PostgreSQL

PostgreSQL es un servidor de base de datos objeto relacional libre, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional, liberado bajo la licencia BSD. Como muchos otros proyectos open source, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo, dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

Historia

PostgreSQL ha tenido una larga evolución, comenzando con el proyecto Ingres en la Universidad de Berkeley. Este proyecto, liderado por Michael Stonebraker, fue uno de los primeros intentos en implementar un motor de base de datos relacional. Después de haber trabajado un largo tiempo en Ingres y de haber tenido una experiencia comercial con el mismo, Michael decidió volver a la Universidad para trabajar en un nuevo proyecto sobre la experiencia de Ingres, dicho proyecto fue llamado **post-ingres** o simplemente **POSTGRES**.

En proyecto **post-ingres** pretendía resolver los problemas con el modelo de base de datos relacional que habían sido aclarados a comienzos de los años 1980. El principal de estos problemas era la incapacidad del modelo relacional de comprender "tipos", es decir, combinaciones de datos simples que conforman una única unidad. Actualmente estos son llamados objetos. Se esforzaron en introducir la menor cantidad posible de funcionalidades para completar el soporte de tipos. Estas funcionalidades incluían la habilidad de definir tipos, pero también la habilidad de describir relaciones - las cuales hasta ese momento eran ampliamente utilizadas pero mantenidas completamente por el usuario. En **POSTGRES** la base de datos "comprendía" las relaciones y podía obtener información de tablas relacionadas utilizando *reglas*.

El siguiente cuadro representa los hitos más importantes en la vida del proyecto **POSTGRES**.

```
1986 - se publicaron varios papers que describían las bases del sistema.
1988 - ya se contaba con una versión utilizable.
1989 - el grupo liberaba la versión 1 para una pequeña comunidad de usuarios.
1990 - se liberaba la versión 2 la cual tenía prácticamente reescrito el sistema
de reglas.
1991 - liberación de la versión 3, esta añadía la capacidad de multiples motores
de almacenamiento
1993 - crecimiento importante de la comunidad de usuarios, la cual demandaba más
características
1994 - antes de la liberación de la versión 4, el proyecto termina y el grupo se
disuelve.
```

Después de que el proyecto **POSTGRES** terminara, dos graduados de la universidad,Andrew Yu and Jolly Chen, comenzaron a trabajar sobre el código de **POSTGRES**, esto fue posible dado que **POSTGRES** estaba licenciado bajo la BSD, y lo primero que hicieron fue añadir soporte para el lenguaje **SQL** a **POSTGRES**, dado que anteriormente contaba con su propio lenguaje de consultas,

creando así el sistema al cual denominaron **Postgres95**.

Para el año 1996 se unen al proyecto personas ajenas a la Universisdad como Marc Fournier, Bruce Momjian y Vadim B. Mikheev quienes comienzan a trabajar para estabilizar el código de Postgres95.

En el año 1996 deciden cambiar el nombre de Postgres95 de tal modo que refleje la característica del lenguaje SQL y lo terminan llamando **PostgreSQL**.

Con el pasar del tiempo muchos desarrolladores entusiastas de los motores de base de datos se unieron al proyecto y entre todos comenzaron a incorporar muchas características al motor.

```
2000, se comienza a implementar el soporte de Ipv6

2004, PostgreSQL 8.0, adopción en el mundo comercial, se le  califico como la 5ta DBMS mas popular en USA.

2005 Julio, PostgreSQL paso el test de Coverty Inspected  encontrando solo 20 errores en 775,000 lineas de código.

2006 Versión 8.1.4
```

Postsgres ha conseguido los siguientes premios:

2000-2003-2004-2005 Best Database
2004 Best Server Application Award
1999-2002-2004 Best Database

Características:

PostgreSQL esta bajo licencia BSD Berkeley Software Distribution

Esta licencia tiene menos restricciones en comparación con otras como la GPL estando muy cercana al dominio público. La licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre.

- **Libertad Cero:** "usar el programa con cualquier propósito". Es decir, el ejercicio de esta libertad implica que lo podemos utilizar con cualquier fin, ya sea educativo, cultural, comercial, político, social, etc. Esta libertad deriva de que hay ciertas licencias que restringen el uso del software a un determinado propósito, o que prohíben su uso para determinadas actividades.
- **Libertad Uno:** "Estudiar como funciona el programa, y adaptarlo a sus necesidades". Significa que podemos estudiar su funcionamiento (al tener acceso al código fuente) lo que nos va a permitir, entre otras cosas: descubrir funciones ocultas, averiguar como realiza determinada tarea, descubrir que otras posibilidades tiene, que es lo que le falta para hacer algo, etc. El adaptar el programa a mis necesidades implica que puedo suprimirle partes que no me interesan, agregarle partes que considero importantes, copiarle una parte que realiza una tarea y adicionarla a otro programa, etc.
- **Libertad Dos:** "Distribuir copias". Quiere decir que somos libres de redistribuir el

programa, ya sea gratis o con algún costo, ya sea por email, FTP o en CD, ya sea a una persona o a varias, ya sea a un vecino o a una persona que vive en otro país, etc.

- **Libertad Tres:** "Mejorar el programa, y liberar las mejoras al publico". Es la libertad de hacer mejor el programa, es decir que podemos hacer menores los requerimientos de hardware para funcionar, que tenga mayores prestaciones, que ocupe menos espacio, que tenga menos errores, entre otras modificaciones. El poder liberar las mejoras al publico quiere decir que si realizamos una mejora que permita un requerimiento menor de hardware, o que haga que ocupe menos espacio, soy libre de poder redistribuir ese programa mejorado, o simplemente proponer la mejora en un lugar publico (un foro de noticias, una lista de correo, un sitio Web, un FTP, un canal de Chat).

De igual forma las libertades que otorga la licencia aportan ventajas tales como:

- Ahorros importante al liberarse del pago de licencias y especialmente por la replicación casi gratuita de aplicaciones comunes a toda la administración. El muy bajo costo del software permitirá la ejecución de programas y proyectos cuyos costos actuales los hacen prohibitivos.
- La empresa deja de depender de terceros (a menudo transnacionales) para el diseño, desarrollo y mantenimiento de sus sistemas de información, retomando el control total de sus procesos, en particular de los procesos críticos.
- El acceso al código fuente, la libertad de inspeccionar el funcionamiento del software, la libertad de decidir la manera en que almacenan los datos y la posibilidad de modificar cualquiera de estos aspectos queda en manos de la empresa, lo cual le permite el control total de la información.
- El software libre realizado por comunidades está sometido a la inspección de un importante número de personas, este número de verificadores es mucho mayor que el del software propietario. Estas personas identifican los problemas, los resuelven, y comparten las soluciones con los demás. Por tal razón los programas libres de las comunidades gozan de gran confiabilidad y estabilidad.
- La información que la empresa maneja generalmente es importante y/o confidencial, puede ser muy peligroso que esta información caiga en manos incorrectas. Por esta razón es imprescindible que la empresa pueda verificar que su software no tenga puertas de entrada traseras, voluntarias o accidentales, y que pueda cerrarlas en caso de encontrarlas; tal control sólo es posible con el software libre.

Fortalezas y Debilidades de Postgres (8.2.x)

PostgreSQL proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB2 u Oracle. La siguiente es una breve lista de algunas de esas características, a partir de PostgreSQL 7.1.x.

- **DBMS Objeto-Relacional:** PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transactions, optimización de consultas, herencia, y arrays.

- **Herencia** :Las tablas puedes ser configuradas para heredar características de una tabla padre. Los datos son compartidos entre las tablas padre e hija(s) . Las tuplas insertadas o eliminadas en la tabla hija serán insertadas o eliminadas en la tabla padre respectivamente.

- **Altamente_Extensible:** PostgreSQL soporta operadores, funcionales métodos de acceso y tipos de datos definidos por el usuario.

- **Soporte_SQL_Comprensivo:** PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.

- **Integridad Referencial:** PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

- **API Flexible:** La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike.

- **Lenguajes Procedurales:** PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

- **MVCC:** MVCC, o Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Si alguna vez ha usado algún DBMS con capacidades SQL, tal como MySQL o Access, probablemente habrá notado que hay ocasiones en las una lectura tiene que esperar para acceder a información de la base de datos. La espera está provocada por usuarios que están escribiendo en la base de datos. Resumiendo, el lector está bloqueado por los escritores que están actualizando registros. Mediante el uso de MVCC, PostgreSQL evita este problema por completo. MVCC está considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor. En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.

- **Cliente/Servidor:** PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

- **Write Ahead Logging (WAL):** La característica de PostgreSQL conocida como *Write Ahead Logging* incrementa la dependencia de la base de datosal registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la base de datos. Esto puede ser enormemente beneficioso en el caso de caída, ya que cualesquiera cambios que no fueron escritos en la base de datos pueden ser recuperados usando el dato que fue previamente registrado. Una vez el sistema ha quedado restaurado, un usuario puede continuar trabajando desde el punto en que lo dejó cuando cayó la base de datos.

Postgres es Full ACID compliant (Atomicity, Consistency, Isolation and Durability)

Atomicidad(Indivisible) es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.

Consistencia es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la base de datos.

Aislamiento es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generará ningún tipo de error.

Durabilidad es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema.

Más características:

- Corre en casi todos los principales sistemas operativos : Linux, Unix, BSDs, Mac OS, Beos, Windows, etc. (34)

- Documentación muy bien organizada, pública y libre, con comentarios de los propios usuarios.

- Comunidades muy activas, varias comunidades en castellano.

- Bajo Costo de Propiedad Total (TCO) y rápido Retorno de la Inversión

Inicial (ROI)

- Altamente adaptable a las necesidades del cliente.

Comparativas con mysql

Lo mejor de PostGreSQL ...

Las características positivas que posee este gestor son:

1. Posee una gran escalabilidad. Es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta (en algunos benchmarks se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL).
2. Implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría.
3. Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle.

... y lo peor

Por contra, los mayores inconvenientes que se pueden encontrar a este gestor son:

1. Consume gran cantidad de recursos.
2. Tiene un límite de 8K por fila, aunque se puede aumentar a 32K, con una disminución considerable del rendimiento.
3. Es de 2 a 3 veces más lento que MySQL.

Lo mejor de MySQL ...

Es evidente que la gran mayoría de gente usa este gestor en Internet, por lo que encontrar opiniones favorables no ha resultado en absoluto complicado:

1. Sin lugar a duda, lo mejor de MySQL es su velocidad a la hora de realizar las operaciones, lo que le hace uno de los gestores que ofrecen mayor rendimiento.
2. Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
3. Las utilidades de administración de este gestor son envidiables para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración e instalación.
4. Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está.
5. El conjunto de aplicaciones Apache-PHP-MySQL es uno de los más utilizados en Internet en servicios de foro (Barrapunto.com) y de buscadores de aplicaciones (Freshmeat.net).

... y lo peor

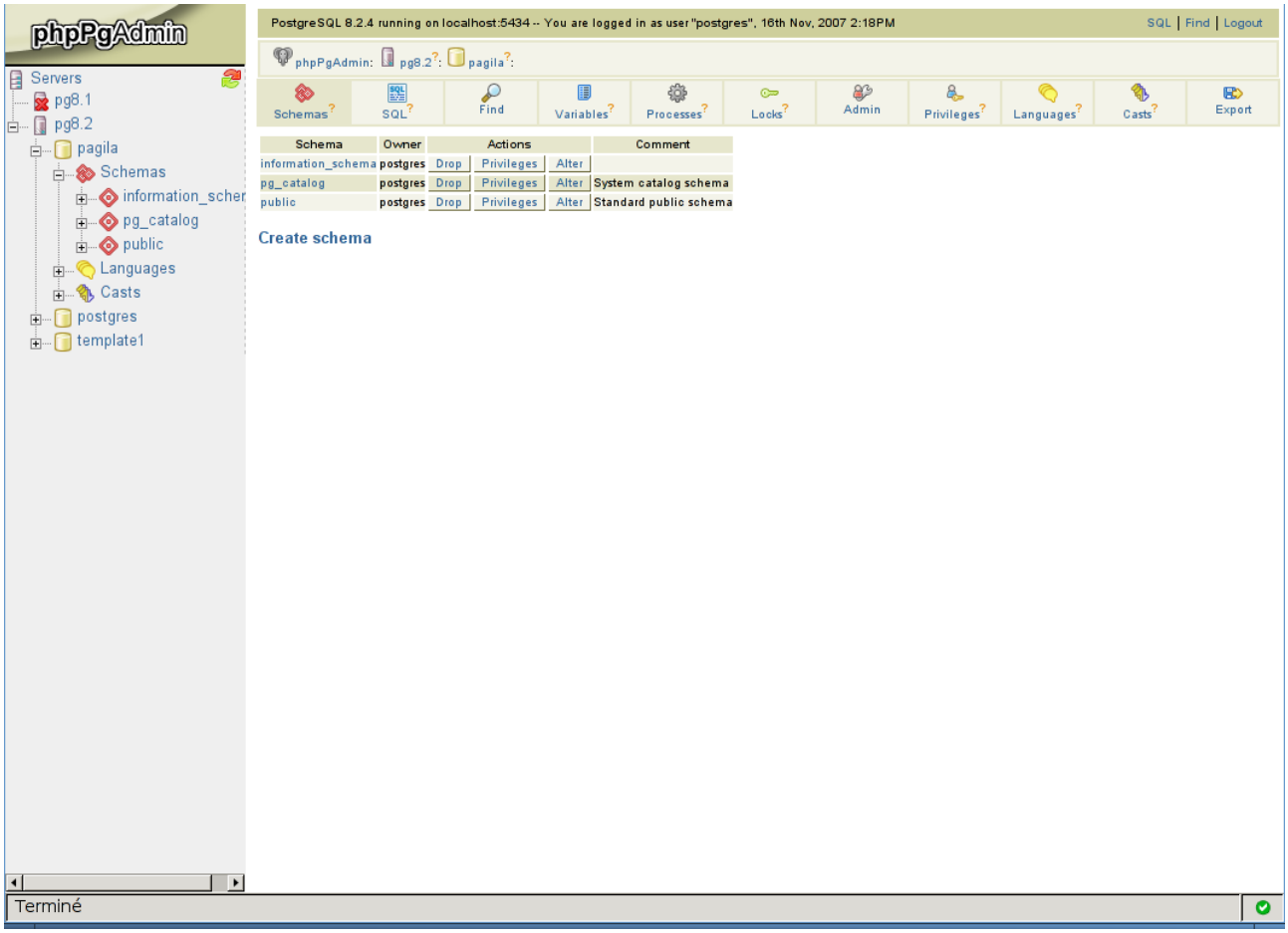
Debido a esta mayor aceptación en Internet, gran parte de los inconvenientes se exponen a continuación, han sido extraídos de comparativas con otras bases de datos:

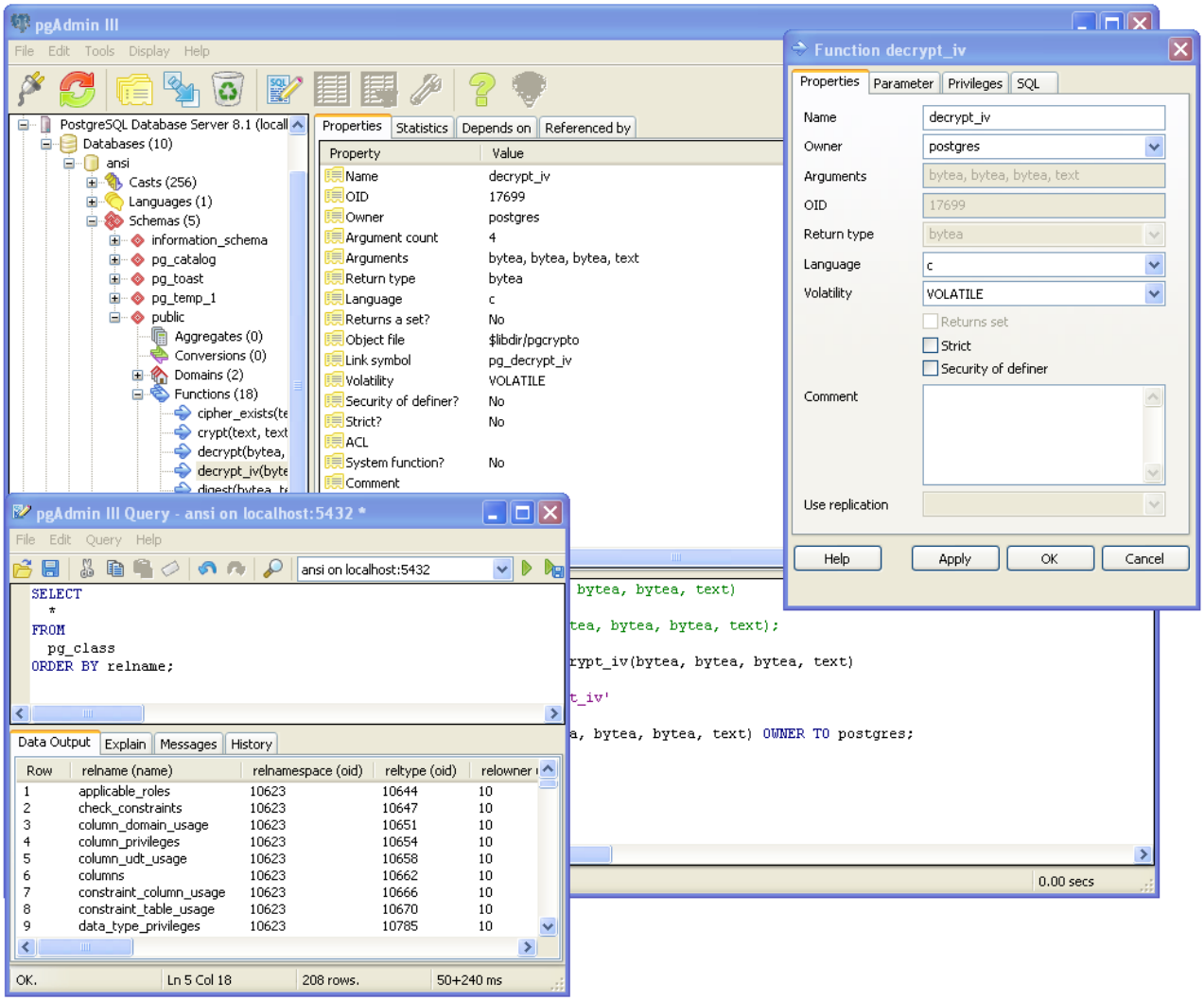
1. Carece de soporte para transacciones, rollback's y subconsultas.
2. El hecho de que no maneje la integridad referencial, hace de este gestor una solución pobre para muchos campos de aplicación, sobre todo para aquellos programadores que provienen de otros gestores que sí que poseen esta característica(version 4 de MySql)
3. No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad.

Administración de postgreSql:

Para administrar postgre pondremos el ejemplo mas común después del uso de la consola, el PhpPgAdmin: fue basado en, pero no comparte código; con phpMyAdmin, que provee las mismas funcionalidades y más a los usuarios del servidor de base de datos MySQL.

Esta herramienta povee de la posibilidad de manejar todos los aspectos de Postgres y ademas pasar los datos obtenidos a otros formatos como XML, XHTML y porsupuesto a SQL.





Además de esta herramienta tambien podemos usar el pgAdmin. PgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia Open Source. Está escrita en C++. PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación Slony-I y mucho más.

Lenguaje PL/PgSQL

Ventajas de usar PL/pgSQL

- **Mayor rendimiento**
- **Soporte SQL**
- **Portabilidad**

Mayor Rendimiento

SQL es el lenguaje que PostgreSQL (y la mayoría del resto de bases de datos relacionales) usa como lenguaje de consultas. Es portable y fácil de aprender. Pero cada estamento SQL debe ser ejecutado individualmente por el servidor de bases de datos.

Esto significa que su aplicación cliente debe enviar cada consulta al servidor de bases de datos, esperar a que se procese, recibir el resultado, realizar alguna computación, y luego enviar otras consultas al servidor. Todo esto incurre en una comunicación entre procesos y también puede sobrecargar la red si su cliente se encuentra en una máquina distinta al servidor de bases de datos.

Con PL/pgSQL puede agrupar un grupo de computaciones y una serie de consultas dentro del servidor de bases de datos, teniendo así la potencia de un lenguaje procedural y la sencilllez de uso del SQL, pero ahorrando una gran cantidad de tiempo porque no tiene la sobrecarga de una comunicación cliente/servidor. Esto puede redundar en un considerable aumento del rendimiento.

Soporte SQL

PL/pgSQL añade a la potencia de un lenguaje procedural la flexibilidad y sencillez del SQL. Con PL/pgSQL puede usar todos los tipos de datos, columnas, operadores y funciones de SQL.

Portabilidad

Debido a que las funciones PL/pgSQL corren dentro de PostgreSQL, estas funciones funcnionarán en cualquier plataforma donde PostgreSQL corra. Así podrá reusar el código y reducir costes de desarrollo.

1.2 Desarrollando en PL/pgSQL

Desarrollar en PL/pgSQL es agradable y rápido, especialmente si ya ha desarrollado con otros lenguajes procedurales de bases de datos, tales como el PL/SQL de Oracle. Dos buenas formas de desarrollar en PL/pgSQL son:

- Usar un editor de texto y recargar el archivo con psql
- Usando la herramienta de usuario de PostgreSQL: PgAccess

Una buena forma de desarrollar en PL/pgSQL es simplemente usar el editor de textos de su elección y, en otra ventana, usar psql (monitor interactivo de PostgreSQL) para cargar las funciones. Si lo está haciendo de ésta forma, es una buena idea escribir la

función usando CREATE OR REPLACE FUNCTION. De esta forma puede recargar el archivo para actualizar la definición de la función. Por ejemplo:

```
CREATE OR REPLACE FUNCTION testfunc(INTEGER) RETURNS INTEGER AS '
....
end; '
LANGUAGE 'plpgsql';
```

Mientras corre psql, puede cargar o recargar la definición de dicha función con

```
\i filename.sql
```

y luego inmediatamente utilizar comandos SQL para testear la función.

Otra buena forma de desarrollar en PL/pgSQL es usando la herramienta administrativa gráfica de PostgreSQL: PgAccess. Esta hace algunas cosas más cómodas , tales como escapar comillas simples, y facilitar la recreación y depuración de funciones.

Estructura de PL/pgSQL

PL/pgSQL es un lenguaje estructurado de bloques. El texto completo de una definición de función debe ser un bloque. Un bloque se define como:

```
[ <<etiqueta>> ]
[ DECLARE
  declaraciones ]
BEGIN
  estamentos
END;
```

Cualquier estamento en la sección de estamentos de un bloque puede ser un subbloque. Los subbloques pueden ser utilizados para agrupaciones lógicas o para lcoalizar variables para un pequeño grupo de estamentos.

Las variables declaradas en la sección de declaraciones que preceden a un bloque son inicializadas a sus valores por defecto cada vez que el bloque es introducido, no sólo una vez por cada llamada a la función. Por ejemplo:

```
CREATE FUNCTION algunafuncion() RETURNS INTEGER AS '
DECLARE
  cantidad INTEGER := 30;
BEGIN
  RAISE NOTICE ''La cantidad aquí es %'',cantidad; -- La cantidad
aquí es 30
  cantidad := 50;
  --
  -- Crea un subbloque
  --
  DECLARE
     cantidad INTEGER := 80;
  BEGIN
     RAISE NOTICE ''La cantidad aquí es %'',cantidad; -- La
cantidad aquí es 80
  END;
  RAISE NOTICE ''La cantidad aquí es %'',cantidad; -- La cantidad
aquí es 50
  RETURN cantidad;
END;
' LANGUAGE 'plpgsql';
```

Es importante no confundir el uso de BEGIN/END para la agrupación de estamentos en PL/pgSQL con los comandos de base de datos para el control de transacciones. Los comandos BEGIN/END de PL/pgSQL son sólo para agrupar; ellos no inician o finalizan una transacción. Los procedimientos de funciones y triggers son siempre ejecutados dentro de una transacción establecida por una consulta externa -no pueden iniciar o validar transacciones, ya que PostgreSQL no tiene transacciones anidadas.

Usuarios Destacados

- .org, .info, .mobi y .aero registros de dominios por Afilias
- La American Chemical Society
- BASF
- IMDB
- Skype

- Sony Online
- U.S. Departamento de Trabajo

Limites de Postgres (8.2.x):

- Máximo de base de datos : ILIMITADO
- Máximo de tamaño de tabla : 32TB
- Máximo de tamaño de registro : 1.6TB
- Máximo de tamaño de campo : 1GB
- Máximo de registros por Tabla : ILIMITADO
- Máximo de campos por tabla : 250 a 1600 (depende de los tipos usados)
- Máximo de índices por tabla : ILIMITADO
- Número de lenguajes en los que se puede programar funciones : aproximadamente 10 (pl/pgsql, pl/java, pl/perl, pl/python, tcl, pl/php, C, C++, Ruby, etc.)
- Métodos de almacenamiento de índices : 4 (B-tree, R- tree, Hash y GisT)

Bibliografia:

- www.wikipedia.es
- www.apesol.org.
- es.geocites.com
- www.postgresSQL.org
- http://gborg.postgresql.org
- http://pgfoundry.org/
- http://phppgadmin.sourceforge.net/
- http://www.netpecos.org/

Versiones del documento :

1.0: **version inicial (lun 21 enero 2008).**

Creada por :

Antonio Aliaga Ibarra

Marcos Agustin Miani Flores