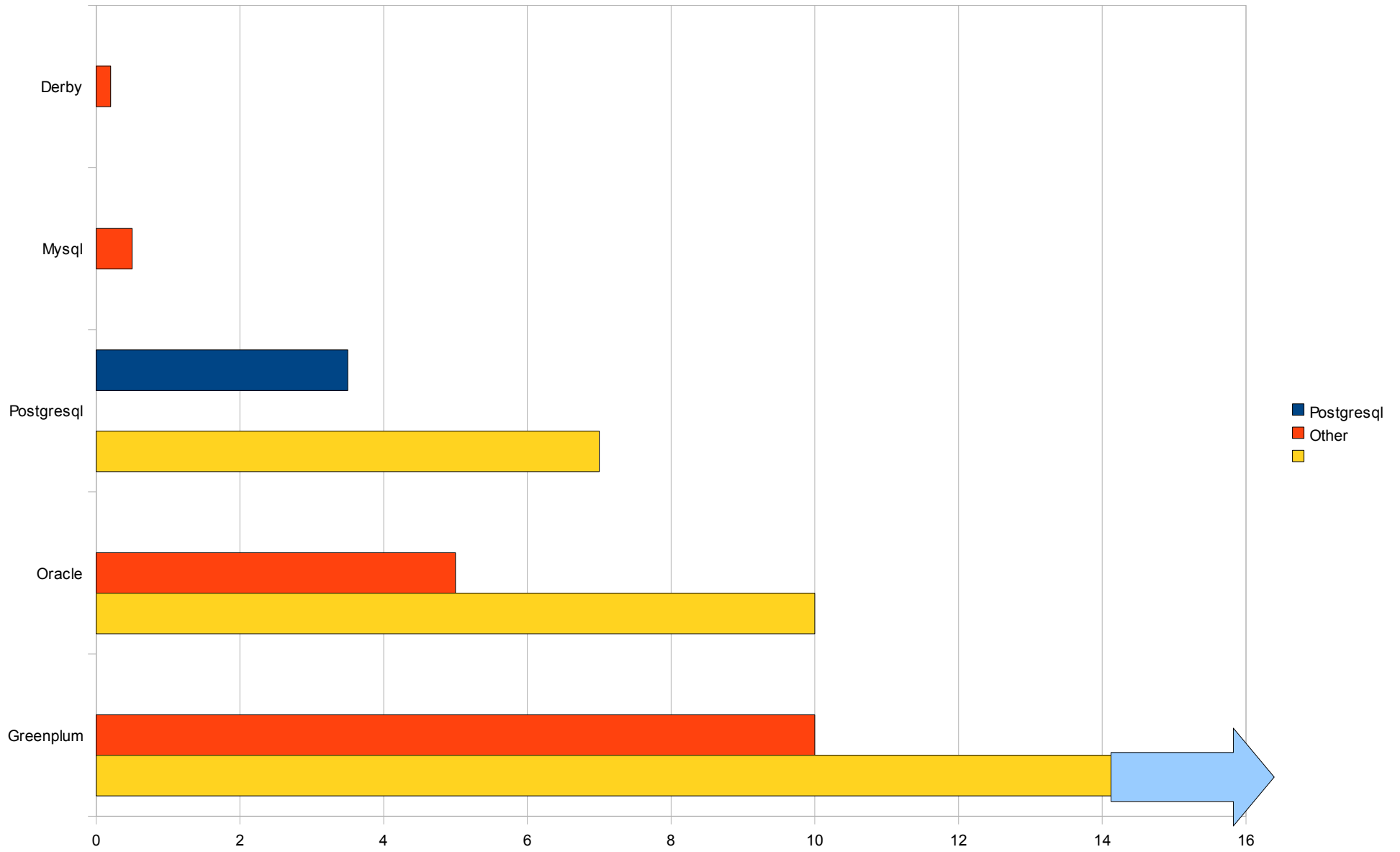


Tweakers.net Database Simulatie - PostgreSQL 8.2 schaalgedrag



# Benchmarks







# Características



# Más beneficios con Solaris

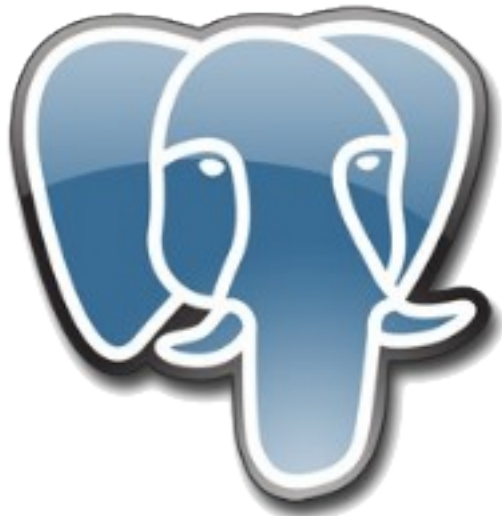
- Sun Cluster support
  - > Sun Cluster agent available
  - > Fully supported
- ZFS + Zones for Hot Standby
  - > ZFS cloning & replication lets you “clone” your database
  - > Zones allows you to “clone” your DB server
  - > OmniTI is using this technique to support a 6TB data warehouse on PostgreSQL



# Zonas ... + seguridad

- Zones are Perfect for “hosted” PostgreSQL
  - > allows 100% isolation of PostgreSQL instances per customer/application
    - > set CPU & RAM limits for each Zone (in SXDE)
      - > PostgreSQL can't do this natively
      - > no per-user limits, no ability to hide the system tables
  - > Being used by Joyent.com to host 100's of customers.





**Instalando**



# Instalar (usando Sun Studio - Homo Simplus Compilatus ® )

```
# ./configure --prefix=/usr/local/pg824 CC=/opt/SUNWspro/bin/cc CFLAGS="-xO3 -xarch=native -xspace -W0,-Lt -W2,-Rcond_elim -Xa -xildoff -xc99=none -xCC" --without-readline --with-perl --with-python --enable-dtrace
```

```
# make
```

```
# make install
```

```
# useradd postgres
```

```
# groupadd postgres
```

```
# useradd -c 'PostgreSQL user' -d /export/home/postgres -g postgres -m -s /bin/bash postgres
```

```
postgres$ $PG_BIN/initdb -D /data -U postgres
```



# Instalar II

```
emanuel@opensolaris_test:~$ pfexec su - postgres
```

```
Sun Microsystems Inc. SunOS 5.11      snv_101b  November 2008
```

```
-bash-3.2$ ls
```

```
bin    data84  include lib    share
```

```
-bash-3.2$ bin/post
```

```
postgres  postmaster
```

```
-bash-3.2$ bin/pg_ctl -D data84/ start
```

```
server starting
```

```
-bash-3.2$ LOG: database system was shut down at 2009-05-18 18:18:51 ART
```

```
LOG: autovacuum launcher started
```

```
LOG: database system is ready to accept connections
```



# Instalar (usando Sun Studio - Homo Paquetus ®)

```
# pkg install SUNWpostgr83server && pkg install SUNWpostgr83client
```

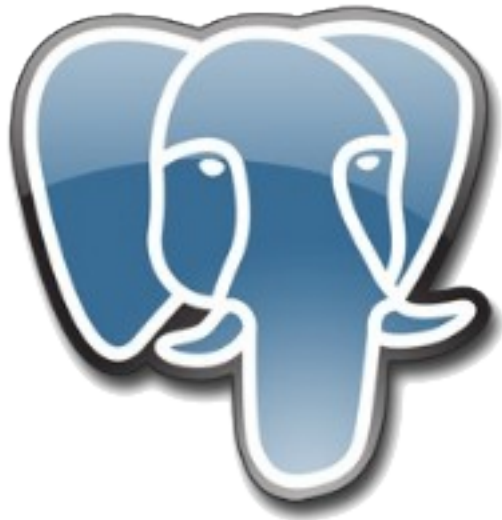
(ultimo en el repo [SUNWpostgr-83-server@8.3.4-0.101](#))

- Levantar el servicio:

```
# svcadm enable svc:/application/database/postgresql_83:default_32bit
```

```
# /usr/sbin/svcadm enable postgresql:version_82
```





**Zonas**

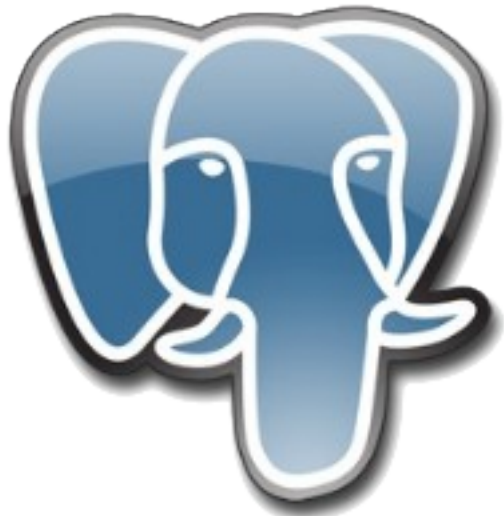


# Ejemplo simple de una zona

```
global# zonecfg -z pg_zone
zonecfg:pg_zone> create
zonecfg:pg_zone> set zonpath=/export/zones/pg_zone
zonecfg:pg_zone> set autoboot=true
zonecfg:pg_zone> add net
zonecfg:pg_zone:net> set address=10.6.222.74/24
zonecfg:pg_zone:net> set physical=ipge0
zonecfg:pg_zone:net> end
zonecfg:pg_zone> verify
zonecfg:pg_zone> commit
zonecfg:pg_zone> exit
```

```
#####instalo
global# zoneadm -z pg_zone install
#####listo
global# zoneadm list -iv
#####booteo
global# zoneadm -z pg_zone boot
#####logueo
global# zlogin -C pg_zone
```





**ZFS**



# Very simple tricks

```
# zfs create -b 8192 rpool/data
```

```
# zfs list
```

```
# zfs set mountpoint=data84 rpool/data84
```

```
# zfs set mountpoint=$PWD/data84 rpool/data84
```

```
# zfs set compression=on rpool/data84
```

```
# zfs list
```

```
# zfs set quota=1g rpool/data84
```

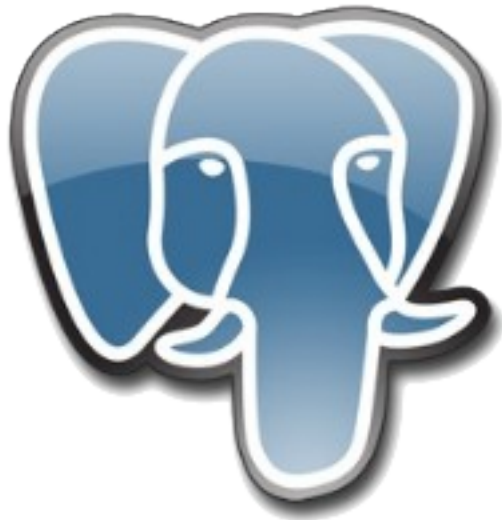
```
# zfs set reservation=100m rpool/data84
```

```
# zfs set sharenfs=rw rpool/data84
```

```
# zpool list
```

```
# zfs list
```





**DTrace**



# Script D

```
#!/usr/sbin/dtrace -s
syscall::entry
/execname== "postgres"/
{
self->ts=timestamp;
}
syscall::return
/execname== "postgres" && self->ts/
{
@a["Count",probefunc] = count();
@b["Time",probefunc] = sum (timestamp - self->ts);
self->ts=0;
}
tick-10sec
{
exit(0);
}
```

# Ejemplo

```
root@opensolaris_test:/usr/local/postgres_home_84# ./syscall.d
```

```
dtrace: script './syscall.d' matched 471 probes
```

```
CPU  ID          FUNCTION:NAME
```

```
0 55976          :tick-10sec
```

```
Count          gtime          47
```

```
Count          pollsys        105
```

```
Count          getpid         109
```

```
Time          gtime         8438387
```

```
Time          getpid        29391576
```

```
Time          pollsys      36537963468
```

# Medición de Transacciones

```
#!/usr/sbin/dtrace -qs

postgresql*:::transaction-start
{
    @startpersec["New"] = count();
}

postgresql*:::transaction-commit
{
    @commitpersec["Commit"] = count();
}

postgresql*:::transaction-abort
{
    @abort["Abort"] = count();
}
```

```
profile:::tick-1s
{
    printf("***** Transactions Per Second
*****\n");

    printf("%20s %15s\n", "Txn Type", "Count");

    printf("=====\n");

    printa("%20s %@15d\n", @startpersec);
    printa("%20s %@15d\n", @commitpersec);
    printa("%20s %@15d\n", @abort);

    printf("\n");

    clear(@startpersec);
    clear(@commitpersec);
    clear(@abort); }
}
```

# Salida

\*\*\*\*\* Transactions Per Second \*\*\*\*\*

Txn Type	Count
----------	-------

=====

New	192
-----	-----

Commit	192
--------	-----

Abort	1
-------	---

\*\*\*\*\* Transactions Per Second \*\*\*\*\*

Txn Type	Count
----------	-------

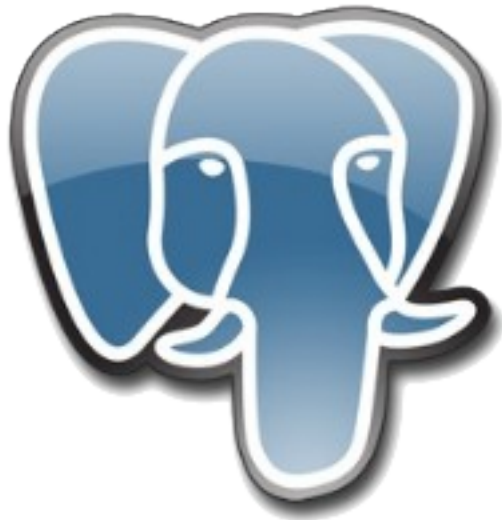
=====

New	175
-----	-----

Commit	172
--------	-----

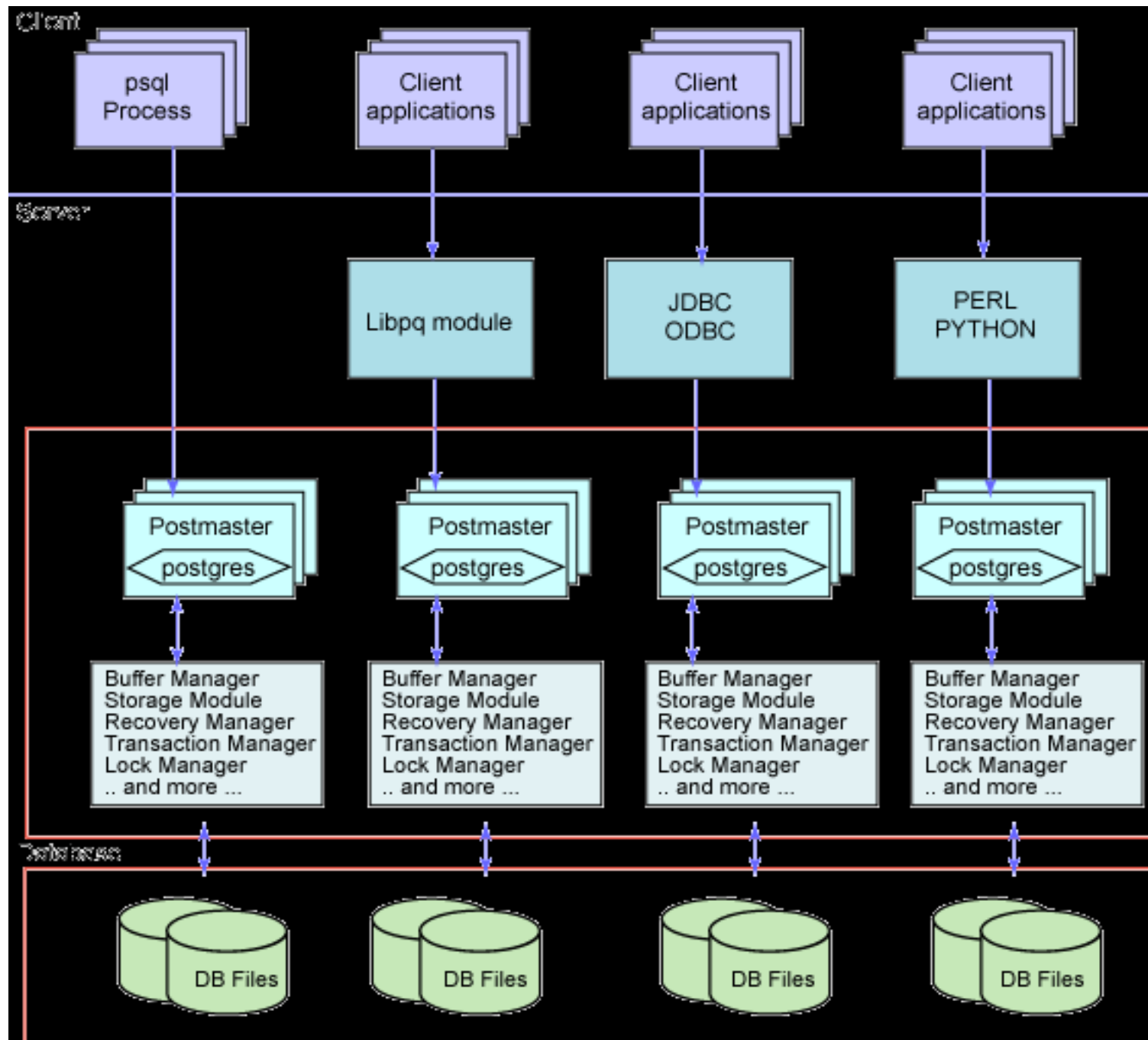
Abort	0
-------	---

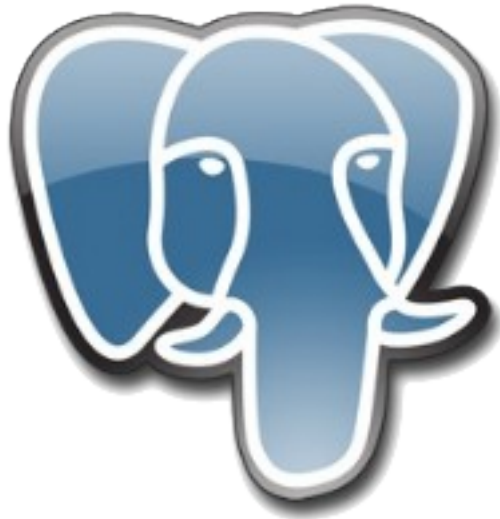
**DELTA:** `select now(),sum(n_tup_ins) as n_tup_ins,sum(n_tup_upd) as n_tup_upd,sum(n_tup_del) as n_tup_del from pg_stat_user_tables;`



# Postgresql Arquitecture

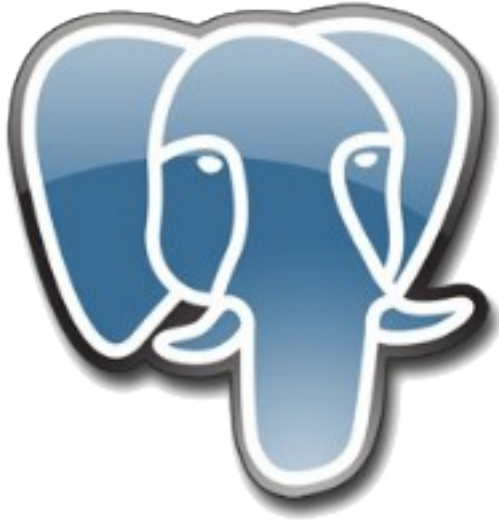






# Another things and Q&A





**Gracias!!**

