Afinamiento de la base de datos

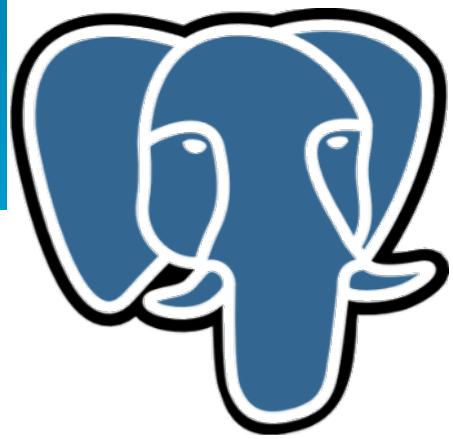
por Jaime Casanova

PostgreSQL

- Consideraciones sobre el sistema de discos.
- Configuraciones de Postgresql.conf relevantes al rendimiento.
  - Controlando el uso de recursos
  - Mitigando el costo de escritura a disco.
  - Indicando a Postgres como escoger los mejores planes.

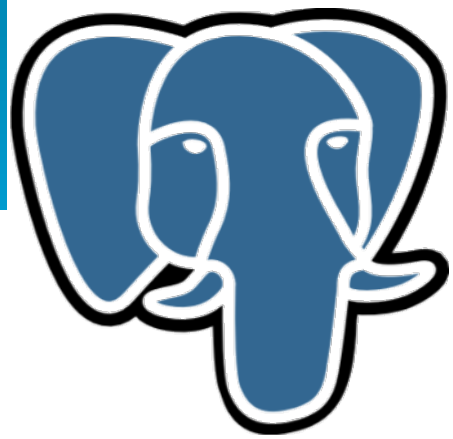
Consideraciones sobre el Sistema de Discos

- El sistema de discos es uno de los cuellos de botella mas comunes en un sistema de base de datos.
    - Aunque aumentar la memoria puede mejorar la situación para bases de datos pequeñas, no siempre es la solución.



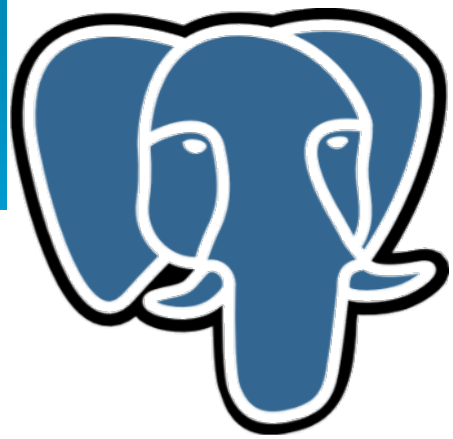
Consideraciones sobre el Sistema de Discos

- Configuración RAID
  - En la comunidad se ha recomendado el nivel 10



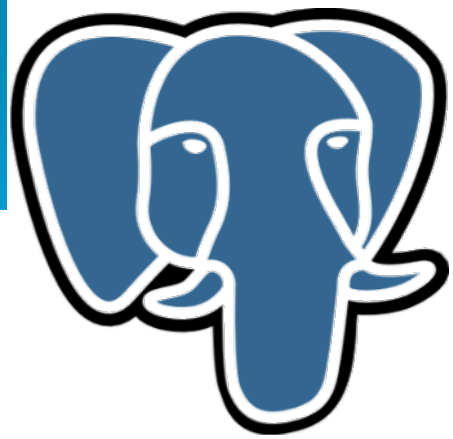
Consideraciones sobre el Sistema de Discos

- Configuración RAID
  - En la comunidad se ha recomendado el nivel 10
- Apagar cache de escritura o usar battery backed array controllers



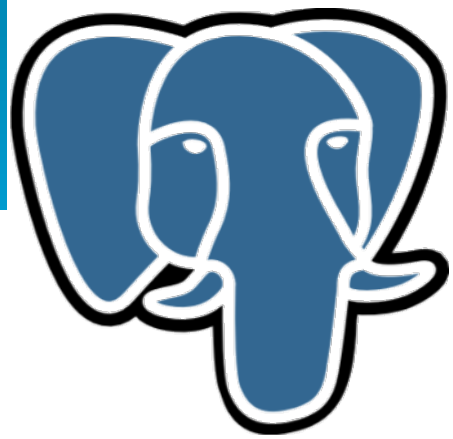
Consideraciones sobre el Sistema de Discos

- Configuración RAID
  - En la comunidad se ha recomendado el nivel 10
- Apagar cache de escritura o usar battery backed array controllers
- Distribución de los discos



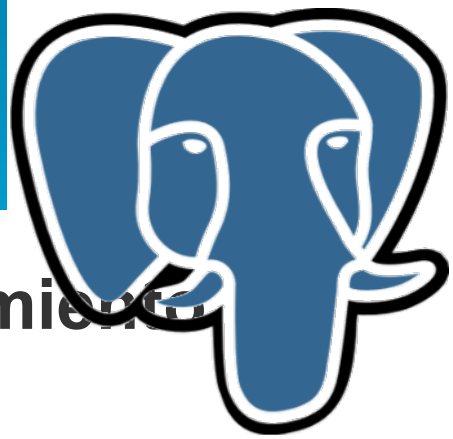
Consideraciones sobre el Sistema de Discos

- Configuración RAID
  - En la comunidad se ha recomendado el nivel 10
- Apagar cache de escritura o usar battery backed array controllers
- Distribución de los discos
  - Poner el WAL ($PGDATADIR/pg_xlog) en un disco aparte



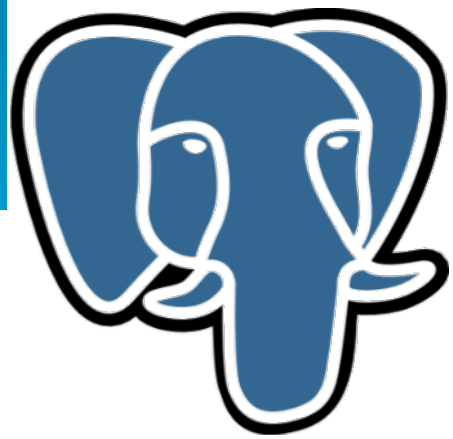
Consideraciones sobre el Sistema de Discos

- Configuración RAID
  - En la comunidad se ha recomendado el nivel 10
- Apagar cache de escritura o usar battery backed array controllers
- Distribución de los discos
  - Poner el WAL ($PGDATADIR/pg_xlog) en un disco aparte
  - Si es posible; disponer de discos separados para poner los datos, índices y archivos temporales.



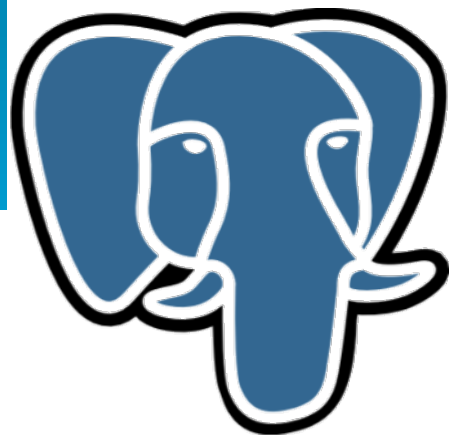
postgresql.conf: Parámetros relevantes al rendimiento

- Controlando el uso de recursos
  - max_connections
  - shared_buffers
  - work_mem
  - max_fsm_pages, max_fsm_relations



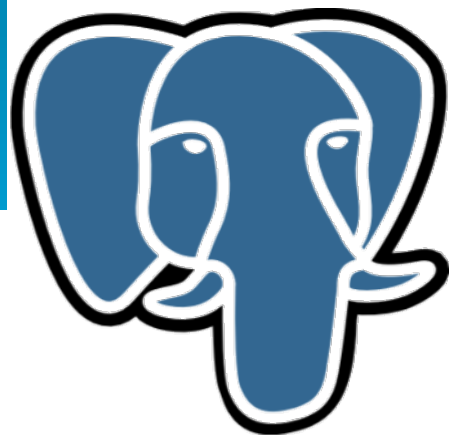
postgresql.conf: max_connections

- Máximo número de conexiones permitidas a la base de datos.
- Debe mantenerse en un valor razonable.
- Al escoger el valor de max_connections se debe considerar el uso de work_mem.
- Se puede usar algún pool de conexiones (PgPool, PgBouncer) para mantener este valor bajo.



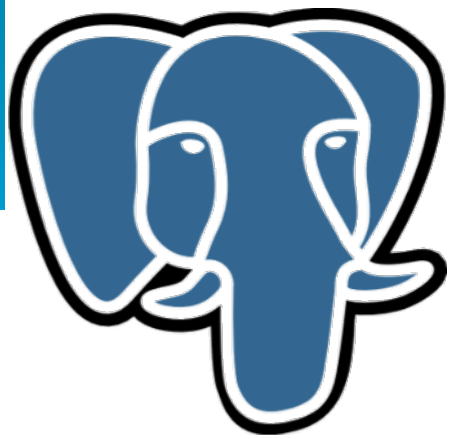
postgresql.conf: shared_buffers

- Este parámetro determina cuanta memoria puede usar PostgreSQL de forma dedicada para mantener datos en cache.
- Un valor razonable (para empezar a probar) es el 25% del total de la memoria RAM disponible.



postgresql.conf: shared_buffers

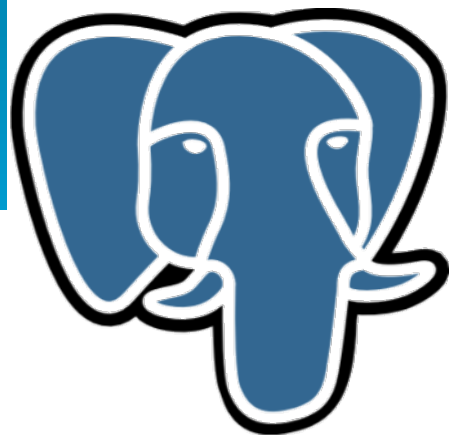
- Si no se indica la unidad se asume 8Kb por cada shared buffer.
  - shared_buffers = 1024 (8Mb)
- Puede necesitarse alterar el valor de SHMMAX.



postgresql.conf: work_mem

- Este parámetro determina cuanta memoria puede usar PostgreSQL en operaciones de ordenamiento y para tablas hash antes de usar archivos temporales.

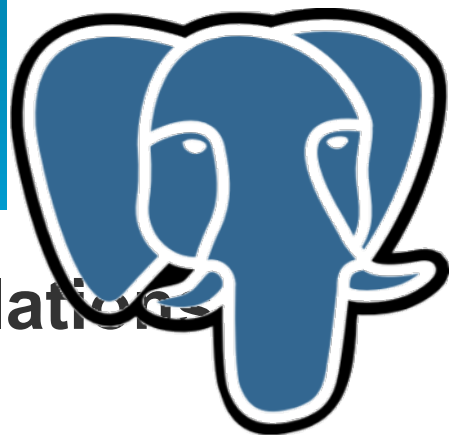
- Este valor se aplica por operación (una consulta realizando varios ordenamientos usará X veces este valor).



postgresql.conf: work_mem

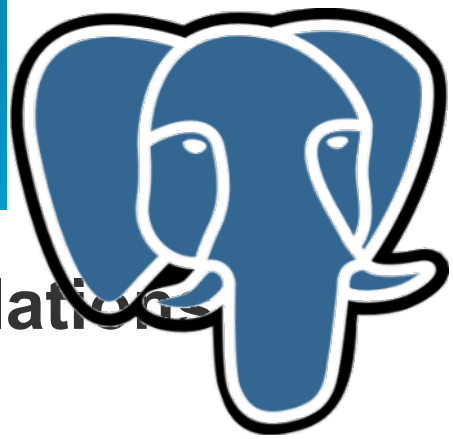
- Al escoger el valor de work_mem se debe considerar el valor de max_connections, un valor razonable puede ser entre el 2-4% del total de la memoria.

- Para mitigar el costo de usar archivos temporales se puede usar el parámetro temp_tablespaces



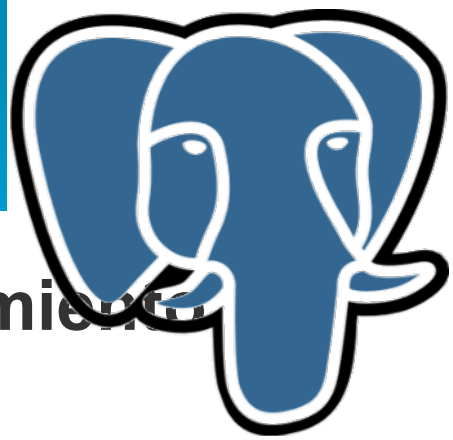
postgresql.conf: max_fsm_pages y max_fsm_relations

- Guardan información sobre el espacio libre (reutilizable) generado debido a la existencia de tuplas muertas en la base de datos.
- El FSM solo se registra durante vacuum.



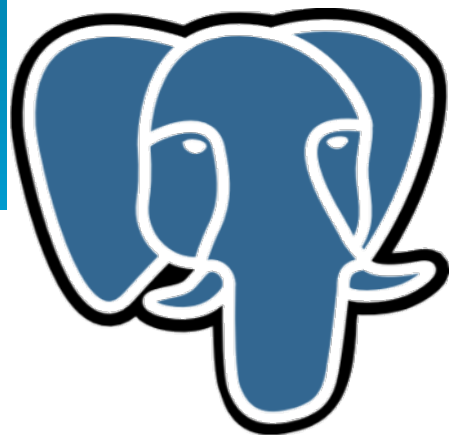
postgresql.conf: max_fsm_pages y max_fsm_relations

- max_fsm_relations debería ser igual al total de tablas e índices en todas las bases de datos de una instalación.
- Se puede usar vacuum verbose para determinar el valor de max_fsm_pages.



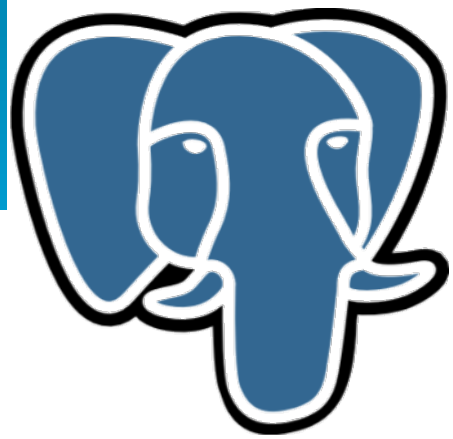
postgresql.conf: Parámetros relevantes al rendimiento

- Mitigando el costo de escritura a disco
  - fsync, synchronous_commit
  - commit_delay, commit_siblings
  - checkpoints
    - checkpoints_segments
    - checkpoints_timeout
    - checkpoints_completion_target



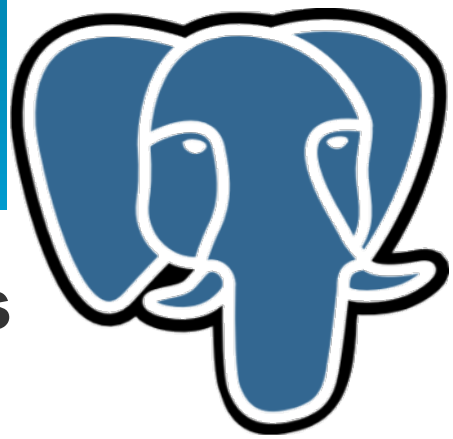
postgresql.conf: fsync, synchronous_commit

- Determina si todas las páginas del WAL deben gabarse a disco antes de que se considere terminada la transacción.
- Setear este valor a OFF podría causar corrupción de datos ante una eventual caída del servidor.
  - En instalaciones de Data Warehouse podría apagarse para aumentar el rendimiento.



postgresql.conf: fsync, synchronous_commit

- synchronous_commit puede proveer la mejora en el rendimiento que fsync=off daría sin el riesgo de corrupción de datos.
  - Se puede setear en cualquier momento.



postgresql.conf: commit_delay, commit_siblings

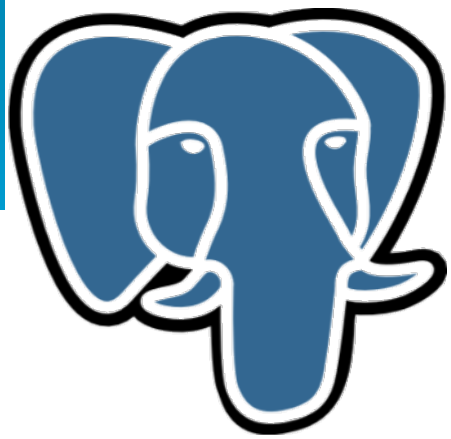
- Se usan para tratar de ejecutar COMMIT simultaneamente en varias transacciones.
- Si existen otros backends activos cuando una transacción está haciendo COMMIT, el servidor espera *commit_delay* microsegundos con la esperanza de que alguna otra de las transacciones (hasta *commit_siblings)* termine.

postgresql.conf: checkpoints

- Pueden ocurrir en 2 circunstancias:
  - Se han llenado todos los segmentos del WAL (checkpoint_segment)
  - Ha transcurrido *chekpoint_timeout* segundos desde el último checkpoint.
- checkpoint_completion_target fue concebido para distribuir uniformemente la ejecución del checkpoint actual durante el período de espera al siguiente.

postgresql.conf: escogiendo los mejores planes

- vacuum_cost_delay
- random_page_cost
- effective_cache_size
- constraint_exclusion

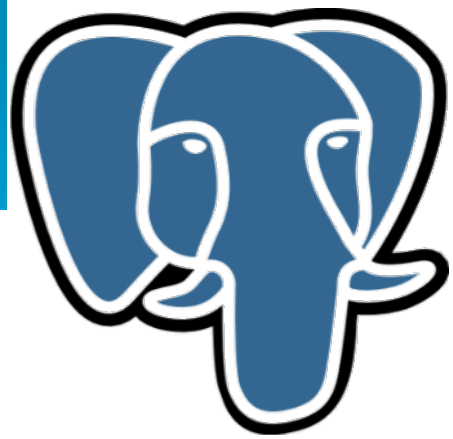


postgresql.conf: vacuum_cost_delay

- Determina el tiempo que vacuum dormirá luego de haber trabajado por un cierto periodo.
  - Hacer uso de este parámetro puede evitar que un vacuum consuma demasiados recursos procesando una tabla muy grande y/o con alta concurrencia.

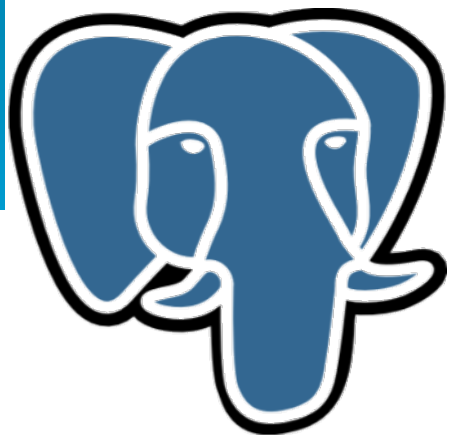
postgresql.conf: random_page_cost

- Determina la forma en que el planeador considera los accesos no secuenciales a disco.
    - Un valor bajo favorecerá el uso de índices; un valor alto, las lecturas secuenciales.
- Aunque es prudente reducir un poco este valor, debe recordarse que el uso de índices tiene un costo y nunca será más rápido que el acceso secuencial (seq_page_cost).



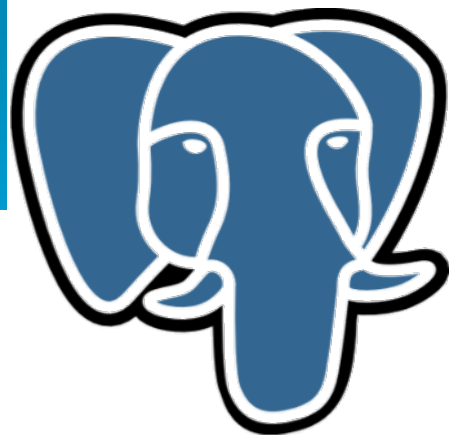
postgresql.conf: effective_cache_size

- Determina la estimación del planeador en cuanto a cuanta memoria hay disponible para mantener un cache para las consultas.
  - Un valor alto favorecerá el uso de índices; un valor bajo, las lecturas secuenciales.



postgresql.conf: effective_cache_size

- Se debería configurar al valor de la memoria que queda luego de considerar shared_buffers, Sistema Operativo y otras aplicaciones.
- Un valor razonable, dependiendo de la instalación, puede ser de 1/2 del total de la memoria.



postgresql.conf: constraint_exclusion

- Permite al planeador considerar los constraints checks en la tabla al determinar el plan de ejecución.
  - Se usa cuando se esta usando un esquema de partición basado en herencia de tablas.

postgresql.conf: constraint_exclusion

- En el siguente ejemplo solo se accesará a la tabla child2000.

```
CREATE TABLE parent(key integer, ...);
CREATE TABLE child1000(
    check (key between 1000 and 1999)
) INHERITS(parent);
CREATE TABLE child2000(
    check (key between 2000 and 2999)
) INHERITS(parent);
...
SELECT * FROM parent WHERE key = 2400;
```