# Guia del Administrador de PostgreSQL

**El equipo de desarrollo de PostgreSQL**

**Editado por**

**Thomas Lockhart**

**Guia del Administrador de PostgreSQL**

por El equipo de desarrollo de PostgreSQL

Editado por Thomas Lockhart

# Tabla de contenidos

# Lista de tablas



# Tabla de figuras



# Tabla de ejemplos

# Summario

Postgres, desarrollada originalmente en el Departamento de Ciencias de la Computación de la Universidad de California en Berkeley, fué pionera en muchos de los conceptos de bases de datos relacionales orientadas a objetos que ahora empiezan a estar disponiblesen algunas bases de datos comerciales. Ofrece suporte al lenguage SQL92/SQL3, integridad de transacciones, y extensibilidad de tipos de datos. PostgreSQL es un descendiente de dominio público y código abierto del código original de Berkeley.

# Capítulo 1. Introducción

Este documento es el manual del programador para el gestor de bases de datos PostgreSQL (http://postgresql.org/), desarrollado inicialmente en la Universidad de California en Berkeley. PostgreSQL se basa en Postgres versión 4.2 (http://s2k-ftp.CS.Berkeley.EDU:8000/postgres/postgres.html). El proyecto Postgres, liderado por el Profesor Michael Stonebraker, ha sido financiado por la Agencia de Proyectos de Investigación de Defensa Avanzados (DARPA), la Oficina de Investigación del Ejército (ARO), la Fundación Nacional de Ciencia (NSF), y ESL, Inc.

## 1.1. Recursos

Este manual está organizado en diferentes partes:

Tutorial

Introduccion para nuevos usuarios. No cubre características avanzadas.

Guia del Usuario

Informacion general para el usuario, incluye comandos y tipos de datos.

Guía del Programador

Información avanzada para programadores de aplicaciones. Incluyendo tipos y extensión de funciones, libreria de interfaces y lo referido al diseño de aplicaciones.

Guia del Administrador

Información sobre instalación y administración. Lista de equipo soportado.

Guia del Desarrollador

Información para desarrolladores de Postgres. Este documento es para aquellas personas que estan contribuyendo al proyecto de Postgres; la información

refererida al desarrollo de aplicaciones aparece en la *Guia del Programador*. Actualmente incluido en la *Guia del Programador*.

Manual de Referencia

Información detallada sobre los comandos. Actualmente incluído en la *Guia del Usuario*.

Ademas de éste manual, hay otros recursos que le servirán de ayuda para la instalacion y el uso de Postgres:

man pages

Las páginas de manual(man pages) contienen mas información sobre los comandos.

FAQs(Preguntas Frequentes)

La sección de Preguntas Frequentes(FAQ) contiene respuestas a preguntas generales y otros asuntos que tienen que ver con la plataforma en que se desarrolle.

LEAME(READMEs)

Los archivos llamados LEAME(README) estan disponibles para algunas contribuciones.

Web Site

El sitio web de Postgres (postgresql.org) contiene información que algunas distribuciones no incluyen. Hay un catálogo llamado mhonarc que contiene el histórico de las listas de correo electrónico. Aquí podrá encontrar bastante información.

Listas de Correo

La lista de correo pgsql-general (mailto:pgsql-general@postgresql.org) (archive (http://www.PostgreSQL.ORG/mhonarc/pgsql-general/)) es un buen lugar para contestar sus preguntas.

Usted!

> Postgres es un producto de código abierto . Como tal, depende de la comunidad de usuarios para su soporte. A medida que empieze a usar Postgres, empezará a depender de otros para que le ayuden, ya sea por medio de documentación o en las listas de correo. Considere contribuir lo que aprenda. Si aprende o descubre algo que no esté documentado, escríbalo y contribuya. Si añade nuevas características al código, hágalas saber.
>
> Aun aquellos con poca o ninguna experiencia pueden proporcionar correcciones y cambios menores a la documentación, lo que es una buena forma de empezar. El pgsql-docs (mailto:pgsql-docs@postgresql.org) (archivo (http://www.PostgreSQL.ORG/mhonarc/pgsql-docs/)) de la lista de correos es un buen lugar para comenzar sus pesquisas.

## 1.2. Terminología

En la documentación siguiente, *sitio* (o *site*) se puede interpretar como la máquina en la que está instalada Postgres. Dado que es posible instalar más de un conjunto de bases de datos Postgres en una misma máquina, este término denota, de forma más precisa, cualquier conjunto concreto de programas binarios y bases de datos de Postgres instalados.

El *superusuario* de Postgres es el usuario llamado `postgres` que es dueño de los ficheros de la bases de datos y binarios de Postgres. Como superusuario de la base de datos, no le es aplicable ninguno de los mecanismos de protección y puede acceder a cualquiera de los datos de forma arbitraria. Además, al superusuario de Postgres se le permite ejecutar programas de soporte que generalmente no están disponibles para todos los usuarios. Tenga en cuenta que el superusuario de Postgres *no* es el mismo que el superusuario de Unix (que es conocido como *root*). El superusuario debería tener un identificador de usuario (*UID*) distinto de cero por razones de seguridad.

El *administrador de la base de datos* (*database administrator*) o DBA, es la persona

responsable de instalar Postgres con mecanismos para hacer cumplir una política de seguridad para un site. El DBA puede añadir nuevos usuarios por el método descrito más adelante y mantener un conjunto de bases de datos plantilla para usar concreatedb.

El postmaster es el proceso que actúa como una puerta de control (clearing-house) para las peticiones al sistema Postgres. Las aplicaciones frontend se conectan al postmaster, que mantiene registros de los errores del sistema y de la comunicación entre los procesos backend. El postmaster puede aceptar varios argumentos desde la línea de órdenes para poner a punto su comportamiento. Sin embargo, el proporcionar argumentos es necesario sólo si se intenta trabajar con varios sitios o con uno que no se ejecuta a la manera por defecto.

El backend de Postgres (el programa ejecutable postgres real) lo puede ejecutar el superusuario directamente desde el intérprete de órdenes de usuario de Postgres (con el nombre de la base de datos como un argumento). Sin embargo, hacer esto elimina el buffer pool compartido y bloquea la tabla asociada con un postmaster/sitio, por ello esto no está recomendado en un sitio multiusuario.

## 1.3. Notación

"..." o `/usr/local/pgsql/` delante de un nombre de fichero se usa para representar el camino (path) al directorio home del superusuario de Postgres.

En la sinopsis, los corchetes ("[" y "]") indican una expresión o palabra clave opcional. Cualquier cosa entre llaves ("{" y "}") y que contenga barras verticales ("|") indica que debe elegir una de las opciones que separan las barras verticales.

En los ejemplos, los paréntesis ("(" y ")") se usan para agrupar expresiones booleanas. "|" es el operador booleano OR.

Los ejemplos mostrarán órdenes ejecutadas desde varias cuentas y programas. Las órdenes ejecutadas desde la cuenta del root estarán precedidas por ">". Las órdenes ejecutadas desde la cuenta del superusuario de Postgres estarán precedidas por "%", mientras que las órdenes ejecutadas desde la cuenta de un usuario sin privilegios estarán precedidas por "$". Las órdenes de SQL estarán precedidas por "=>" o no

estarán precedidas por ningún prompt, dependiendo del contexto.

> **Nota:** En el momento de escribir (Postgres v6.5) la notación de las órdenes flagging (o flojos) no es universalmente estable o congruente en todo el conjunto de la documentación. Por favor, envíe los problemas a la Lista de Correo de la Documentación (o Documentation Mailing List) (mailto:docs@postgresql.org).

## 1.4. Y2K Statement (Informe sobre el efecto 2000)

> **Autor:** Escrito por Thomas Lockhart (mailto:lockhart@alumni.caltech.edu) el 22-10-1998.

El Equipo de Desarrollo Global (o Global Development Team) de PostgreSQL proporciona el árbol de código de software de Postgres como un servicio público, sin garantía y sin responsabilidad por su comportamiento o rendimiento. Sin embargo, en el momento de la escritura:

- El autor de este texto, voluntario en el equipo de soporte de Postgres desde Noviembre de 1996, no tiene constancia de ningún problema en el código de Postgres relacionado con los cambios de fecha en torno al 1 de Enero de 2000 (Y2K).
- El autor de este informe no tiene constancia de la existencia de informes sobre el problema del efecto 2000 no cubiertos en las pruebas de regresión, o en otro campo de uso, sobre versiones de Postgres recientes o de la versión actual. Podríamos haber esperado oír algo sobre problemas si existiesen, dada la base que hay instalada y dada la participación activa de los usuarios en las listas de correo de soporte.

- Por lo que el autor sabe, las suposiciones que Postgres hace sobre las fechas que se escriben usando dos números para el año están documentadas en la Guía del Usuario (http://www.postgresql.org/docs/user/datatype.htm) en el capítulo de los tipos de datos. Para años escritos con dos números, la transición significativa es 1970, no el año 2000; ej. “70-01-01” se interpreta como “1970-01-01”, mientras que “69-01-01” se interpreta como “2069-01-01”.
- Los problemas relativos al efecto 2000 en el SO (sistema operativo) sobre el que esté instalado Postgres relacionados con la obtención de "la fecha actual" se pueden propagar y llegar a parecer problemas sobre el efecto 2000 producidos por Postgres.

Diríjase a The Gnu Project (http://www.gnu.org/software/year2000.html) y a The Perl Institute (http://language.perl.com/news/y2k.html) para leer una discusión más profunda sobre el asunto del efecto 2000, particularmente en lo que tiene que ver con el open source o código abierto, código por el que no hay que pagar.

## 1.5. Copyrights y Marcas Registradas

*La traducción de los textos de copyright se presenta aquí únicamente a modo de aclaración y no ha sido aprobada por sus autores originales. Los únicos textos de copyright, garantías, derechos y demás legalismos que tienen validez son los originales en inglés o una traducción aprobada por los autores y/o sus representantes legales. .*

documentación, incluso si la Universidad ha sido notificada de la posibilidad de tales daños.

La Universidad de California rehusa específicamente ofrecer cualquier garantia, incluyendo, pero no limitada únicamente a, la garantía implícita de comerciabilidad y capacidad para cumplir un determinado propósito. El software que se distribuye aquí se entrega "tal y cual", y la Universidad de California no tiene ninguna obligación de mantenimiento, apoyo, actualización, mejoramiento o modificación.

Unix es una marca registrada de X/Open, Ltd. Sun4, SPARC, SunOS y Solaris son marcas registradas de Sun Microsystems, Inc. DEC, DECstation, Alpha AXP y ULTRIX son marcas registradas de Digital Equipment Corp. PA-RISC y HP-UX son marcas registradas de Hewlett-Packard Co. OSF/1 es marca registrada de Open Software Foundation.

# Capítulo 2. Portes

Este manual describe la versión 6.5 Postgres. La comunidad de desarrollo de Postgresha compilado y probado Postgres en varias plataformas. Visita esta pagina web (http://www.postgresql.org/docs/admin/ports.htm) para tener la última información.

## 2.1. Plataformas actualmente soportadas

En el momento de esta publicación, las siguientes plataformas han sido probadas:

**Tabla 2-1. plataformas soportadas**

| OS | Procesador | Versión | Enviado | Apuntes |
|---|---|---|---|---|
| AIX 4.3.2 | RS6000 | v6.5 | 1999-05-26 | (Andreas Zeug |
| BSDI | x86 | v6.5 | 1999-05-25 | (Bruce Momjia |
| FreeBSD 2.2.x-4.0 | x86 | v6.5 | 1999-05-25 | (Tatsuo Ishii (r |
| DGUX 5.4R4.11 | m88k | v6.3 | 1998-03-01 | v6.4 probablen |
| Digital Unix 4.0 | Alpha | v6.5.3 | 1999-11-04 | (Pedro J. Lobo |
| HPUX | PA-RISC | v6.4 | 1998-10-25 | Both 9.0x and |
| IRIX 6.5.6f | MIPS | v6.5.3 | 2000-02-18 | MIPSPro 7.3.1 |
| linux 2.0.x | Alpha | v6.5.3 | 1999-11-05 | (Ryan Kirkpat |
| linux 2.2.x | arm41 | v6.5.3 | 1999-11-05 | (Mark Knox ( |
| linux 2.2.x/glibc2 | x86 | v6.5.3 | 1999-11-05 | (Lamar Owens |
| linux 2.0.x | MIPS | v6.4 | 1998-12-16 | Cobalt Qube ( |
| linux 2.0.x | Sparc | v6.4 | 1998-10-25 | (Tom Szybist ( |
| linuxPPC 2.1.24 | PPC603e | v6.4 | 1998-10-26 | Powerbook 24 |
| mklinux DR3 | PPC750 | v6.4 | 1998-09-16 | PowerMac 760 |
| NetBSD | arm32 | v6.5 | 1999-04-14 | (Andrew McM |
| NetBSD 1.3.2 | x86 | v6.4 | 1998-10-25 | (Brook Milliga |

| OS | Procesador | Versión | Enviado | Apuntes |
|---|---|---|---|---|
| NetBSD | m68k | v6.4.2 | 1998-12-28 | Mac SE/30 (M |
| NetBSD-current | NS32532 | v6.4 | 1998-10-27 | pequeños prob |
| NetBSD/sparc 1.3H | Sparc | v6.4 | 1998-10-27 | (Tom I Helbek |
| NetBSD 1.3 | VAX | v6.3 | 1998-03-01 | (Tom I Helbek |
| QNX-4.25 | x86 | v6.5.2 | 1999-11-08 | requiere peque |
| SCO OpenServer 5 | x86 | v6.5 | 1999-05-25 | (Andrew Merr |
| SCO UnixWare 7 | x86 | v6.5 | 1999-05-25 | (Andrew Merr |
| Solaris | x86 | v6.4 | 1998-10-28 | (Marc Fournie |
| Solaris 2.6-2.7 | Sparc | v6.4 | 1998-10-28 | (Tom Szybist ( |
| SunOS 4.1.4 | Sparc | v6.3 | 1998-03-01 | Patches submit |
| SVR4 | MIPS | v6.4 | 1998-10-28 | Sin soporte par |
| Windows | x86 | v6.4 | 1999-01-06 | Librerías del la |
| Windows NT | x86 | v6.5 | 1999-05-26 | Trabajando con |

Plataformas listadas para v6.3.x y v6.4.x también trabajan con la v6.5, pero no hemos recibido confirmación explícita de la misma en el momento de la creación de la lista.

> **Nota:** Para Windows NT, el porte de la parte del servidor Postgres se ha conseguido recientemente Postgres Se requiere de la librería Cygnus para compilarlo.

## 2.2. Plataformas no soportadas

Hay pocas plataformas con las cuales se haya intentado y se haya informado que no trabaja con la distribución estándar. Otras listadas aquí no proveen de suficientes

librerías para intentarlo.

**Tabla 2-2. Posiblemente Plataformas incompatibles**

| OS | Procesador | Versión | Enviado | Apuntes |
|---|---|---|---|---|
| MacOS | all | v6.x | 1998-03-01 | Sin librerías co |
| NextStep | x86 | v6.x | 1998-03-01 | Sólo soporte cl |
| SVR4 4.4 | m88k | v6.2.1 | 1998-03-01 | Confirmado co |

# Capítulo 3. Opciones de Configuración

## 3.1. Parámetros de configuración (configure)

El conjunto de parámetros disponibles en configure se puede obtener escribiendo

```
$ ./configure -help
```

Los siguientes parámetros pueden ser de interés para los instaladores:

```
Nombre de directorios y ficheros:
  -prefix=PREFIX         ficheros de instalación independiente de la ar-
quitectura en PREFIX
                          [/usr/local/pgsql]
  -bindir=DIR            ejecutables de usuario en el DIR [EPREFIX/bin]
  -libdir=DIR            librerías de código objeto en el DIR [EPREFIX/lib]
  -includedir=DIR        ficheros de cabeceras C en el in DIR [PREFIX/includ
  -mandir=DIR            documentación man en el DIR [PREFIX/man]
Características y paquetes:
  -disable-FEATURE       no incluir la FEATURE (lo mismo que -enable-
FEATURE=no)
  -enable-FEATURE[=ARG]  incluir FEATURE [ARG=yes]
  -with-PACKAGE[=ARG]    usar PACKAGE [ARG=yes]
  -without-PACKAGE       no usar PACKAGE (lo mismo que -with-PACKAGE=no)
-enable y -with opciones reconocidas:
  -with-template=template                          usar el fiche-
ro plantilla del sistema operativo
                              ver directorio plantilla
  -with-includes=incdir                            sitio donde están los fi-
chero cabecera para tk/tcl, etc. en el DIR
  -with-libs=incdir                           buscar librerías tam-
bién en DIR
```

```
  -with-libraries=libdir                          buscar librerías tam-
bién en DIR
  -enable-locale         activa el soporte local
  -enable-recode         activa el soporte de codificación cirílica
  -with-mb=encoding                        activa el soporte pa-
ra multi-byte
  -with-pgport=portnum                        cambia el puerto de ini-
cio por defecto
  -with-maxbackends=n                        define el número má-
ximo por defecto de procesos servidores
  -with-tcl              construye interfaces Tcl y pgtclsh
  -with-tclconfig=tcldir                          tclConfig.sh y tk-
Config.sh están en DIR
  -with-perl             construye interfaces con Perl
  -with-odbc             construye el paquete del driver ODBC
  -with-odbcinst=odbcdir                          cambia el direc-
torio por defecto de odbcinst.ini
  -enable-cassert        activa los chequeos de afirmación (depurando)
  -with-CC=compiler                        usa el compilador de C especifi
  -with-CXX=compiler                        usa el compilador de C++ espec
  -without-CXX           previene la construcción de código C++
```

Algunos sistemas pueden tener problemas de construcción con algunas características específicas de Postgres. Por ejemplo, sistemas con el compilador de C++ dañado pueden necesitar especificar `-without-CXX` para el proceso de construcción para saltarse la construcción de `libpq++`.

## 3.2. Parámetros de construcción (make)

Muchas parámetros relacionados con la instalación pueden activar en la etapa de construcción de la instalación de Postgres.

En muchos casos, estos parámetros deben colocarse en un fichero, `Makefile.custom`, utilizado para este propósito. La distribución por defecto no contiene este fichero opcional, pero puedes crearlo con el editor de texto que tu elijas. Cuando actualizas una instalación, tu puedes simplemente copiar tu viejo Makefile.custom a la nueva instalación antes que hagas la construcción.

```
make [ variable=value [,...] ]
```

Unas pocas de las muchas variables que puedes especificar son:

POSTGRESDIR

Lo más alto en el árbol de la instalación.

BINDIR

Localización de las aplicaciones y utilidades.

LIBDIR

localización de las librerías, incluyendo las librerías compartidas.

HEADERDIR

Localización de los ficheros include.

ODBCINST

localización de las librerías, incluyendo las librerías compartidas psqlODBC (ODBC) .

Hay otros parámetros opcionales que no se utilizan comúnmente. Muchos de las que listan debajo son apropiadas cuando se estaba desarrollando el código del servidor Postgres .

CFLAGS

Establece los flags para el compilador de C. Debe ser especificado con "+=" para conservar los parámetros por defecto.

YFLAGS

Establece los flags para el parser yacc/bison.Puede usarse `-v` para ayudar a diagnosticar problemas de construcción de un nuevo parser. Debe ser especificado con "+=" para conservar los parámetros por defecto.

USE_TCL

Activa el constructor del interfaces Tcl.

HSTYLE

Páginas HTML estilo DocBook para construir la documentación de partida. No usar a menos que tu estés desarrollando nueva documentación de documentos fuente SGML compatibles con DocBook en `doc/src/sgml/`.

PSTYLE

Páginas estilo DocBook para construir la documentación impresa de partida. No usar a menos que tu estés desarrollando nueva documentación de documentos fuente SGML compatibles con DocBook en `doc/src/sgml/`.

Aquí hay un ejemplo de `Makefile.custom` para un sistema Linux PentiumPro:

```
# Makefile.custom
# Thomas Lockhart 1999-06-01

POSTGRESDIR= /opt/postgres/current
CFLAGS+= -m486 -O2

# documentation

HSTYLE= /home/tgl/SGML/db118.d/docbook/html
```

```
PSTYLE= /home/tgl/SGML/db118.d/docbook/print
```

## 3.3. Soporte Local

> **Nota:** Escrito por Oleg Bartunov. Ver Oleg's web page (http://www.sai.msu.su/~megera/postgres/) para más información sobre el soporte de lengua local y Rusa.

Mientras que estaba en un proyecto para una compañía en Moscú, Rusia, Me encontré con el problema que postgresql no tenia soporte para alfabetos nacionales. Después de mirar posibles soluciones alternativas decidí desarrollar un soporte local yo mismo. No soy un programador en C pero ya había tenido experiencia con la programación local cuando trabajo en perl (depurando) y glimpse. Después de bastantes días sumergido por el árbol de fuente de Postgres Realice muy pocas correcciones en src/backend/utils/adt/varlena.c and src/backend/main/main.c para conseguir lo que quería! Di soporte sólo para LC_CTYPE and LC_COLLATE, pero más tarde otros lo añadieron para LC_MONETARY . Tuve muchos mensajes de la gente a cerca de este parche por eso decidí enviarselo a los desarrolladores y (sorprendentemente) lo incorporaron dentro de la distribución Postgres .

La gente a veces se queja que el soporte local no funciona para ellos. Hay algunos errores comunes:

- No configurar debidamente postgresql antes de compilarlo. Tu debes ejecutar la configuración con la opción –enable-locale para activar el soporte local. No iniciar el entorno correctamente cuando se inicia postmaster. Tu debes definir las variables de entorno LC_CTYPE and LC_COLLATE antes de ejecutar postmaster porque por

detrás coge información local del entorno. Yo uso el siguiente shell script (runpostgres):

```
#!/bin/sh

export LC_CTYPE=koi8-r
export LC_COLLATE=koi8-r
postmaster -B 1024 -S -D/usr/local/pgsql/data/ -o '-Fe'
```

y lo ejecuto en rc.local así

```
/bin/su - postgres -c "/home/postgres/runpostgres"
```

- Un soporte local estropeado en un OS (por ejemplo, el soporte local en libc bajo Linux algunas veces ha sido cambiado y esto ha causado muchos problemas) El más reciente perl tiene también soporte local y si el soporte local es defectuoso **perl -v** da un aviso parecido a esto:

```
8:17[mira]:~/WWW/postgres>setenv LC_CTYPE not_exist
8:18[mira]:~/WWW/postgres>perl -v
perl: warning: Setting locale failed.
perl: warning: Please check that your locale settings:
LC_ALL = (unset),
    LC_CTYPE = "not_exist",
    LANG = (unset)
are supported and installed on your system.
perl: warning: Falling back to the standard locale ("C").
```

- Localización errónea de los ficheros locales! Las posibles localizaciones son: `/usr/lib/locale` (Linux, Solaris), `/usr/share/locale` (Linux), `/usr/lib/nls/loc` (DUX 4.0). Chequea **man locale** para encontrar la localización correcta. Bajo Linux yo hice un enlace simbólico entre `/usr/lib/locale` y `/usr/share/locale` para estar seguro que la próxima libc no estropea mi soporte local.

### 3.3.1. Cuales son los Beneficios?

Tu puedes usar ~* y el operador order by para cadenas que contienen caracteres de alfabetos nacionales. Los usuarios no Ingleses definitivamente lo necesitan. Si tu no quieres usar el soporte local libera la variable USE_LOCALE.

### 3.3.2. Cuales son las Desventajas?

Hay una evidente desventaja si utilizamos el soporte local - que el la velocidad! Por eso, utilízalo sólo si verdaderamente lo necesitas.

## 3.4. Autenticación Kerberos

Kerberos es un sistema de autenticación de seguridad estándar industrial indicado para ser distribuido bajo redes publicas.

### 3.4.1. Disponibilidad

El sistema de autenticacion Kerberos no se distribuye con Postgres. Las versiones de Kerberos están típicamente disponibles como software opcional para los vendedores de sistemas operativos. Además, la distribución del código fuente se pede obtener a través de MIT Project Athena (ftp://athena-dist.mit.edu).

> **Nota:** Puedes desear obtener la versión del MIT si tu proveedor te proporciona una versión, ya que en algunos puntos de venta está deliberadamente capado o proporcionado sin interoperatividad con la versión del MIT.

Los usuarios que se encuentran fuera de los Estados Unidos de América o Canadá esta avisados de que la distribución del código actual de encriptación de Kerberos está

restringida por las leyes de exportación de Gobierno U. S.

Las preguntas acerca de tu Kerberos deben estar dirigidas a tu proveedor o a MIT Project Athena (info-kerberos@athena.mit.edu). Notar que las FAQLs (Lista de preguntas frecuentes) se envían periódicamente a la Kerberos mailing list (mailto:kerberos@ATHENA.MIT.EDU) (envía  to subscribe (mailto:kerberos-request@ATHENA.MIT.EDU)), y USENET news group (news:comp.protocols.kerberos).

## 3.4.2. Instalación

Instalación de Kerberos se trata en detalle en las *Notas de instalación de Kerberos* . Tener cuidado que el fichero llave servidor (the `srvtab` o `keytab`) es de alguna manera accesible para la lectura por la cuenta Postgres .

Postgres y sus clientes pueden ser compilables para usar la versión 4 o la versión 5 de los protocolos Kerberos del MIT configurando la variable KRBVERS en el fichero `src/Makefile.global` al valor apropiado. Puedes también cambiar la localización donde espera encontrar Postgres las librerías asociadas, los ficheros de cabecera y el fichero llave del servidor.

Cuando la compilación se haya completado, Postgres debe registrarse como un servicio Kerberos . Mirar las *Notas de Operaciones de Kerberos* y las páginas del manual relacionadas para más detalle del registro de servicios.

## 3.4.3. Operaciones

Después de la instalación inicial, Postgres debe operar en todos los sentidos como un servicio normal Kerberos . Para más detalles en el uso de la autenticación, ver la *Guía del Usuario PostgreSQL*  en las secciones referentes a postmaster y psql.

En los comentarios de la versión 5 de Kerberos , las siguientes suposiciones están hechas para el usuario y el nombrado de servicio:

- Los nombres de los usuarios principales (anames) se asemen para contener el nombre del usuario actual Unix/Postgres en el primer componente.
- El servicio Postgres se asume para ser tenido como dos componentes, el nombre de servicio y el nombre de host, canonizada en la versión 4 (p.e., con todos los sufijos de dominio borrados).

**Tabla 3-1. Ejemplos de Parámetros de Kerberos**

| Parámetro | Ejemplo |
|---|---|
| user | frew@S2K.ORG |
| user | ao-ki/HOST=miyu.S2K.Berkeley.EDU@S2K.ORG |
| host | postgres_dbms/ucbvax@S2K.ORG |

El soporte de la versión 4 desaparecerá después de algún tiempo tras la puesta en producción de la revisión de la versión 5 del MIT.

# Capítulo 4. Distribución del Sistema

**Figura 4-1. Distribución de los archivos de Postgres**

*Distribución de los archivos de Postgres* muestra cómo se ordena la distribución de Postgres en su disco, cuando es instalada de la forma usual. Para simplificar, asumiremos que Postgres se ha instalado en el directorio `/usr/local/pgsql`. Por lo tanto, siempre que se mencione el directorio `/usr/local/pgsql`, usted deberá sustituirlo por aquél en el que haya instalado Postgres en su sistema. Todas los órdenes de Postgres se instalan en en el directorio `/usr/local/pgsql/bin`. Por consiguiente, debería incluir este directorio en su variable de entorno de path. Si utiliza un intérprete de órdenes derivado del Berkeley C, como csh o tcsh, incluya la línea

```
set path = ( /usr/local/pgsql/bin path )
```

en el archivo .login de su directorio particular (home). Si, en cambio, utiliza un intérprete de órdenes derivado del Bourne, como sh, ksh o bash, deberá agregar las líneas:

```
PATH=/usr/local/pgsql/bin:$PATH
export PATH
```

al archivo .profile ubicado en su directorio de inicio. De aquí en adelante se asumirá que usted ha agregado el directorio `/usr/local/pgsql/bin` a su path. Además, en este documento ser hará referencia frecuentemente a «fijar una variable del intérprete de órdenes» o «fijar una variable de entorno». Si no comprendió totalmente el último párrafo sobre la modificación de la variable PATH, debería consultar la documentación sobre el funcionamiento de su intérprete de órdenes antes de continuar.

Si no ha realizado la instalación con las opciones por defecto, tal vez tenga algo de trabajo extra. Por ejemplo, si el servidor de base de datos está ubicado en una máquina remota, necesitará colocar el nombre del servidor en la variable de entorno PGHOST. Tal vez también necesite fijar la variable de entorno PGPORT. Básicamente, si intenta ejecutar una aplicación y ésta se queja por no poder conectarse con el proceso principal (postmaster), deberá volver un poco sobre sus pasos y asegurarse de que ha establecido todas las variables de entorno necesarias con los valores correctos.

# Capítulo 5. Instalación

Instrucciones para la instalación de PostgreSQL 7.0.

Los comandos que se mencionana acontinuación fueron probados utilizando el shell `bash` sobre la distribución RedHat 5.2 de Linux. A menos que se indique lo conrario, estos comandos serán igualmente aplicables para la mayoría de los sistemas. Comandos como **ps** y **tar** pueden variar entre las distintas plataformas en cuanto a las opciones que deben usarse. *Utilice su sentido común* antes de teclear cualquiera de stos comandos.

Si aún no tiene la distribución de PostgreSQL, puede obteneral en ftp.postgresql.org (ftp://ftp.postgresql.org). Una vez obtenida, desempaquetela utilizando los siguientes comandos:

```
$ gunzip postgresql-7.0.tar.gz
$ tar -xf postgresql-7.0.tar
$ mv postgresql-7.0 /usr/src
```

Nuevamente, estos comandos pueden ser distintos en su sistema.

## 5.1. Antes de comenzar

Para compilar PostgreSQL se requiere la utilidad GNU make. Otras utilidades similares *no funcionarán* en este caso. En los sistemas GNU/Linux, GNU make es a herramienta por defecto. En otros sistemas puede que encuentre que la herramienta GNU make se encuentre instalada con el nombre “gmake”. De aquí en adelante, utilizaremos este nombre para referirnos a GNU make, sin importar cuál sea el nombre que tiene en su sistema. Para probar GNU make teclee:

```
$ gmake -version
```

Si necesita obtener GNU make, lo puede encontrar en ftp://ftp.gnu.org.

En http://www.postgresql.org/docs/admin/ports.htm (http://www.postgresql.org/docs/admin/ports.htm) puede encontrar información actualizada sobre las plataformas soportadas. En general, la mayoría de la plataformas compatibles con Unix que utilicen bibliotecas actualizadas debería ser capaz de ejecutar PostgreSQL. En el subdirectorio `doc` de la distribución existen varios documentos del tipo LEAME y otros con Preguntas de Uso Frecuente (FAQ en inglés) específicos para esa distribución, que pueden resultarle útiles si está teniendo problemas.

La cantidad mínim de memoria que se requiere para ejecutar PostgreSQL es de sólo 8 MB. Sin embargo, se verifica una notable mejora en la velocidad cuando ésta se expante a 96 MB o más. La regla es que, por más memoria que usted instale en su sistema, nunca será demasiada.

Verifique que esista suficiente espacio libre en el disco. Necestará alrededor de 30 MB para los archivos con el código fuente durante la compilación, y unos 5 MB para el directorio de instalación. Una base de datos vacía ocupa aproximadamente 1 MB. De no estar vacía, la base ocupará unas cinco veces el espacio que ocuparía un archivo de texto que contuviera los mismos datos.Si ejecuta las pruebas de regresión, necestiará alrededor de 20 MB extra como espacio temporal.

Para revisar el espacio libre en el disco, utilice:

```
$ df -k
```

Dados los precios actuales de los discos duros, debería considerar consegui uno grande y rápido antes de poner a trabajar una base de datos.

# 5.2. Procedimiento de Instalación

**Instalación de PostgreSQL**

Para una instalación de nuevas, o una actualización desde una versión previa de PostgreSQL:

1. Cree la cuenta de superusuario PostgreSQL. Éste es el usuario bajo el que corre el servidor. Para el uso en producción, deberá usted crear una cuenta de usuario diferente, sin privilegios (habitualmente se utiliza (`postgres`). Si no tiene usted acceso como root, o quiere evitarse este paso, su propia cuenta de usuario es suficiente.

   Ejecutar PostgreSQL como `root`, `bin`, o cualquier otra cuenta con permisos de acceso especiales es un riesto de seguridad, y por ello está permitido.

   No necesitará usted compilar e instalar bajo esta cuenta (aunque puede hacerlo). Se le dirá cuando necesite conectarse como el superusuario de la base de datos.

2. Si no está usted actualizando un sistema existente, salte a paso 4.

   Ahora necesitará usted realizar una copia de seguridad (backup) de su base de datos existente. Si tiene una instalación razonablemente reciente de su base de datos (posterior a la 6.0), conseguirá un vaciado de la misma tecleando:

   ```
   $ pg_dumpall > db.out
   ```

   Si quiere usted conservar las identificación de los objetos (oids), utilice la opción -o al ejecutar pg_dumpall. Sin embargo, a no ser que tenga usted una razón especial para hacer esto, como podría ser utilizar estos identificadores como claves en las tablas, no lo haga.

   Asegurese de utilizar el comando pg_dumpall de la versión que está usted ejecutando actualemente. Pero no utilice el script de 6.0 o el superusuario PostgreSQL tomará la propiedad de *todo*. Si es esta la versión que tiene usted, debería usted utilizar el comando pg_dumpall de una versión 6.x.x posterior. El correspondiente a la versión 7.0 no trabajará en bases de datos anteriores. Si está usted actualizando desde una versión previa a Postgres95 v1.09, deberá usted realizar un backup de su base de datos, instalar Postgres95 v1.09, restaurar su base de datos, y realizar el backup de nuevo.

> **Atención**
>
> Debe usted asegurarse de que su base de datos no se actualiza durante su backup. Si es necesario, pare el postmaster, edite los permisos del fichero `/usr/local/pgsql/data/pg_hba.conf` para permitirle a usted sólo su uso, y relance de nuevo postmaster.

3. Si está usted actualizando un sistema existente, mate ahora el servidor de la base de datos. Teclee:

   ```
   $ ps ax | grep postmaster
   ```

   Esto debería listar los números de proceso para una serie de procesos, de un modo similar a:

   ```
   263  ?  SW   0:00 (postmaster)
   777  p1 S    0:00 grep postmaster
   ```

   Teclee la siguiente línea, reemplazando *`pid`* con el identificador (id) del proceso `postmaster` (263 en el caso anterior). (No utilice el id del proceso "grep postmaster".) (N. del T. también puede hacerlo con la línea

   ```
   $ ps ax | grep postmaster |grep -v grep
   ```

   que le dará la misma salida, pero sin incluir la línea correspondiente al mismo proceso "grep". Fin de la N. del T.)

   ```
   $ kill pid
   ```

   > **Sugerencia:** En sistemas que arrancan PostgreSQL en el durante la secuencia de arranque de la máquina, probáblemente se encontrara un fichero startup que cumplirá el mismo cometido. Por ejemplo, en un sistema Linux RedHat, se debería encontrar que
   >
   > ```
   > $ /etc/rc.d/init.d/postgres.init stop
   > ```
   >
   > funcione correctamente para parar la base.

También deberá trasladar los directorios anteriores a otro sitio. Teclee lo siguiente:

```
$ mv /usr/local/pgsql /usr/local/pgsql.old
```

con sus propias rutas particulares.

4. Configure el código fuente para su sistema. Este es el paso en el que puede usted especificar su ruta de instalación actual para el proceso de construcción, y hacer elecciones sobre lo que tenga usted instalado. Cambiese al subdirectorio `src` y teclee:

   ```
   $ ./configure
   ```

   seguido de todas las opciones que desee usted dar. Para una primera instalación, debería ir todo bien sin dar ninguna. Para obtener una lista completa de las opciones, teclee:

   ```
   ./configure -help
   ```

   Algunas de las opciones que se utilizan más a menudo son:

   –prefix=BASEDIR

   Selecciona un directorio base diferente para la instalación de PostgreSQL. La opción de defecto es `/usr/local/pgsql`.

   –enable-locale

   Si quiere usted utilizar locales.

   –enable-multibyte

   Le permitirá utilizar páginas de caracteres multibyte. Se emplea principalmente para lenguajes como japonés, coreano o chino.

   –with-perl

   Construye la interface Perl. Note por favor que la interface Perl se instalará en el lugar habitual de los módulos Perl (habitualmente bajo

`/usr/lib/perl`), de modo que deberá usted tener acceso root para realizar esta opción correctamente.

–with-odbc

Construye el paquete del driver ODBC.

–with-tcl

Construye las librerías de interface y los programas que requieren Tcl/Tk, incluyendo libpgtcl, pgtclsh y pgtksh.

5. Compile el programa. Teclee:

   ```
   $ gmake
   ```

   El proceso de compilación ocupará entre 10 minutos y una hora, variando en función de la máquina y de las opciones elegidas.

   La última línea que se muestre por el proceso debería ser:

   ```
   All of PostgreSQL is successfully made. Ready to install.
   ```

   Recuerde que “gmake” se puede llamar “make” en su sistema.

6. Instale el programa. Teclee:

   ```
   $ gmake install
   ```

7. Dígale a su sistema como encontrar las nuevas librerías compartidas. Cómo hacer esto varía de unas plataformas a otras. Lo que tiende a trabajar en todas partes es fijar la variable de entorno LD_LIBRARY_PATH:

   ```
   $ LD_LIBRARY_PATH=/usr/local/pgsql/lib
   $ export LD_LIBRARY_PATH
   ```

   Quizá quiera usted poner estas dos líneas en un script de arranque de su shell, como `~/.bash_profile`.

   En algunos sistemas se prefiere el siguiente método, pero debe usted tener acceso root. Edite el fichero `/etc/ld.so.conf` y añada una línea

```
/usr/local/pgsql/lib
```

Y ahora corra el comando **/sbin/ldconfig**.

En la duda, diríjase a las páginas de manual de su sistema. Si recibe usted más tarde un mensaje como

```
./psql: error in loading shared libraries
libpq.so.2.1: cannot open shared object file: No such file or di-
rectory
```

entonces es que todo lo anterior era necesario. Símplemente realice este paso de nuevo.

8. Cree la instalación de la base de datos. Para hacer esto, debe usted conectarse como su cuenta de superusuario de PostgreSQL. No trabajará como root.

   ```
   $ mkdir /usr/local/pgsql/data
   $ chown postgres /usr/local/pgsql/data
   $ su - postgres
   $ /usr/local/pgsql/initdb -D /usr/local/pgsql/data
   ```

   La opción `-D` especifica la situación donde se almacenarán los datos. Puede usted utilizar cualquier otro path, porque no tiene porqué estar bajo el directorio de la instalación. Sólo asegúrese de que la cuenta del superusuario puede escribir en el directorio (o crearlo) antes de arrancar **initdb**. (Si estaba usted siguiendo los pasos de la instalación hasta ahora como el superusuario de PostgreSQL, puede que tenga usted que conectarse como root temporalmente para crear el directorio de datos.)

9. Los pasos previos deberían haberle indicado como arrancar el servidor de la base de datos. Ahagamos ahora:

   ```
   $ /usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data
   ```

   Esto arrancará el servidor en primer término. Para mandarlo a segundo plano, utilice la opción `-S`.

10. Si está usted actualizando desde una instalación anterior, extraiga sus datos con:

    ```
    $ /usr/local/pgsql/bin/psql < db.out
    ```

Y haga también una copia de seguridad de su anterior fichero `pg_hba.conf`, así como de todos los demás ficheros que pueda usted haber creado para la autenticación, tales como ficheros de claves de acceso.

Con todo esto concluímos la instalación propiamente dicha. Para hacer su vida más productiva y agradable, debería mirar los siguientes pasos y sugerencias opcionales.

- La vida será más conveniente si fija usted algunas variables de entorno. Primero, probablemente quiera usted incluir `/usr/local/pgsql/bin` (o su equivalente) en su PATH. Para hacer esto, añada lo siguiente en su fichero de arranque de la shell, tal como `~/.bash_profile` (o `/etc/profile`, si quiere usted que afecte a todos los usuarios):

  ```
  PATH=$PATH:/usr/local/pgsql/bin
  ```

  Aún más, si usted fija la variable PGDATA en el entorno del superusuario de PostgreSQL, podrá usted omitir la opción `-D` para `postmaster` y `initdb`.

- Probáblemente quiera usted instalar la documentación man y HTML. Teclee

  ```
  $ cd /usr/src/pgsql/postgresql-7.0/doc
  $ gmake install
  ```

  Esto instalará ficheros bajo `/usr/local/pgsql/doc` y `/usr/local/pgsql/man`. Para permitir a su sistema encontrar la documentación man, necesitará añadir una línea como la siguiente en el fichero de arranque de la shell:

  ```
  MANPATH=$MANPATH:/usr/local/pgsql/man
  ```

  La documentación está también disponible en formato Postscript. Si tiene usted una impresora Postscript, o tiene su impresora ya preparada para aceptar ficheros Postscript utilizando un filtro de impresión, podrá imprimir la Guía de Usuario simplemente tecleando

  ```
  $ cd /usr/local/pgsql/doc
  $ gunzip -c user.ps.tz | lpr
  ```

Aquí tiene lo que debería hacer usted si tiene Ghostscript en su sistema y está escribiendo en una impresora lasejet:

```
$ alias gshp='gs -sDEVICE=laserjet -r300 -dNOPAUSE'
$ export GS_LIB=/usr/share/ghostscript:/usr/share/ghostscript/fonts
$ gunzip user.ps.gz
$ gshp -sOUTPUTFILE=user.hp user.ps
$ gzip user.ps
$ lpr -l -s -r manpage.hp
```

En caso de dudas, refierase a sus manuales o a su experto local.

Probáblemente debería empezar por leer la Guía del Administrador si es usted completamente nuevo en PostgreSQL, porque contiene información sobre como declarar usuarios y la autenticación a la base de datos.

- Habitualmente, querrá usted modificar su computadora de modo que arranque el servidor de base de datos siempre que se ponga en marcha. Esto no es necesario; el servidor PostgreSQL se puede ejecutar normalmente desde cuentas no privilegiadas sin intervención de root.

  Diferentes sistemas tienen diferentes convenciones para arrancar demonios en el momento de la puesta en marcha, de modo que deberá usted familiarizarse primer con ellos. La mayoría de los sitemas tienen un fichero `/etc/rc.local` o `/etc/rc.d/rc.local` que en la mayoría de los casos no es un mal lugar para situar este comando. Siempre que lo haga, el postmaster deberá ser ejecutado por el superusuario de PostgreSQL (`postgres`) *y no por root* o cualquier otro usuario. Por ello, probáblemente quiera usted formar las líneas de comando iniciandolas con `su -c '...' postgres`.

  Podría ser interesante mantener un registro de las salidas del servidor. Para arrancar de esta forma el servidor, intente:

  ```
  nohup su -c 'postmaster -D /usr/local/pgsql/data > server.log 2>&1' post-
  gres &
  ```

Aquí tenemos algunas otras sugerencias específicas del sistema operativo:

- Edite el fichero rc.local en NetBSD o el fichero rc2.d en SPARC Solaris 2.5.1 para que contenga la siguiente línea:

  ```
  su postgres -c "/usr/local/pgsql/bin/postmaster -S -D /usr/local/pgsql/da
  ```

- En FreeBSD RELEASE-2.2 editE /usr/local/etc/rc.d/pgsql.sh para que contenga las siguientes líneas y hégale chmod 755 y chown root:bin.

  ```
  #!/bin/sh
  [ -x /usr/local/pgsql/bin/postmaster ] && {
      su -l pgsql -c 'exec /usr/local/pgsql/bin/postmaster
          -D/usr/local/pgsql/data
          -S -o -F > /usr/local/pgsql/errlog' &
      echo -n ' pgsql'
  }
  ```

  Usted puede colocar las rupturas de líneas como se muestra antes. La shell es capaz de seguir traduciendo más allá del final de la línea si no se ha terminado una expresión. El exec salva un nivel de shell bajo el proceso postmaster, de modo que el padre es init.

- En Linux RedHat, añada un fichero `/etc/rc.d/init.d/postgres.init` que se basará en el ejemplo que se encuentra en `contrib/linux/`. Y a continuación haga in link simbólico a este fichero desde `/etc/rc.d/rc5.d/S98postgres.init`.

- Ejecute los test de regresión. Los test de regresión son un conjunto de pruebas que verifican que PostgreSQL corre en su máquina en la forma en que los desarrolladores esperan que lo haga. Debería hacer esto definitivamente antes de poner una servidor en uso en producción. El fichero `/usr/src/pgsql/postgresql-7.0/src/test/regress/README` contiene instrucciones detalladas para correr e interpretar los tests de regresión.

# Capítulo 6. Instalacion en Win32

Instrucciones para la instalacion de Postgres version 6.4(libreria de clientes) en Win32.

## 6.1. Construccion de librerias

Los "makefiles" incluidos en Postgres estan escritos para Microsoft Visual C++ y probablemente no trabajen en otros sistemas. Es posible compilar las librerias manualmente en otros casos.

Para contruir las librerias, cambie al directorio `src` y escriba los comandos

```
copy include\config.h.win32 include\config.h
nmake /f win32.mak
```

Esto asume que usted tiene Visual C++ en su camino.

Los siguientes archivos seran creados:

- `interfaces\libpq\Release\libpq.dll` - La libreria dinamica enlazable
- `interfaces\libpq\Release\libpqdll.lib` - Libreria Importada para conectar el programa a libpq.dll
- `interfaces\libpq\Release\libpq.lib` - Version Estatica de la libreria
- `bin\psql\Release\psql.exe` - El monitor interactivo de SQL

## 6.2. Instalacion de las librerias

La unica parte de la libreria que sera realmente instalada es la libreria de `libpq.dll`.

El archivo en la mayoria de los casos debe ser puesto en el directorio `WINNT\SYSTEM32` (o en `WINDOWS\SYSTEM` en un sistema Windows 95/98). Si este archivo es instalado usando un programa de instalacion, debe ser instalado con un examinador de versiones usando VERSIONINFO que esta incluido en el archivo, para asegurar que una version mas reciente de la libreria no sea sobre escrita.

Si planea desarrolar usando libpq en esta maquina, tendra que anadir los directorios `src\include` y `src\interfaces\libpq` al camino en sus configuraciones.

## 6.3. Usando las librerias

Para usar las librerias, debe anadir los archivos`libpqdll.lib` a su projecto (en Visual C++, solo haga un click con el boton derecho en el projecto y escoja anadirlo).

Una vez que esto esta hecho, sera posible usar la libreria como lo haria en cualquiera plataforma basada en UNIX.

# Capítulo 7. Entorno de tiempo de ejecución

En este capítulo se detalla la interacción entre Postgres y el sistema operativo.

## 7.1. Utilizando Postgres desde Unix

Todas las órdenes de Postgres que se ejecutan directamente desde un shell de Unix se encuentran en el directorio ".../bin". Incluir este directorio en su variable de entorno path facilitará mucho la ejecución de los mismos.

Existe una colección de catálogos del sistema en cada servidor. La misma incluye una clase (`pg_user`) que contiene una instancia para cada usuario válido en Postgres. La instancia especifica un conjunto de privilegios sobre Postgres, como la posibilidad de actuar como superusuario en Postgres, la posibilidad de crear/destruir bases de datos y la posibilidad de actualizar los catálogos del sistema. Un usuario de Unix no puede hacer nada con Postgres hasta que se instale una instancia apropiada en dicha clase. Puede encontrarse más información sobre los catálogos del sistema ejecutando consultas sobre las clases apropiadas.

## 7.2. Iniciando postmaster

No le puede suceder nada a una base de datos a menos que esté corriendo el proceso postmaster. Como administrador, hay una serie de cosas que debe recordar antes de iniciar postmaster. Vea las secciones de instalación y configuración en este mismo manual. De todos modos, si ha unstalado Postgres siguiendo las instrucciones de instalación al pie de la letra, lo único que tendrá que hacer para iniciar el proceso postmaster es introducir esta simple órden:

```
% postmaster
```

Ocasionalmente, postmaster escribe mensajes que le serán de ayuda para resolver problemas. Si desea ver los mensajes de diagnóstico de postmaster, puede iniciarlo con la opción -d y redirigir la salida al archivo de registro:

```
% postmaster -d > pm.log 2>&1 &
```

Si no desea ver los mensajes, inícielo de la forma

```
% postmaster -S
```

y postmaster será "S"ilencioso. Observe que al no haber el signo ampersand ("&") al final del último ejemplo, no se ejecuta como proceso de fondo.

## 7.3. Usando pg_options

**Nota:** Contribución de Massimo Dal Zotto (mailto:dz@cs.unitn.it)

El archivo opcional `data/pg_options` contiene opciones usadas por el backend para controlar los mensajes de trazado y otros parámetros ajustables del mismo. El archivo se vuelve a leer cuando el backend recibe la señal SIGHUP, permitiendo cambiar las opciones de tiempo de ejecución al vuelo, sin que sea preciso reiniciar Postgres. En este archivo se pueden incluír opciones de depuración usadas por el paquete de trazado (`backend/utils/misc/trace.c`) o parámetros numéricos usados por el backend para controlar su comportamiento.

Todas las pg_options se inicializan a cero al iniciar el backend. Si se añaden o se modifican opciones, serán leídas por todos los backend que se inicien a continuación. Para que cualquier cambio tome efecto en los backend que están activos, es preciso enviar una señal SIGHUP al postmaster, quien reenviará la señal a todos los backend activos. Se pueden activar los cambios para un backend específico enviándole directamente una señal SIGHUP.

Las pg_options pueden especificarse también con la opción `-T` de Postgres:

```
postgres opciones -T "verbose=2,query,hostlookup-"
```

Las funciones usadas para indicar errores y mensajes de depuración pueden usar la facilidad *syslog(2)*. Los mensajes que se escriben en stdout o stderr incluyen un prefijo con la fecha, la hora y el número de proceso del backend que las genera.

```
#timestamp          #pid    #message
980127.17:52:14.173 [29271] StartTransactionCommand
980127.17:52:14.174 [29271] ProcessUtility: drop table t;
980127.17:52:14.186 [29271] SIIncNumEntries: table is 70% full
980127.17:52:14.186 [29286] Async_NotifyHandler
980127.17:52:14.186 [29286] Waking up sleeping backend process
980127.19:52:14.292 [29286] Async_NotifyFrontEnd
980127.19:52:14.413 [29286] Async_NotifyFrontEnd done
980127.19:52:14.466 [29286] Async_NotifyHandler done
```

Este formato facilita la lectura de los mensajes y permite saber exactamete lo que hace cada backend y en qué momento. También facilita la escritura de scripts awk o perl sencillos que analicen los historiales para detectar errores o problemas en la base de datos o para calcular estadísticas temporales de transaciones.

Los mensajes escritos a través de syslog utilizan la facilidad LOG_LOCAL0. El uso de syslog se controla mediante la opción syslog en pg_options. Por desgracia, muchas

funciones llaman directamente a `printf()` para escribir sus mensajes a stdout o stderr, cuya salida no se puede controlar mediante la opción syslog ni puede incluír fecha y hora. Sería recomendable sustituir todas las llamadas a printf con la macro PRINTF, y todas las escrituras a stderr por la macro EPRINTF, para poder controlar toda la salida de un modo uniforme.

El formato del archivo `pg_options` es como sigue:

```
# comentario
option=valor_entero  # Establece el valor de  option
option                # establece option = 1
option+               # establece option = 1
option-               # establece option = 0
```

Observe que *`keyword`* puede ser una abreviatura de un nombre de opción de los definidos en `backend/utils/misc/trace.c`.

**Ejemplo 7-1. Archivo pg_options**

Por ejemplo, mi archivo pg_options contiene los siguientes valores:

```
verbose=2
query
hostlookup
showportnumber
```

### 7.3.1. Opciones reconocidas

Actualmente están definidas las opciones:

all

Marca de traza global. Los valores permitidos son:

0

Mensajes de trazado activados individualmente

1

Activar todos los mensajes de trazado

-1

Inhibir todos los mensajes de trazado

verbose

Marca de verbosidad. Valores permitidos:

0

Sin mensajes, éste es el valor por omisión.

1

Escribir mensajes de información.

2

Escribir más mensajes de información.

query

Trazar peticiones. Valores permitidos:

0

No escribir la petición.

1

Escribir una versión condensada de la petición en una línea.

4

Escribir la consulta completa.

plan

Escribir el plan de consulta.

parse

Escribir la salida del traductor de consultas.

rewritten

Escribir la consulta reescrita.

parserstats

Escribir las estadísticas del traductor de consultas.

plannerstats

Escribir las estadísticas del planificador.

executorstats

Escribir las estadísticas de ejecución.

shortlocks

: De momento no se usa, pero se precisa para habilitar nuevas características en el futuro.

locks

: Trazar bloqueos.

userlocks

: Trazar bloqueos de usuario.

spinlocks

: Trazar ’spin locks’.

notify

: Trazar funciones de notificación.

malloc

: Sin uso por el momento.

palloc

: Sin uso por el momento.

lock_debug_oidmin

: Minimum relation oid traced by locks.

lock_debug_relid

: oid, if not zero, of relation traced by locks.

lock_read_priority

: Sin uso por el momento.

deadlock_timeout

Temporizador de comprobación de bloqueos circulares..

syslog

Marca de syslog. Valores permitidos:

0

Mensajes a stdout/stderr.

1

Mensajes a stdout/stderr y syslog.

2

Mensajes solamente a syslog.

hostlookup

Habilitar la consulta de nombre de host en ps_status.

showportnumber

Mostrar el número de puerto en ps_status.

notifyunlock

Desbloqueo de pg_listener después de notify.

notifyhack

Borrar tuplas duplicadas de pg_listener.

# Capítulo 8. Seguridad

La seguridad de la base de datos esta implementada en varios niveles:

- Protección de los ficheros de la base de datos. Todos los ficheros almacenados en la base de datos estan protegidos contra escritura por cualquier cuenta que no sea la del superusuario de Postgres.
- Las conexiones de los clientes al servidor de la base de datos estan permitidas, por defecto, únicamente mediante sockets Unix locales y no mendiante sockets TCP/IP. Ha de arrancarse el demonio con la opcion `-i` para permitir la conexion de clientes no locales.
- Las conexiones de los clientes se pueden restringir por dirección IP y/o por nombre de usuario mediante el fichero pg_hba.conf situado en PG_DATA.
- Las conexiones de los clientes pueden ser autentificadas mediante otros paquetes externos.
- A cada usuario de Postgres se le asigna un nombre de usuario y (opcionalmente) una contraseña. Por defecto, los usarios no tienen permiso de escritura a bases de datos que no hayan creado.
- Los usuarios pueden ser incluidos en *grupos*, y el acceso a las tablas puede restringirse en base a esos grupos.

## 8.1. Autentificacion de Usuarios

*Autentificacion* es el proceso mediante el cual el servidor de la base de datos y el postmaster se aseguran de que el usario que está solicitando acceso a la base de datos es en realidad quien dice ser. Todos los usarios que quieren utilizar Postgres se

comprueban en la tabla `pg_user` para asegurarse que están autorizados a hacerlo. Actualmente, la verificación de la identidad del usuario se realiza de distintas formas:

Desde la shell del usuario

Un demonio que se lanza desde la shell del usuario anota el id original del usuario antes de realizar un `setuid` al id del usuario *`postgres`*. El id original del usuario se emplea como base para todo tipo de comprobaciones.

Desde la red

Si Postgres se instala como distribuido, el acceso al puerto TCP del postmaster está disponible para todo el mundo. El ABD configura el fichero pg_hba.conf situado en el directorio PG_DATA especificando el sistema de autentificacion a utilizar en base al equipo que realiza la conexión y la base de datos a la que se conecta. Ver *pg_hba.conf(5)* para obtener una descripción de los sistemas de autentificación disponibles. Por supuesto la autentificación basada en equipos no es perfecta incluso en los sistemas Unix. Es posible, para determinados intrusos, enmascarar el equipo de origen. Estos temas de seguridad están fuera del alcance de Postgres.

## 8.2. Nombres de usuario y grupos

Para crear un nuevo usuario, ejecute el programa createuser.

Para añadir un usario o un grupo de usarios a un nuevo grupo uno de los usuarios debe crear el grupo y añadir al resto a ese grupo. En Postgres estos pasos no pueden realizarse actualmente mendiante el comando **create group**. Los grupos se definen añadiendo los valores a la tabla `pg_group`, y usando el comando **grant** para asignar privilegios al grupo.

### 8.2.1. Crear Usuarios

### 8.2.2. Crear Grupos

Actualmente no hay una forma facil de crear grupos de usuarios. Hay que añadirlos/actualizarlos uno a uno en la tabla `pg_group table`. Por ejemplo: jolly=> insert into pg_group (groname, grosysid, grolist) jolly=> values (’posthackers’, ’1234’, ’{5443, 8261}’); INSERT 548224 jolly=> grant insert on foo to group posthackers; CHANGE jolly=> Los campos de pg_group son: * groname: El nombre del grupo. Este campo debe de ser unicamente alfanumérico. No añadas subrayados u otros signos de puntuación. * grosysid: El id del grupo. El tipo del campo es int4 y debe de ser único para cada grupo. * grolist: La lista de id de usarios que pertenecen al grupo. Este campo es de tipo int4.

### 8.2.3. Asignar usuarios a los Grupos

## 8.3. Control de Acceso

Postgres proporciona mecanismos para permitir a los usuarios limitar el acceso que otros usuarios tendrán a sus datos.

SuperUsuarios de la Base de Datos

Los SuperUsuarios de la base de datos (aquellos que tienen el campo `pg_user.usesuper` activado) ignoran todos los controles de acceso descritos anteriormente con dos excepciones: las actualizaciones del catálogo del sistema no están permitidas si el usuario no tiene el campo `pg_user.usecatupd` activado, y

nunca se permite la destrucción del catálogo del sistema (o la modificación de sus estructuras).

Privilegios de acceso

El uso de los privilegios de acceso para limitar la lectura, escritura y la puesta de reglas a las clases se trata en *grant/revoke(l)*.

Borrado de clases y modificación de estructuras.

Los comandos que borran o modifican la estructura de una clase, como **alter**, **drop table**, y **drop index**, solo funcionan con el propietario de la clase. Como hemos dicho antes, estas operaciones no están permitidas *nunca* en los catálogos del systema.

# 8.4. Funciones y Reglas

Las funciones y las reglas permiten a los usuarios insertar código en el servidor de la base de datos que otros usuarios pueden ejecutar sin saberlo. Ambos mecanismos permiten a los usuarios alojar *caballos de troya* con relativa impunidad. La única protección efectiva es un estrecho control sobre quien puede definir funciones (por ejemplo, escribir en tablas con campos SQL) y reglas. También se recomienda auditar seguimientos y alertas en `pg_class`, `pg_user` y `pg_group`.

## 8.4.1. Funciones

Las funciones escritas en cualquier lenguaje excepto SQL se ejecutan por el servidor de la base de datos con el mismo permiso que el usuario *`postgres`* (el servidor de la base de datos funciona con el user-id de *`postgres`*. Es posible cambiar las estructuras de datos internas del servidor por los usuarios, desde dentro de funciones de confianza. Es por ello que este tipo de funciones pueden, entre otras cosas, evitar cualquier

sistema de control de acceso. Este es un problema inherente a las funciones definidas por los usuarios en C.

## 8.4.2. Reglas

Como en las funciones SQL, las reglas tambíen se ejecutan con la identidad y los permisos del usuario que llamó al servidor de la base de datos.

## 8.4.3. Caveats

There are no plans to explicitly support encrypted data inside of Postgres (though there is nothing to prevent users from encrypting data within user-defined functions). There are no plans to explicitly support encrypted network connections, either, pending a total rewrite of the frontend/backend protocol.

User names, group names and associated system identifiers (e.g., the contents of `pg_user.usesysid`) are assumed to be unique throughout a database. Unpredictable results may occur if they are not.

# Capítulo 9. Agregar y Eliminar Usuarios

createuser permite que usuarios específicos accedan a Postgres. dropuser elimina usuarios y previene que éstos accedan a Postgres.

Estas órdenes sólo afectan a los usuarios con respecto a Postgres; no tienen efecto en otros privilegios del usuario o en su estado con respecto al sistema operativo subyacente.

# Capítulo 10. Gestión de Disco

## 10.1. Localizaciones Alternativas

Se puede crear una base de datos en una localización diferente a la establecida por defecto durante la instalación. Recuerde que todos los accesos a base de datos ocurren realmente a traves del proceso en segundo plano, así que éste debe poder acceder a cualquier especificación.

Se crean localizaciónes alternativas y referencias mediante una variable de entorno que da el path absoluto hasta la situación de almacenamiento deseada. Esta variable de entorno debe estar definida antes de que el proceso en segundo plano sea arrancado y debe ser modificable mediante la cuenta del administrador de postgres. Cualquier variable de entorno puede ser utilizada para referirse a una localización alternativa, si bien se recomienda la utilización de un nombre de variable con prefijo PGDATA para evitar confusión y conflicto con otras variables.

> **Nota:** En versiones previas de Postgres, también estaba permitido utilizar un nombre de path absoluto para especificar una localización de almacenamiento alternativa. Se prefiere el método de especificación de variables de entorno, puesto que concede al administrador del sistema más flexibilidad en la gestión del almacenamiento en disco. Si prefiere utilizar paths absolutos, puede hacerlo definiendo "ALLOW_ABSOLUTE_DBPATHS" y recompilando Postgres. Para hacer esto, añada cualquiera de estas líneas
>
> ```
> #define ALLOW_ABSOLUTE_DBPATHS 1
> ```
>
> al archivo `src/include/config.h`, o especifique
>
> ```
> CFLAGS+= -DALLOW_ABSOLUTE_DBPATHS
> ```
>
> en su `Makefile.custom`.

Recuerde que la creación de una base de datos la ejecuta realmente un proceso de la base de datos en segundo plano. Por lo tanto, cualquier variable de entorno que especifique una localización alternativa debe ser definida antes de que el proceso en segundo plano sea arrancado. Para definir una localización alternativa apuntando a PGDATA2 `/home/postgres/data`, primero escriba

```
% setenv PGDATA2 /home/postgres/data
```

para definir la variable de entorno que será utilizada con las órdenes siguientes. Normalmente, querrá definir esta variable en el fichero de inicialización del super usuario de Postgres, `.profile` o `.cshrc` para asegurar que está definido al arrancar el sistema. Se puede utilizar cualquier variable de entorno para referirse a una localización alternativa, aunque se prefiere que las variables estén prefijadas con "PGDATA" para eliminar confusiones y la posibilidad de conflictos con otras variables, o su reescritura.

Para crear un area de almacenamiento de datos en PGDATA2, asegúrese de que `/home/postgres` ya existe y puede ser escrito por el administrador de postgres. Después desde la linea de órdenes, escriba

```
% setenv PGDATA2 /home/postgres/data
% initlocation $PGDATA2
Creating Postgres database system directory /home/postgres/data

Creating Postgres database system directory /home/postgres/data/base
```

Para comprobar la nueva localización, cree una base de datos test escribiendo

```
% createdb -D PGDATA2 test
% dropdb test
```

# Capítulo 11. Gestión de una base de datos

Si el programa postmaster de Postgres está cargado y en ejecución, podemos crear algunas bases de datos con las que experimentar. En este documento describiremos las órdenes básicos para gestionar una base de datos.

## 11.1. Creación de una base de datos

Supongamos que quiere crear una base de datos llamada mibase. Puede hacerlo con el siguiente orden:

```
% createdb nombredb
```

Postgres le permite crear cualquier número de bases de datos en una máquina dada, y usted se convierte automáticamente en el administrador de la base de datos que acaba de crear. Los nombres de las bases de datos han de comenzar por un caracter alfabético, y su longitud está limitada a 31 caracteres. No todos los usuarios están autorizados para convertirse en administradores de bases de datos. Si Postgres rechaza la orden de crear bases de datos, es que necesita que el administrador del sistema le garantice derechos para crear las bases de datos. Consulte a su administrador de sistemas si le ocurre eso.

## 11.2. Acceso a la base de datos

Una vez que ha construido la base de datos, puede acceder a ella por los siguientes medios:

- Ejecutando el programa monitor de terminal de Postgres (psql) que le permite

introducir, editar y ejecutar órdenes SQL de un modo interactivo.

- Escribiendo un programa en C que use la biblioteca de rutinas `libpq`. Esto le permite enviar órdenes SQL desde C y obtener las respuestas y mensajes de estado en su programa. Esta interfaz se discute en el documento *Guía de programación en PostgreSQL*.

Puede que quiera ejecutar el programa psql, para probar los ejemplos de este manual. Puede activarlo para la base de datos *`nombredb`* escribiendo la orden:

```
% psql nombredb
```

Recibirá como respuesta el siguiente mensaje:

```
Welcome to the Postgres interactive sql monitor:

  type \? for help on slash commands
  type \q to quit
  type \g or terminate with semicolon to execute query
You are currently connected to the database: nombredb

nombredb=>
```

Este indicativo le informa de que el monitor de terminal se encuentra dispuesto y que puede escribir consultas SQL en el espacio de trabajo creado por el citado monitor de terminal. El programa psql responde a códigos de escape que comienzan con la barra invertida, "\". Por ejemplo, puede obtener ayuda sobre la sintaxis de varias órdenes SQL de Postgres escribiendo:

```
nombredb=> \h
```

Una vez que ha terminado de introducir sus consultas en el espacio de trabajo, puede pasar el contenido de éste al servidor de Postgres escribiendo:

```
nombredb=> \g
```

Esto le dice al servidor que procese la consulta. Si termina su consulta con punto y coma, no es necesario que introduzca la secuencia \g. psql procesará automáticamente consultas terminadas en punto y coma. Para leer las consultas de un fichero, en lugar de introducirlas interactivamente, escriba:

```
nombredb=> \i fichero
```

Para salir de psql y volver a Unix, escriba:

```
nombredb=> \q
```

y psql terminará y le devolverá el intérprete de órdenes. (Para conocer más órdenes de escape, escriba \h en la entrada del monitor.) Los espacios en blanco (espacios, tabuladores y saltos de línea) pueden usarse libremente en las consultas SQL. Los comentarios en una sola línea se indican mediante dos guiones (“–”). Todo lo que vaya desde los guiones hasta el fin de la línea será ignorado. Los comentarios que abarcan múltiples líneas, o que están dentro de una línea se encierran entre “/* ... */”, una convención que se tomó prestada de Ingres.

## 11.3. Destrucción de una base de datos

Si usted es el administrador de la base de datos mibase, puede destruirla usando el siguiente orden de Unix:

```
% dropdb nombredb
```

Esta acción elimina físicamente todos los ficheros asociados con la base de datos y no puede ser invertida, por lo que sólo debe ser usada con gran precaución.

Es también posible destruir una base de datos desde una sesión SQL usando:

```
> drop database nombredb
```

## 11.4. Copia de seguridad y restauración

> **Atención**
>
> Deben realizarse copias de seguridad de las bases de datos regularmente. Dado que Postgres gestiona sus propios ficheros en el sistema, *no se recomienda* confiar en los sistemas de copia de seguridad del sistema para las copias de respaldo de las bases de datos; no hay garantía de que los ficheros estén en un estado consistente que permita su uso después de la restauración.

Postgres proporciona dos utilidades para realizar las copias de seguridad de su sistema: pg_dump para copias de seguridad de bases de datos individuales y pg_dumpall para realizar copias de seguridad de toda la instalación de una sola vez.

La copia de seguridad de una sola base de datos puede realizarse usando la siguiente orden:

```
% pg_dump nombredb > nombredb.pgdump
```

y puede ser restaurada usando

```
cat nombredb.pgdump | psql nombredb
```

Esta técnica puede usarse para mover bases de datos a una nueva localización y para renombrar bases de datos existentes..

## 11.4.1. Bases de datos grandes

**Autor:** Escrito por Hannu Krosing (hannu@trust.ee) on 1999-06-19.

Dado que Postgres permite tablas de mayor tamaño que el permitido por el sistema de ficheros, puede resultar problemático el volcado de una tabla a un fichero, ya que el fichero resultante seguramente superará el tamaño máximo permitido.

Como pg_dump escribe en stdout, puede usar las herramientas *nix para sortear estos posibles problemas:

- Uso de volcados comprimidos:

  ```
  % pg_dump nombredb | gzip > nombrefichero.dump.gz
  ```

  la recuperamos con:

  ```
  % createdb nombredb
  % gunzip -c nombrefichero.dump.gz | psql nombredb
  ```

  o

  ```
  % cat nombrefichero.dump.gz | gunzip | psql nombredb
  ```

- Use split:

  ```
  % pg_dump nombredb | split -b 1m - nombrefichero.dump.
  ```

  y lo recuperamos con:

  ```
  % createdb nombredb
  ```

```
% cat nombrefichero.dump.* | pgsql nombredb
```

Por supuesto, el nombre del fichero (*nombrefichero*) y el contenido de la salida de pg_dump no tiene por qué coincidir con el nombre de la base de datos. Además, la base de datos restaurada puede tener un nombre distinto, por lo que este mecanismo también es efectivo para renombrar bases de datos.

# Capítulo 12. Tratamiento de problemas

## 12.1. Fallos de inicio de Postmaster

Hay varias posibles razones para que postmaster no pueda inicializarse. Compruebe el fichero de registro de postmaster, o inícielo manualmente (sin redirigir la salida estándar o la de errores) para ver los mensajes que aparecen. Alguno de los posibles mensajes de error son autoexplicativos, pero los hay que pueden no serlos tanto:

```
FATAL: StreamServerPort: bind() failed: Address already in use
        Is another postmaster already running on that port?
```

Esto normalmente significa lo que sugiere: accidentalmente ha iniciado una segunda instancia de postmaster en el mismo puerto en el que ya se está ejecutando uno. Sin embargo, si el mensaje de error del núcleo no es "Address already in use" o alguna variante, puede estar ocurriendo otro problema. Por ejemplo, el tratar de iniciar una sesión de postmaster en un puerto de error reservado puede producir algo como:

```
$ postmaster -i -p 666
FATAL: StreamServerPort: bind() failed: Permission denied
        Is another postmaster already running on that port?
```

```
IpcMemoryCreate: shmget failed (Invalid argument) key=5440001, si-
ze=83918612, permission=600
FATAL 1:  ShmemCreate: cannot create region
```

Un mensaje como éste posiblemente indica que el limite impuesto al tamaño de las zonas de memoria compartidas es menor que área de «buffer» que Postgres está

intentando crear. (O puede significar que no dispone de soporte para la memoria compartida de tipo SysV configurado en su núcleo.) Como arreglo temporal puede tratar de iniciar postmaster con un número de «buffers» menor de lo normal (parámetro -B). Sin embargo, debería reconfigurar su núcleo para incrementar el tamaño permitido para la memoria compartida. Este mensaje puede aparecer cuando trate de iniciar varias sesiones de postmaster en la misma máquina, si el total de espacio necesario excede el límite impuesto por el núcleo.

```
IpcSemaphoreCreate: semget failed (No space left on device) key=5440026, num
mission=600
```

Un mensaje como éste *no* significa que se haya quedado sin espacio en el disco; significa que la cantidad máxima de semáforos permitidos por el núcleo para el SysV es menor que la cantidad que Postgres intenta crear. Como antes, puede evitar este problema iniciando el postmaster con un numero de procesos «backend» menor (parámetro -N), pero sería mejor que incrementara el límite impuesto por el núcleo.

## 12.2. Problemas con la conexión del Cliente

Una vez que tiene el postmaster en ejecución, al tratar de conectar con él mediante una aplicación cliente puede producirse un fallo por varias razones. Los ejemplos de mensajes de error mostrados aquí son para clientes basados en las versiones recientes de libpq; los clientes basados en otras bibliotecas de interfaz pueden producir otros mensajes, con más o menos información.

```
connectDB() - connect() failed: Connection refused
Is the postmaster running (with -i) at 'server.joe.com' and accep-
ting connections on TCP/IP port '5432'?
```

Este el es fallo genérico de 'No puedo encontrar un postmaster con el que comunicarme'. Puede ocurrir algo así cuando se intenta una comunicación TCP/IP o

mediante socket Unix con un postmaster local:

```
connectDB() - connect() failed: No such file or directory
Is the postmaster running at ’localhost’ and accepting connections on Unix s
ket ’5432’?
```

La última línea es útil para verificar que el cliente está tratando de conectar donde se supone que debe hacerlo. Si en realidad no hay ningún postmaster ejecutándose allí, el mensaje de error del núcleo será del tipo de ’Conexión rehusada’ o de ’No existe el fichero o directorio’, como los anteriores. (Es particularmente importante tener en cuenta que ’Conexión rehusada’ en este contexto *no* no significa que el postmaster haya recibido la petición de conexión y la haya rechazado; en este caso se produce un mensaje diferente, como se verá.) Otros mensajes de error, como el de "Connection timed out" sí indican problemas más importantes, como la falta de conectividad en la red.

```
No pg_hba.conf entry for host 123.123.123.123, user joeblow, data-
base testdb
```

Esto es lo más probable que obtenga si consigue contactar con un postmaster, pero éste no quiere hablar con usted. Como sugiere el mensaje, el postmaster rehúsa la petición de conexión porque no encuentra un renglón de autorización en su fichero de configuración pg_hba.conf

```
Password authentication failed for user ’joeblow’
```

Los mensajes como éste indican que ha contactado con el postmaster, y éste está dispuesto a hablar con usted, pero no lo hará hasta que supere el método de autorización especificado en el fichero pg_hba.conf. Compruebe la clave que está enviando, o su programa IDENT o Kerberos, si el mensaje de error menciona alguno de esos tipos de autenticación.

```
FATAL 1:  SetUserId: user 'joeblow' is not in 'pg_shadow'
```

Esta es otra variante de fallo de autenticación: no se ha ejecutado la orden de Postgres 'create_user' para el nombre de usuario indicado.

```
FATAL 1:  Database testdb does not exist in pg_database
```

No hay base de datos con ese nombre bajo el control de ese postmaster. Nótese que si no especifica el nombre de la base de datos, se aplica por defecto su nombre de usuario en Postgres, lo que puede no ser lo correcto.

## 12.3. Depuración de mensajes

El postmaster presenta ocasionalmente mensajes que pueden ser de ayuda en la solución de problemas. Si desea ver mensajes de depuración de postmaster, puede iniciarlo con la opción -d y redirigir la salida a un fichero de registro:

```
% postmaster -d > pm.log 2>&1 &
```

Si no desea ver estos mensajes, puede escribir

```
% postmaster -S
```

y el postmaster entrará en modo 'S'ilencioso. Nótese que no se incluye el simbolo '&' en el último ejemplo, ya que el postmaster se ejecutará en segundo plano.

## 12.3.1. pg_options

**Nota:** Contribución de Massimo Dal Zotto (mailto:dz@cs.unitn.it)

El fichero opcional `data/pg_options` contiene opciones de ejecución usadas por el backend para controlar mensajes de ejecución y otros parámetros ajustables. Lo que hace interesante a este fichero es el hecho de que es releído por el backend cuando recibe una señal SIGHUP, haciendo así posible cambiar opciones de ejecución sin tener que reiniciar Postgres. Las opciones especificadas en este fichero pueden incluir puntos de depuración usados por el paquete trace (`backend/utils/misc/trace.c`) o parámetro numéricos que puede usar el backend para controlar su comportamiento. Se pueden definir nuevas opciones y parámetros en `backend/utils/misc/trace.c` y en `backend/include/utils/trace.h`.

Las opciones de pg_option pueden especificarse con el parámetro `-T` de Postgres:

```
postgres opciones -T "verbose=2,query,hostlookup-"
```

Las funciones usadas para imprimir errores y mensajes de depuración pueden ahora usar la utilidad *syslog(2)*. Los mensajes impresos en stdout o stderr son precedidos por una etiqueta informativa que incluye la fecha y hora y el pid del backend:

```
#timestamp          #pid    #message
980127.17:52:14.173 [29271] StartTransactionCommand
980127.17:52:14.174 [29271] ProcessUtility: drop table t;
980127.17:52:14.186 [29271] SIIncNumEntries: table is 70% full
980127.17:52:14.186 [29286] Async_NotifyHandler
980127.17:52:14.186 [29286] Waking up sleeping backend process
980127.19:52:14.292 [29286] Async_NotifyFrontEnd
980127.19:52:14.413 [29286] Async_NotifyFrontEnd done
```

```
980127.19:52:14.466 [29286] Async_NotifyHandler done
```

Este formato mejora la legibilidad de los registros, y permite comprender qué «backend» concreto está haciendo qué y en qué momento. También hace más fácil escribir guiones (scripts) en awk o perl que monitoricen el fichero de registro para detectar errores o problemas en la base de datos, o para contabilizar estadísticas temporales de las transacciones.

Los mensajes impresos por syslog usan la utilidad de registro LOG_LOCAL0. El uso de syslog puede ser controlado por las opciones referentes a él en syslog. Desgraciadamente, muchas funciones llaman directamente a `printf()` para mostrar sus mensajes en stdout o stderr y esta salida no puede ser redirigida a syslog o incluir información sobre fecha y hora. Sería aconsejable que todas las llamadas a printf pudieran ser reemplazadas por la macro PRINTF y la salida a stderr se cambiaran para que usaran EPRINTF en su lugar, de modo que se pudieran controlar todas las salidas de un modo uniforme.

El formato del fichero `pg_options` es como sigue:

```
# comentario
opción=valor_entero  # set value for opción
opción                # set opción = 1
opción+               # set opción = 1
opción-               # set opción = 0
```

Nótese que *`palabra_clave`* puede ser también una abreviación del nombre de opción definido en `backend/utils/misc/trace.c`.

Véase *Usando pg_options* para una lista completa de las opciones y sus posible valores.

# Capítulo 13. Recuperación de bases de datos

Esta sección necesita ser escrita. ¿Algún voluntario?

# Capítulo 14. Pruebas de regresión

Instruciones y análisis del test de regresión

Los tests de regresión de PostgreSQL son un conjunto completo de pruebas para la implementación de SQL embebidos en PostgreSQL. Realizan pruebas tanto sobre operaciones SQL estándar, como también sobre las capacidades añadidas por PostgreSQL.

Los tests de regresión fueron desarrollados originalmente por Jolly Chen y Andrew Yu, y fueron extensamente repasados/reempaquetados por Fournier y Thomas Lockhart. Para PostgreSQL v6.1 en adelante los tests de regresión forman parte de cada release oficial.

Algunas bases de datos PostgreSQL correctamente instaladas y totalmente funcionales pueden fallar en alguno de estos test de regresión debido a problemas con la representación del punto flotante y el soporte de zona horaria. Los tests actuales son evaluados usando un sencillo algoritmo "diff", y son muy sensibles a pequeñas diferencias en el sistema. Para tests aparentemente fallidos, si se examinan estas diferencias, pueden revelar no ser significativas.

Las notas sobre tests de regresión de abajo asumen lo siguiente (excepto en casos indicados):

- Las instrucciones son compatibles con Unix. Vea la nota abajo.
- Se usan las opciones por defecto excepto donde se indica.
- El usuario postgres es el superusuario Postgres.
- La ruta de las fuentes es /usr/src/pgsql (son posibles otras rutas).
- La ruta de los ejecutables es /usr/local/pgsql (son posibles otras rutas).

# 14.1. Entorno de regresión

Para preparar los tests de regresión, haga **make all** en el directorio de los tests de regresión. Esto compila un programa C con funciones extendidas PostgreSQL en un librería compartida. Se generan algunos guiones (scripts) SQL localizados y archivos de salida comparativos para los tests que los necesiten. La localización reemplaza macros en los archivos de fuentes con rutas absolutas y nombres de usuario.

Normalmente, los tests de regresión deben ser ejecutados por el usuario postgres ya que el directorio ’src/test/regress’ y subdirectorios son de su propiedad. Si ejecuta los test de regresión con otro usuario el directorio ’src/test/regress’ debe tener permisos de escritura para ese usuario.

Antes era estrictamente necesario ejecutar el postmaster con la zona horaria del sistema establecida en PST, pero ya no es necesario. Puede ejecutar los tests de regresión sobre su configuración habitual del postmaster. El guión (script) del test establecerá la variable de entorno PGTZ para asegurar que los tests dependientes de la zona horaria produzcan los resultados esperados. De todas formas, su sistema debe proporcionar librerías de soporte para la zona horaria PST8PDT, o los tests dependientes de la zona horaria fallarán. Para comprobar que su equipo soporta esto, escriba lo siguiente:

```
setenv TZ PST8PDT
date
```

La orden "date" de arriba tiene que devolver la hora actual del sistema en la zona horaria PST8PDT. Si la base de datos PST8PDT no está disponible, entonces el sistema tiene que devolver la hora en GMT. Si la zona horaria PST8PDT no está disponible, puede establecer las reglas para esa zona horaria explicitamente.

```
setenv PGTZ PST8PDT7,M04.01.0,M10.05.03
```

## 14.2. Estructura de directorios

**Nota:** Esto debería ser una tabla en la sección anterior.

```
input/ .... .Archivos fuente que son convertidos, usando 'make all', en
            alguno de los archivos .sql en el subdirectorio 'sql'

output/ ... .Archivos fuente que son convertidos, usando 'make all', en
            archivos .out en el subdirectorio 'expected'

sql/ ...... .Archivos sql usados para ejecutar los tests de regresión

expected/ . .Archivos .out que representan lo que *esperamos* que parezcan
            los resultados

results/ .. .Archivos .out que contienen lo que los resultados *realmente*
            parecen. Además es usado como almacén temporal para el test de
            copia de tablas.
```

## 14.3. Procedimiento para el test de regresión

Las instrucciones están probadas en un RedHat Linux versión 4.2 usando el intérprete de órdenes bash. Excepto donde se indique, seguramente funcione en la mayoría de sistemas. Instrucciones como `ps` y `tar` cambian mucho las opciones que debe usar en cada plataforma. *Use el sentido común* antes de escibir estas instrucciones.

Para una instalación nueva o si está actualizando una versión anterior de Postgres:

**Configuración de la Regresión de Postgres**

1. El archivo /usr/src/pgsql/src/test/regress/README tiene instrucciones detalladas para la ejecución e interpretación de los tests de regresión. Lo que sigue es una versión más corta:

   Si el postmaster no se está ejecutando ya, inicie el postmaster en una ventana que esté disponible escribiendo

   ```
   postmaster
   ```

   o inicie el demonio postmaster en segundo plano escibiendo

   ```
   cd
   nohup postmaster > regress.log 2>&1 &
   ```

   Ejecute postmaster desde la cuenta de superusuario de Postgres (normalmente la cuenta postgres).

   **Nota:** No ejecute `postmaster` desde la cuenta de root.

2. Si ha ejecutado anteriormente los tests de regresión, borre el directorio de trabajo con:

   ```
   cd /usr/src/pgsql/src/test/regress
   gmake clean
   ```

   No necesita escribir "gmake clean" si es la primera vez que está ejecuntado los tests.

3. Ejecute los tests de regresión. Escriba

```
cd /usr/src/pgsql/src/test/regress
gmake all
```

4. Ejecute los tests de regresión. Escriba

```
cd /usr/src/pgsql/src/test/regress
gmake runtest
```

5. Debería obtener en la pantalla (y además escrito en el archivo ./regress.out) una serie de líneas indicando qué tests han pasado y qué tests han fallado. Tenga en cuenta que puede ser normal que alguno de los tests falle. Para los tests fallidos, use diff para comparar los archivos de los directorios ./results y ./expected. Si falla float8, escriba algo como esto:

```
cd /usr/src/pgsql/src/test/regress
diff -w expected/float8.out results
```

6. Después de ejecutar los tests y examinar los resultados, escriba

```
dropdb regression
cd /usr/src/pgsql/src/test/regress
gmake clean
```

para recuperar el espacio en disco temporal usado por los tests.

## 14.4. Análisis de Regresión

Los resultados se encuentran en los archivos del directorio ./results. Estos resultados pueden ser comparados con los resultados del directorio ./expected usando 'diff'. (El

guión (script) del test hace esto por usted, y deja las diferencias en ./regression.diffs.)

Los archivos pueden no corresponderse de forma exacta. El guión del test informará deuna diferencia como "failure" (fallo), pero la diferencia puede deberse a pequeñas variaciones entre plataformas en los mensajes de error, comportamiento de la librería matemáica, etc. "Fallos" de este estilo no indican necesariamente un problema con Postgres.

Por tanto, es necesario examinar las diferencias de cada test "fallido" con el fin de determinar si existe un problema realmente. Los siguientes puntos intentan proporcionar una guía para determinar si una diferencia es significativa o no.

## 14.4.1. Diferencias en los mensajes de error

Alguno de los tests de regresión incluyen intencionadamente valores de entrada no válidos. Los mensajes de error pueden venir tanto del código de Postgres como de las rutinas de sistema de la plataforma en la que nos encontremos. En el último caso, los mensajes pueden variar entre plataformas, pero deben reflejar información similar. Estas diferencias en los mensajes darán como resultado un test "fallido" que puede ser validado mediante una inspección.

## 14.4.2. Diferencias en fechas y horas

Muchos de los resultados de fecha y hora son dependientes del entorno de la zona horaria. Los achivos de referencia están generados para la zona horaria PST8PDT (Berkeley, California) y aparentemente pueden parecer fallos si los tests no se ejecutan con esta zona horaria establecida. El programa que ejecuta los tests de regresión establece la variable de entorno PGTZ a PST8PDT para asegurar resultados parecidos.

Parece que algunos sistemas no aceptan la sintaxis recomendada para establecer explicitamente las reglas de la zona horaria local; puede ser que necesite usar una forma distinta para establecer PGTZ en estas máquinas.

Algunos sistemas que usan librerias antiguas de zonas horarias fallan al aplicar las correciones de ahorro de luz solar en las fechas anteriores a 1970, causando que las

horas de esas fechas aparezcan en PST a pesar de todo. Esto dará pie a diferencias localizadas en los resultados de los tests.

## 14.4.3. Diferencias en punto flotante

Algunos de los tests implican calcular números de 64-bits (float8) a partir de las columnas de una tabla. Se han observado diferencias en los resultados que devuelven funciones matemáticas en columnas de tipo float8. Los tests con float8 y de geometría son particularmente propensos a pequeñas diferencias entre plataformas. Se precisa una comparación con lupa por parte humana para determinar diferencias que normalmente se encuentran 10 posiciones a la derecha del punto decimal.

Algunos errores de señales del sistema con pow() y exp() difieren de los mecanismos que espera el actual código de Postgres.

## 14.4.4. Diferencias en polígonos

Varios de los tests incluyen operaciones con coordenadas sobre el callejero de Oakland/Berkley CA. Los datos de este mapa vienen expresados como polígonos cuyos vértices están representados en pares de números float8 (latitud y longitud decimal). Inicialmente, se crean y llenan algunas tablas con coordenadas, después se crean algunas vistas (Views) haciendo el Join de dos tablas usando el operador de intersección de polígonos (##), y despúes se realiza un Select sobre la vista. Cuando comparamos los resultados de diferentes plataformas, las diferencias aparecen en el segundo o tercer lugar a la derecha del punto decimal. Las instrucciones SQL donde se dan estos problemas son las siguientes:

```
QUERY: SELECT * from street;
QUERY: SELECT * from iexit;
```

### 14.4.5. Diferencias aleatorias

Hay al menos un caso de test en random.out que esta diseñado para producir resultados aleatorios. Esto causa que random falle el test de regresión cada vez. Escribir

```
diff results/random.out expected/random.out
```

debe producir una o unas pocas líneas de diferencias por esta razón, pero otras variaciones en punto flotante o en arquitecturas distintas pueden causar más diferencias.

### 14.4.6. Los archivos “expected”

Los archivos `./expected/*.out` fueron adaptados del monolítico archivo original `expected.input` proporcionado por Jolly Chen. Versiones más modernas de estos archivos generadas en varias máquinas de desarrollo han sido sustituidas después de una cuidadosa (?) inspección. Muchas de estas máquinas de desarrollo están ejecutando variantes del Unix OS (FreeBSD, Linux, etc) en hardware Ix86. El archivo original `expected.input` fue creado en un sistema SPARC Solaris 2.4 usando el código de `postgres5-1.02a5.tar.gz`. Fue comparado con un archivo creado en un sistema I386 Solaris 2.4 y las diferencias fueron solamente en los polígonos de punto flotante en el tercer dígito a la derecha del punto decimal. (vea más arriba) El archivo original `sample.regress.out` se obtuvo de la entrega 1.01 de postgres construida por Jolly Chen y se incluye aquí para referencia. Tendría que haberse ejecutado con una máquina DEC ALPHA ya que el `Makefile.global` en la version 1.01 de postgres tiene PORTNAME=alpha.

## 14.5. Archivos de comparación específicos de la plataforma

Como alguno de los tests producen resultados inherentes a la plataforma usada, hemos

proporcionamos una forma para suplir los archivos de comparación especificos para cada plataforma. Frecuentemente se da la misma variación en múltiples plataformas; en vez de dar un archivo de comparación separado para cada plataforma, existe un archivo guía que define qué archivo de comparación usar. De forma que, para eliminar fallos tontos de una plataforma en particular, debe elegir o crear un fichero de resultados variantes, y añadir una línea al archivo guía, que es "mapa de resultados".

Cada línea del archivo guía es de la siguiente forma

```
testname/platformnamepattern=comparisonfilename
```

El nombre del test (testname) es sencillamente el nombre del módulo de regresión de ese test en particular. El patrón del nombre de la plataforma (platformnamepattern) está generado al estilo de expr(1) (que es una expresión regular con el símbolo ^ implícito al principio). Esta se comprueba con el nombre de la plataforma tal como viene escrito en config.guess. El nombre del fichero de comparación (comparisonfilename) es el nombre del sustituto del fichero de resultados de comparación.

Por ejemplo: el test de regresión int2 incluye una entrada deliberada de un valor que es demasiado largo para caber en un int2. El mensaje de error específico que es producido es dependiente de la plataforma; nuestra plataforma de referencia saca

```
ERROR:  pg_atoi: error reading "100000": Numerical result out of ran-
ge
```

pero en un buen número de otras plataformas Unix saca

```
ERROR:  pg_atoi: error reading "100000": Result too large
```

En este caso, proporcionamos una variante del archivo de comparación, int2-too-large.out, que incluye la sintaxis de este mensaje de error. Para no mostrar estos "fallos" tontos en las plataformas HPPA, el resultmap (mapa de resultados) incluye

```
int2/hppa=int2-too-large
```

que se activará en cualquier máquina en el que la salida de config.guess comience por ’hppa’. Otras líneas en el resultmap seleccionan la variante del archivo de comparación para otras plataformas donde sea apropiado.

# Capítulo 15. Notas de versiones

## 15.1. Version 6.5.3

Esta es basicamente una limpieza de la version 6.5.2. Hemos añadido un nuevo pgacces que se perdio en la 6.5.2, e instalado una correcion especifica para NT.

### 15.1.1. Migracion a v6.5.3

*No* se requiere un dump/restores para aquellos que esten ejecutando una 6.5.*.

### 15.1.2. Lista Detallada de Cambios

```
Version actualizada de pgacces 0.98
Parche especifico para NT
Correccion para reglas de volcado en tablas heredadas
```

## 15.2. Version 6.5.2

Esta es basicamente una limpieza de la version 6.5.1. Hemos corregio una variedad de problemas reportados por usuarios de 6.5.1.

### 15.2.1. Migracion to v6.5.2

*No* se requiere un dump/restores para aquellos que esten ejecutando una 6.5.*.

## 15.2.2. Lista Detallada de Cambios

```
corregidas las subselect+CASE (Tom)
Añadida configuracion de SHLIB_LINK para los portes de solaris_i386 y so-
laris_sparc(Daren Sefcik)
Correciones para CASE y WHERE en clausulas "join"(Tom)
Correcion para aborto en BTScan(Tom)
Reparada la comprobacion para UNIQUE redundante e indices PRIMARY KEY(Thomas
Mejorado para que se compruebe las restricciones en multi-columnas(Thomas)
Correcion para Win32 que tenia problemas con MB habilitado(Hiroki Kataoka)
Permite a yacc de BSD y a bison compilar codigo pl(Bruce)
Corregido el trabajo con SET NAMES
corregidos los int8 (Thomas)
Correccion del consumo de memoria de "vacuum"(Hiroshi,Tatsuo)
Reduccion del consumo total de memoria de "vacuum"(Tom)
Correcion para timestamp(datetime)
Correcciones de problemas en las reglas de paso al analizador sintactico(Tom
Correccion del problema de cuotas en mkMakefile.tcldefs.sh.in y mkMakefile.t
This is to re-use space on index pages freed by vacuum(Vadim)
documentado -x para pg_dump(Bruce)
Correcin para operadores unarios en la regla de paso al analizador sintactic
Comentado el FileUnlink de exceso de segmentos durante mdtruncate() (Tom)
Enlazamiento en Irix corregido por Yu Cao >yucao@falcon.kla-tencor.com<
Reparado el error logico en LIKE: no debia devolver un LIKE_ABORT
   cuando alcanza el final de un patron antes del final del texto(Tom)
Reparada limpieza incorrecta de montones de memoria reservadas du-
rante aborto de transaccion(Tom)
Version actualizada de pgaccess 0.98
```

# 15.3. Version 6.5.1

Esta es basicamente una limpieza de la version 6.5. Hemos corregio una variedad de problemas reportados por usuarios de 6.5.

## 15.3.1. Migracion to v6.5.1

*No* se requiere un dump/restores para aquellos que esten ejecutando una 6.5.

## 15.3.2. Lista Detallada de Cambios

```
Añadido un fichero NT LEAME
Correciones de portabilidad para linux_ppc, Irix, linux_alpha, OpenBSD, alph
Eliminado QUERY_LIMIT, utilizar SELECT...LIMIT
Correcion para EXPLAIN en herencia(Tom)
Parche para permitir "vacuum" en tablas con multi-segmentos(Hirosi)
Correcion para la selectividad del optimizador R-Tree(Tom)
Corregida laguna el descriptor de ficheros ACL(Atsushi Ogawa)
Nueva expresion del codigo de sub arboles(Tom)
Se evitan escrituras en disco para transacciones de solo-lectura(Vadim)
Correccion para eliminacion de tablas temporales si la ultima tran-
saccion fue abortada (Bruce)
Correccion para prevenir que sean creadas tuplas demasiado largas(Bruce)
correciones en plpgsql
Se permiten numeros de puerto de 32k - 64k(Bruce)
Añadido ^ precidence(Bruce)
Renombrados ficheros ordendo llamados pg_temp a pg_sorttemp(Bruce)
Correcion para microsegundos en valores temporales(Tom)
Limpieza de la fuente del Tutorial
Nuevo porte a linux_m68k
Correccion para la ordenacion de NULL’s en algunos casos(Tom)
Corregidas las dependencias para librerias compartidas(Tom)
Corregido fallos tecnicos que afectaban a GROUP BY en subselects(Tom)
Correccion para algunas alarmas del compilador (Tomoaki Nishiyama)
```

```
Añadido soporte para Win1250 (Checo) (Pavel Behal)
```

## 15.4. Version 6.5

Esta version marca un avance grande en el conocimiento que el equipo de desarrollo tiene del codigo fuente que heredamos de Berkeley. Veras que podemos añadir mayores features mas facilmente, gracias al incremento en tamaño y experiencia de nuestro mundialmente extenso equipo de desarrollo.

He aqui un conciso sumario de los mas notables cambios:

Control de concurrencia multi-version(MVCC en ingles Multi-version concurrency control)

> Esto elimina nuestro viejo bloqueo a nivel de tabla, y lo reemplaza con un bloqueo del sistema que es superior a la mayoria de los sistemas de bases de datos comerciales. En un sistema tradicional, cada registro que es modificado se bloquea hasta que se confirma la transaccion, previniendo lecturas por otros usuario. MVCC utiliza de modo natural el caracter multi-version de PostgreSQL para permitir que las lecturas continuen leyendo datos consistentes durante la actividad de escritura. Las escrituras continuan utilizando el sistemas de transacciones compacto pg_log. Todo esto se realiza sin tener que reservar un bloqueo para cada registro como en los sistemas tradicionales de base de datos. Asi que, basicamente, ya no estaremos restringidos por mas tiempo por el bloqueo simple a nivel de tabla; tenemos algo mejor que el bloqueo a nivel de registro.

Copias de seguridad en caliente con pg_dump

> pg_dump se beneficia de las nuevas prestaciones de MVCC para dar consistencia a un dump/backup mientras la base de dato pemanece en linea y disponible para ser consultada.

Tipos de datos numericos

Ahora tenemos tipos de datos verdaderamente numericos, con precision especificada por el usuario.

Tablas temporales

Se garantiza que las tablas temporales tienen nombres unicos durante una sesion en la base de datos, y que son destruidas al salir de la sesion.

Nuevas prestaciones SQL

Ahora tenemos soporte para declaraciones CASE, INTERSECT, and EXCEPT. Tenemos nuevos LIMIT/OFFSET, SET TRANSACTION ISOLATION LEVEL, SELECT ... FOR UPDATE, y un mejorado comando LOCK TABLE.

Aceleramiento

Continuamos acelerando PostgreSQL, gracias a la variedad de talentos que hay en nuetsro equipo. Hemos acelerdo la asignacion de memoria, la optimizacion, las uniones de tablas (table join), y las rutinas de transferencias de registros.

Portes

Continuamos ampliando nuestra lista de portes, esta vez incluimos WinNT/ix86 y NetBSD/arm32.

Interfaces

La mayoria de interfaces tienen una nueva version, y la funcionalidad existentes ha sido mejorada.

Documentacion

Nuevo y actualizado material esta presente por toda la documentacion. Se han aportado nuevas FAQs para las plataformas SGI y AIX. El *Tutorial* tiene informacion introductoria sobre SQL de Stefan Simkovics. Para la *Guia del Usuario*, hay paginas de referencia cubriendo la utilidad postmaster y mas programas de utilidad, un apendice nuevo contiene detalles sobre el

comportamiento de date/time. La *Guia del Administrador* tiene un nuevo capitulo sobre resolucion de problemas de Tom Lane. Y la *Guia del Programador* tiene una descripcion del proceso de interrogacion, tambien de Stefan, y detalles acerca de como obtener el arbol del codigo fuente de Postgres por CVS anonimo y CVSup.

## 15.4.1. Migracion to v6.5

Un dump/restore utilizando pg_dump es necesario para aquellos que deseen migrar datos de cualquier version previa de Postgres. pg_upgrade *not* puede ser utilizado para actualizar esta version porque la estructura en disco de las tablas ha cambiado comparada con versiones precedentes.

La nueva caracteristica de Control de Concurrencia Multi-Version (MVCC, en ingles) puede dar comportamientos un poco diferentes en entornos multiusuarios. *Lea y comprenda la seccion siguiente para asegurar que sus aplicaciones existentes le proporcionaran el comportamiento que necesita.*

### 15.4.1.1. Control de Concurrencia Multi-Version

A causa de que las lecturas en 6.5 no bloquean los datos, a pesar del nivel de aislamiento de transaccion, los datos leidos por una transaccion pueden ser sobre escritos por otra. En otras palabras, si un registro es devuelto por **SELECT** eso no significa que este registro exista realmente en el momento en que es devuelto. (i.e algunas veces despues de que la sentencia o la transaccion comience) ni tampoco que el registro este protegido de ser borrado o actualizado por una transaccion en concurrente antes de que la transccion en curso haga commit o rollback.

Para asegurar la existencia actual de un registro y protegerlo contra actulizaciones concurrentes se debe utilizar **SELECT FOR UPDATE** o una sentencia **LOCK TABLE** apropiada. Esto deberia ser tenido en cuento cuando se porten aplicaciones desde versiones precedentes de Postgres y otros entornos.

Tenga todo lo anterior en mente su utiliza triggers `contrib/refint.*` para integridad referencial. Ahora se requieren tecnicas adicionales. Un modo es utilizar el comando

**LOCK parent_table IN SHARE ROW EXCLUSIVE MODE** si una transaccion va a actualizar/borrar una clave primaria y utilizar el comando **LOCK parent_table IN SHARE MODE** si una transaccion va a actualizar/insertar una clave foranea.

> **Nota:** Notese que si ejecuta una transaccion en modo SERIALIZABLE entonces debe ejecutar el comando **LOCK** anterior antes de la ejecucion de cualquier sentencia DML (**SELECT/INSERT/DELETE/UPDATE/FETCH/COPY_TO**) en la transaccion.

Esto inconvenientes desapareceran en el futuro cuando la habilidad para leer datos sucios (no confirmados) (a pesar del nivel de aislamiento) y la verdadera integridad referencial sea implementada.

## 15.4.2. Lista Detalla de Cambios

```
Correciones de errores
------
Correcion de las funciones de conversion text<->float8 y text<->float4(Thoma
Correcion para creacion de tablas con constraints con mixed-case(Billy)
Cambiado comportamiento de exp()/pow() para generar error en underflow/overf
Correcion de error en pg_dump -z
Limpiezas de invasiones de memoria(Tatsuo)
Correcion para abortos de lo_import(Tatsuo)
Ajustes en el manejo de nombres de tipo de datos para suprimir do-
bles comillas(Thomas)
Uso de coersion de tipo para emparejar columnas y DEFAULT(Thomas)
Correccion de deadlock de este modo solo verifica una vez despues de un se-
gundo de espera(Bruce)
Correciones para agregaciones y PL/pgsql(Hiroshi)
Correcion para aborto de subquery(Vadim)
```

```
Correccion de libpq para la funcion PQfnumber y nombres que no dis-
tinguen mayusculas-minusculas(Bahman Rafatjoo)
Correcion para objetos grandes escritos-en-medio, no bloques extra, con-
sumo de memoria(Tatsuo)
Correcion para pg_dump -d o -D y entrecomillado de caracteres es-
peciales en INSERT
Reparados serios problemas con dynahash(Tom)
Correjidos problemas de portabilidad para INET/CIDR
Correcion de problema con error de selectividad en ALTER TABLE ADD COLUMN(Br
Correcion del ejecutor de ese modo trabaja merjejoin de diferentes ti-
pos de columnas(Tom)
Correcion de error para selectividad de OR en Alpha
Correcion para problema de selectividad de indice OR(Bruce)
Correcion \d para que muestre de ese modo la extension apropiada pa-
ra char()/varchar()(Ryan)
Correcion en el codigo del tutorial(Clark)
Mejoras en la comprobacion de destroyuser(Oliver)
Correcion para Kerberos(Rodney McDuff)
Correcion para borrado de la base de datos mientras los buffers no se han li
Correcion la secuencia nextval() para que pueda asi distinguir mayusculas-
minusculas
Correcion para operador !!=
Borrado de buffers antes de destruir los ficheros de las base de datos(Bruce
Correcion del caso en que el ejecutor evalua las funciones dos veces(Tatsuo)
Se permite a las acciones de secuencia netxval distinguir mayusculas-
minusculas(Bruce)
Correcion de optimizador de indexacion para que no trabaje para nu-
meros negativos(Bruce)
Correcion de perdidas de memoria en ejecuciones con fjIsNull
Correcion de perdidas de memoria para aggregate(Erik Riedel)
Se permite que username contenga una almohadilla (#, dash en ingles) en los
misos GRANT
Limpieza de NULL en los tipos inet
Limpieza de errores en tablas del sistema(Tom)
Correcion de problemas de PAGER y del comando \?(Masaaki Sakaida)
Reducido el tamaño de fichero de multi-segment por defecto a 1GB(Peter)
Correcion del volcado de CREATE OPERATOR(Tom)
```

```
Correccion para escaneo hacia atras de cursores(Hiroshi Inoue)
Correccion para COPY FROM STDIN cuando se utiliza \i(Tom)
Correcion para una subselect cuando es comparada dentro de una expresion(Jan
Correcion para manejo de devolucion de error mientras se duelven registros(T
Correcion de problemas con referencia a tipos de array(Tom,Jan)
Se previene oid de UPDATE SET (Jan)
Correcion de pg_dump asi ña opcion -t puede manejar nombres de ta-
bla en mayusculas-minusculas
Correciones para GROUP BY es casos especiales(Tom, Jan)
Correcion de perdidas de memoria en queries falladas(Tom)
DEFAULT soporta ahora identificadores mixed-case(Tom)
Correcion para que multi-sefment utilice DROP/RENAME de tabla, de in-
dice(Ole Gjerde)
Dehabilitacion de uso de pg_dump con las dos opciones -o y -d(Bruce)
Se permite a pg_dump volver adecuadamente permisos de GROUP(Bruce)
Correcion para GROYP BY en INSERT INTO table SELECT * FROM table2(Jan)
Correccion para computaciones en vistas(Jan)
Correciones para agregaciones en array con indices(Tom)
Correccion para como maneja DEFAULT entrecomillado simple en valo-
res que requieren demasiadas comillas
Correccion de problema de seguridad con no super-usuarios que im-
portan/exportan objectos de gran tamaño.(Tom)
Vuelta atras de transaccion que crea table la limpia adecuadamente(Tom)
Correccion para permitir que tablas largas y nombres de columnas ge-
neren nombres en serie adecuados(Tom)

Mejoras
--------
Añadida la utilidad "vacuumdb"
Se acelera libpq por mejor asignacion de memoria(Tom)
EXPLAIN utiliza todos los indices(Tom)
Implementadas las expresiones CASE, COALESCE, NULLIF(Thomas)
Nuevo formato de salida de pg_dump(Constantin)
Añadida la cadena min()/max() a las funciones(Thomas)
Extendidas nuevo tipo de coersiones para agregaciones(Thomas)
Nueva contribucion moddatetime (Terry)
Actualizacion a pgaccess 0.96(Constantin)
```

```
Añadida rutina para byte unico en tipo de caracter "char"(Thomas)
Mejorada la funcion substr()(Thomas)
Mejorado el manejo de  multi-byte (Tatsuo)
Control de concurrencia Multi-version /MVCC(Vadim)
Nuevo modo Serialized(Vadim)
Correcion para tablas por encima de 2gigs(Peter)
Nuevo SET TRANSACTION ISOLATION LEVEL(Vadim)
Nuevo LOCK TABLE IN ... MODE(Vadim)
Actualizado el driver ODBC(Byron)
Nuevo tipo de datos NUMERIC(Jan)
Nueva SELECT FOR UPDATE(Vadim)
Manejo de "NaN" e "Infinity" para valores de entrada(Jan)
Mejorado el manejo de date/year(Thomas)
Mejorado el manejo de conexiones con el motor de base de datos(backend)(Magn
Nuevas opciones ELOG_TIMESTAMPS y la opcion USE_SYSLOG para fiche-
ros de registro(Massimo)
Nueva opcion TCL_ARRAYS (Massimo)
Nueva INTERSECT y EXCEPT(Stefan)
Nuevo pg_index.indisprimary para restro de claves primarias(D'Arcy)
Nuevas opcion pg_dump para permitir el borrado de tablas antes de su creacio
Acelaracion de las rutinas de salida de registro(Tom)
Nuevo nivel de aislamiento de READ COMMITTED (Vadim)
Nuevas tablas/indices TEMP (Bruce)
Se evita el ordenamiento si el resultado ya esta ordenado(Jan)
Nueva optimizacin para la asignacion de memoria(Jan)
Se permite a psql ejecutar \p\g(Bruce)
Se permiten multiples reglas de acciones(Jan)
Añadida funcionalidad LIMIT/OFFSET (Jan)
Mejorado el optimizador cuando se unen un numero grande de tablas(Bruce)
Nueva introduccion a SQL de La Tesis de Doctorado de S. Simkovics(Stefan, Th
Nueva introduccion a procesamiento del motor de base de datos (bac-
kend) de la Tesis de Doctorado de S. Simkovics(Stefan)
Mejorado el soporte para int8(Ryan Bradetich, Thomas, Tom)
Nuevas rutinas para convertir entre tipos int8 y text/varchar(Thomas)
Nuevos planes arboreos, donde se unen meta-tablas(Bruce)
Habilitadas consultas por la mano derecha por defecto(Bruce)
```

Se permite que el numero maximo de procesos en servidor (backends) se parametrice en el momento de la configuracion
(-with-maxbackends and postmaster switch (-N backends))(Tom)
GEQO por defecto tenga ahora 10 tablas porque el optimizador se acelera(Tom)
Se permite NULL=Var para MS-SQL portabilidad(Michael, Bruce)
Modificados contrib check_primary_key() y consecuentemente "automatic" or "dependent"(Anand)
Se permite que psql \d en una vista muestre la consulta(Ryan)
Se acelera el LIKE(Bruce)
Correciones/prestaciones Ecpg , vease fichero src/interfaces/ecpg/ChangeLog(
Correciones/prestacines JDBC , vease src/interfaces/jdbc/CHANGELOG(Peter)
Se hace que el operador % tenga precedencia como /(Bruce)
Añadida la nueva opcion postgres -O para permitir cambios en la estructura de tablas del sistema(Bruce)
Actualizado el script contrib/pginterface/findoidjoins (Tom)
Major aceleracion en vacuum de lineas borradas con indices(Vadim)
Se permitte la ejecucion de diferentes versiones de funciones noSQL basadas en argumentos(Tom)
Añadida la opcion -E que muestra las consultas actuales enviadas pro \dt y s
gos(Masaaki Sakaida)
Añadido numero de version en los banners de arranque de psql(Masaaki Sakaida
Nuevo contrib/vacuumlo que elimina objetos grandes no referenciados(Peter)
Nueva inicializacion para tamaños de tablas, asi las tablas no vacuumeadas se ejecutan mejor(Tom)
Mejorados los mensajes de error cuando una conexion es rechazada(Tom)
Soporte para array de campos char() y varchar() (Massimo)
Revision del codigo has para incrementar fiabilidad y prestaciones(Tom)
Actualizacion a PyGreSQL 2.4(D'Arcy)
Cambiadas las opciones de depuracion asi -d4 y -d5 producen diferentes displays de nodos(Jan)
nuevas opciones: pretty_plan, pretty_parse, pretty_rewritten(Jan)
Mejor optimizacion de estadisticas para los accesos a tablas del sistema(Tom
Mejor manejo de tamaño de bloque no por defecto(Massimo)
Mejorado el optmizador de comsumo de memoria GEQO(Tom)
UNION soporta ahora ORDER BY de columnas que no estan en la lista de target(
Mejoras grandes en libpq++ (Vince Vielhaber)
pg_dump utiliza ahora -z(ACL's) por omision(Bruce)

```
cache de procesos en servidor (backend), aceleracion de memoria(Tom)
Se hace que pg_dump lo haga todo en una transaccion snapshot(Vadim)
correcion de perdidas de memoria para objetos grandes, correccion pa-
ra pg_dumping(Tom)
INET escribe ahora respecto a la netmask para comparaciones
Se hace que VACUUM ANALYZE solo utilice un readlock(Vadim)
Se permiten vistas (VIEW) en UNIONS(Jan)
pg_dump puede generar ahora snapshots consistentes en bases de da-
tos activas(Vadim)

Cambios en el Arbol Fuente
--------------
Mejorado el emparejamiento en el porte(Tom)
Correcciones de portabilidad para SunOS
Añadido porte para NT/Win32 del proceso en el servidor (backend) y se ha-
bilita carga dinamica(Magnus and Daniel Horak)
Nuevo porte a Cobalt Qube(Mips) ejecutando Linux(Tatsuo)
Porte a NetBSD/m68k(Mr. Mutsuki Nakajima)
Porte a NetBSD/sun3(Mr. Mutsuki Nakajima)
Porte a NetBSD/macppc(Toshimi Aoki)
Correccion para configuracion de tcl/tk(Vince)
Eliminada la clave CURRENT para consultas por regla(Jan)
carga dinamica en NT ahora funciona(Daniel Horak)
Añadido soporte para ARM32(Andrew McMurry)
Mejor soporte para HPUX 11 y Unixware
Mejorado el manejo de ficheros para ser mas uniforme, previene la-
gunas en los descriptores de ficheros.(Tom)
Nuevos comandos de instalacion para plpgsql(Jan)
```

# 15.5. Version 6.4.2

La verson 6.4.1 fue incorrectamente empaquetada. Esta tiene tambien una correcion adicional para un bug

## 15.5.1. Migracion a v6.4.2

*No* se requiere un dump/restores para aquellos que esten ejecutando una 6.4.*.

## 15.5.2. Lista Detallada de Cambios

```
Correcion para problema de constante de fecha y hora en algunas pla-
taformas(Thomas)
```

# 15.6. Version 6.4.1

Esta es basicamente una limpieza de la version 6.4. Hemos corregio una variedad de problemas reportados por usuarios de 6.4

## 15.6.1. Migracion a v6.4.1

*No* se requiere un dump/restores para aquellos que esten ejecutando una 6.4.

## 15.6.2. Lista Detallada de Cambios

Añadida la opcion -N a pg_dump para forzar dobles comillas alrede-
dor de los identificadores. Este es la opcion por omision(Thomas)
Correccion para NOT in donde la clausula causa abortos(Bruce)
Correcion para el coredump de EXPLAIN VERBOSE (Vadim)
Correcion para problemas de shared-library en Linux
Correcion del test de existencia de tabla para permitir combinacion de ma-
yusculas y minusculas y
espacios en blanco en el nombre de tabla(Thomas)
Fijacion de un par de problemas de pg_dump
La configuracion empareja mejor entradas similares en la platilla tem-
plate/. (Tom)
Cambios en la construccion de los nombre de la funcion de SPI_* a spi_*
Correcion para la clausula OR WHERE(Vadim)
Correciones para nombres de tablas con mayusculas y minusculas(Billy)
Correcion para contrib/linux/postgres.init.csh/sh (Thomas)
rebasamiento de memoria en libpq corregido
Correciones para SunOS(Tom)
Se cambia el comportamiento de exp() para generar error en negativos(Thomas)
pg_dump fixes for memory leak, inheritance constraints, layout change
Actualizado pgaccess a 0.93
Se corrige el prototipo para plataformas de 64-bit
Correciones para Multi-byte(Tatsuo)
Nueva pagina de manual ecpg
Corregidos rebasamientos de memoria(Tatsuo)
Correcion para fallos en lo_import()(Bruce)
Mejores busquedas para el programa de instalacion(Tom)
Correciones para zona horaria(Tom)
Correciones para HPUX(Tom)
Se utiliza coersiones de tipo implicito para emparejar valores DEFAULT(Thoma
Anadidas rutinas para ayudar a tipo de caracter bytes-solo(single-
byte) (internal)(Thomas)
Correcciones para compilacion de libpq en Win32(Magnus)
Actualizacion a PyGreSQL 2.2(D’Arcy)

# 15.7. Version 6.4

Hay *muchas* prestaciones nuevas y mejoras en esta version. Gracias a unestros desarrolladores y mantenedores, casi todos los aspectos del sistema han recibido alguna atencion desde la version anterior. He aqui un resumen, sumario incompleto:

- Las vistas y las reglas son ahora funcionales gracias al extensivo nuevo codigo en las reglas de re escritura de Jan Wieck. Tambien escribio un capitulo sobre ello en la *Guia del Programador*.
- Jan tambien contribuyo al segundo lenguaje procedural, PL/pgSQL, para ir con el original lengaje procedural PL/pgTCL con el que el contribuyo la ultima version.
- Tenemos soporte opcional de caracter multiple-byte de Tatsuo Iisho para complementar nuestro soporte local.
- Las comunicaciones cliente/servidor han sido depuradas, con mejor soporte para mensajes asincronos e interrupciones, gracias a Tom Lane.
- El depurador de sintactico ejecuta ahora coersiones de tipo automatico para emparejar argumentos a los operadores y funciones disponibles, y para emparejar columnas y expresiones con columnas destino. Esto utiliza un mecanismo generico que soporta las prestaciones de extensibilidad de tipo de Postgres. Hay un nuevo capitulo en la *Guia de Usuario* que cubre este asunto.
- Tres nuevos tipos de datos han sido anadidos. Dos tipos, inet y cidr, soportan varias formas de trabajo en red IP, subred, y direccionamiento por maquina. Ahora hay un tipo entero de 8 byte disponible para algunas plataformas. Vease el capitulo de tipos de datos en la *Guia del Usuario* para mas detalles. Un cuarto tipo, serial, se soporta ahora por el depurador de sintactico como una amalgama de tipo int4, una secuencia, y un indice unico.
- Han sido añadidas varias prestaciones mas sintacticas compatibles con SQL92, incluyendo **INSERT DEFAULT VALUES** Several more SQL92-compatible syntax features have been added, including **INSERT DEFAULT VALUES**

- La instalacion y configuracion automatica del sistema ha recibido alguna atencio, y deberia ser mas robusta para mas plataformas de lo que nunca ha sido.

## 15.7.1. Migracion a v6.4

Se requiere un dump/restore utilizando pg_dump o pg_dumpall para aquellas que desen migrar datos desde cualquier version anterior de Postgres.

## 15.7.2. Lista Detallada de Cambios

```
Correciones de errores
------
Correcion para una minuscula perdida en PQsetdb/PQfinish(Bryan)
Se elimina char2-16 de los tipos de datos, se utiliza char/varchar(Darren)
Pqfn no maneja un mensaje de NOTICE(Anders)
Reducidas elevadas esperas por ocupacion a causa de bloqueos en tran-
sacciones  con muchos procesos en servidor(backends) (dg)
Deteccion de bloqueos de transacciones atascadas (dg)
Correcion para masrcas de tiempo en estilo "ISO" en decodificacion y codific
Correccion del problema con borrado de tabla (drop) despues de des-
hacer (rollback) una transaccion(Vadin)
Cambiado mensaje de error y eliminado mensaje actualizado no funcional(Vadim
Correccion para verificacion de la matriz (array) de COPY
Correccion para SELECT 1 UNION SELECT NULL
Correccion para perdidas de buffer en llamadas a objetos grandes(Pascal)
Cambio de propietario de tipo oid a int4(Bruce)
Correccion de error en la compatibilidad con oracle de las funcio-
nes btrim() ltrim() y rtrim()
Correccion de invalidacion en rebasamientos de cache compartida(Massimo)
Prevencion de perdidas en descriptores de ficheros en COPY's fallidos(Bruce)
Correccion para perdidas en la pg_select de libpgtcl(Constantin)
Correccion de problemas con usuario/contrasena de mas de 8 caracteres(Tom)
```

```
Correccion de problemas con manejo de NOTIFY asincronos en el pro-
ceso en servidor(backend)(Tom)
Correccion de muchas entradas de sistema malas(Tom)

Mejoras
--------
Actualizacion de ecpg y ecpglib, vease src/interfaces/ecpg/ChangeLog(Michael
Se muestra en indice utilizado en un EXPLAIN(Zeugswette)
EXPLAIN invoca una regla de sistema y muestra plan(es) para la rees-
critura de consultas(Jan)
Conocimiento multi-byte de muchos tipos de datos y funciones, via configure(
Nuevo configure con la opcion -with-mb(Tatsuo)
Nueva opcion initdb -pgencoding(Tatsuo)
Nueva opcion createdb -E multibyte(Tatsuo)
Select version(); ahora devuelve la version de PostgreSQL(Jeroen)
Libpq permite ahora clientes asincronos(Tom)
Se permite la cancelacion desde el cliente de una consulta en el pro-
ceso en servidor(backend)(Tom)
Psql cancelas las consultas ahora con Control-C(Tom)
Usuarios de Libpq no necesitan dar consultas dummy para obtener men-
sajes NOTIFY(Tom)
NOTIFY envia ahora al PID del emisor, asi que puedes decir si eras tu mismo(
La estructura de PGresult ahora incluye un mensaje de error asocia-
do, si lo hay(Tom)
Se definen los argumentos "tz_hour" y "tz_minute" como date_part()(Thomas)
Se anaden rutinas para convertir entre varchar y bpchar(Thomas)
Se anaden runtinas para permitir el dimensionamiento de
varchar y bpchar dentro de las columnas de destino(Thomas)
Se anade un bit a las etiquetas (flags) oara soportar
zona horaria y minutos en la devolucion de fecha(Thomas)
Se permiten mas variaciones en numeros de coma flotante (por ej. ".1", "1e6"
Se corrigen el analisis sintactico de menores unarios empezando con espacios
Se implementa TIMEZONE_HOUR, TIMEZONE_MINUTE por especificaciones SQL92(Thom
Se verifica i se ignora adecuadamente constraints de columna FOREIGN KEY(Tho
SE defina USER como sinonimo de CURRENT_USER por especificacines de SQL92(Th
Se habilita HAVING en clausulas pero no se corrije en ningun otro lu-
gar aun.
```

```
Se hace el tipo "char" un sinonimo de "char(1)" (actualmente imple-
mentado como bpchar)(Thomas)
Se guarda el tipo de cadena si esta especificado por el manejo de la clau-
sula DEFAULT(Thomas)
Operaciones de coercion abarcan diferentes tipos de datos(Thomas)
Se permite a algunos indices utilizar columnas de diferentes tipos(Thomas)
Anadidas capcidades para coersiones de tipo automatico(Thomas)
Depuraciones para objetos grandes, de este modo un fichero es trun-
cado en su apertura(Peter)
Depuraciones de la lectura de lineas(Tom)
Se permite que psql \f \ tomen los espacios en blanco como delimitadores(Bru
Se pasa el pg_attribute.atttypmod al frontal de la aplicacion pa-
ra las longitudes de los campos(Tom,Bruce)
Libreria de compatibilidad con Msql en /contrib(Aldrin)
Se elimina el requerimiento de que las clausulas identificadoras OR-
DER/GROUP BY
estuvieran incluidas en la lista de la busqueda(David)
Se convierten columnas para emparejarlas en las clausulas de UNION(Thomas)
Se elimina fork()/ecec() y solo se hace fork()(Bruce)
Depuraciones en Jdbc(Peter)
Se muestra el estado del proceso en el servidor (backend) en la
linea de comandos de ps(solo funciona en algunas plataformas)(Bruce)
Pg_hba.conf tiene ahora una opcion sameuser en el campo de la ba-
se de datos
Se hace que lo_unlink tome el parametro oid, no el int4
Nueva DISABLE_COMPLEX_MACRO para compiladores que no pueden mane-
jar nuestras macros(Bruce)
Libpgtcl maneja los NOTIFY ahora como un evento Tcl, no se necesi-
tan enviar consultas tontas(Tom)
Depuraciones en libpgtcl(Tom)
Anadida una opcion -error al comando pg_result de libpgtcl(Tom)
Anadido parche locale, vease docs/README/locale(Oleg)
Correccion para pg_dump con ella la sintaxis de CONSTRAINT y CHECK es correc
Nuevo codigo contrib/lo para eliminar grandes objetos huerfanos(Peter)
Nuevp comando psql "SET CLIENT_ENCODING TO 'encoding'" para prestaciones
multi.byte, vease /doc/README.mb(Tatsuo)
codigo /contrib/noupdate para revocar permisos de actualizacion en una colum
```

```
Libpq puede ser compilada ahora en win32(Magnus)
Anadido PQsetdbLogin() en libpq
Nuevo tipo entero 8-byte, comprobado por el configure del soporte pa-
ra OS(Thomas)
Mejor soporte para nombres entrecomillados de tabla/columnas(Thomas)
Se rodean los nombres de tabla y columnas con dobles comillas en pg_dump(Tho
PQreset() trabaja ahora con contrasenas(Tom)
Handle case of GROUP BY target list column number out of range(David)
Se permite UNION en las subconsultas
Anadido auto-dimensionamiento a la pantalla a los comandos \d?(Bruce)
Se utiliza UNION para mostrar todos los resultados de \d? en una consulta(Br
Se anade la prestacion de busqueda de campo \d?(Bruce)
Pg_dump utiliza menos peticiones \connect(Tom)
Se hace que la opcion -z de pg_dump trabaje mejor, se documenta en la pa-
gina de manual(Tom)
Se anade la clausula HAVING con total soporte para subconsultas y uniones(St
Full text indexing routines in contrib/fulltextindex(Maarten)
Los ids de transacciones se almacenan ahora en memoria compartida(Vadim)
Nuevo PGCLIENTENCODING cuando ejecutan el comando COPY(Tatsuo)
Soporte para la sintaxis SQL92 "SET NAMES"(Tatsuo)
Soporte para LATIN2-5(Tatsuo)
Anadido el caso UNICODE los test de refresion(Tatsuo)
Depuracion de gestor de bloqueos, nuevos modos de bloqueos para LLL(Vadim)
Se permite el uso de indice en clausulas OR(Bruce)
Se permite "SELECT NULL ORDER BY 1;"
La explicacion VERBOSE del plan lo imprime, y ahora imprime en bo-
nito el plan al
fichero de log del postmaster(Bruce)
Se anaden indices al display para el comando \d(Bruce)
Se permite el GROUP BY en funciones(David)
Nuevo pg_class.relkind para obejtos grandes(Bruce)
Nuevo modo de enviar libpq mensajes NOTICE a diferentes localizaciones(Tom)
Nuevo comando de escritura \w para psql(Bruce)
Nuevo /contrib/findoidjoins escanea columnas oid para encontrar re-
laciones de union(Bruce)
Se permite que sean considerados indices compatibles binarios cuan-
do se verifican
```

```
indices validos para una clausula de restriccion conteniendo una constante(T
Nuevo codigo ISBN/ISSN en /contrib/isbn_issn
Se permite NOT LIKE, IN, NOT IN, BETWEEN, y NOT BETWEEN constraint(Thomas)
Nuevo sistema de reescritura corrige muchos problemas con reglas y vistas(Ja
* Reglas en trabajos relacionados
* Event qualifications on insert/update/delete work
* New OLD variable to reference CURRENT, CURRENT will be remove in future
* Update rules can reference NEW and OLD in rule qualifications/actions
* Insert/update/delete rules on views work
* Multiple rule actions are now supported, surrounded by parentheses
* Regular users can create views/rules on tables they have RULE permits
* Rules and views inherit the permissions on the creator
* No rules at the column level
* No UPDATE NEW/OLD rules
* New pg_tables, pg_indexes, pg_rules and pg_views system views
* Only a single action on SELECT rules
* Total rewrite overhaul, perhaps for 6.5
* handle subselects
* handle aggregates on views
* handle insert into select from view works
System indexes are now multi-key(Bruce)
Oidint2, oidint4, and oidname types are removed(Bruce)
Use system cache for more system table lookups(Bruce)
New backend programming language PL/pgSQL in backend/pl(Jan)
New SERIAL data type, auto-creates sequence/index(Thomas)
Enable assert checking without a recompile(Massimo)
User lock enhancements(Massimo)
New setval() command to set sequence value(Massimo)
Auto-remove unix socket file on startup if no postmaster running(Massimo)
Conditional trace package(Massimo)
New UNLISTEN command(Massimo)
Psql and libpq now compile under win32 using win32.mak(Magnus)
Lo_read no longer stores trailing NULL(Bruce)
Identifiers are now truncated to 31 characters internally(Bruce)
Createuser options now availble on the command line
Code for 64-bit integer supported added, configure tested, int8 type(Thomas)
Prevent file descriptor leaf from failed COPY(Bruce)
```

```
New pg_upgrade command(Bruce)
Updated /contrib directories(Massimo)
New CREATE TABLE DEFAULT VALUES statement available(Thomas)
New INSERT INTO TABLE DEFAULT VALUES statement available(Thomas)
New DECLARE and FETCH feature(Thomas)
libpq's internal structures now not exported(Tom)
Allow up to 8 key indexes(Bruce)
Remove ARCHIVE keyword, that is no longer used(Thomas)
pg_dump -n flag to supress quotes around indentifiers
disable system columns for views(Jan)
new INET and CIDR types for network addresses(TomH, Paul)
no more double quotes in psql output
pg_dump now dumps views(Terry)
new SET QUERY_LIMIT(Tatsuo,Jan)

Source Tree Changes
-------------
/contrib cleanup(Jun)
Inline some small functions called for every row(Bruce)
Alpha/linux fixes
Hp/UX cleanups(Tom)
Multi-byte regression tests(Soonmyung.)
Remove -disabled options from configure
Define PGDOC to use POSTGRESDIR by default
Make regression optional
Remove extra braces code to pgindent(Bruce)
Add bsdi shared library support(Bruce)
New -without-CXX support configure option(Brook)
New FAQ_CVS
Update backend flowchart in tools/backend(Bruce)
Change atttypmod from int16 to int32(Bruce, Tom)
Getrusage() fix for platforms that do not have it(Tom)
Add PQconnectdb, PGUSER, PGPASSWORD to libpq man page
NS32K platform fixes(Phil Nelson, John Buller)
Sco 7/UnixWare 2.x fixes(Billy,others)
Sparc/Solaris 2.5 fixes(Ryan)
Pgbuiltin.3 is obsolete, move to doc files(Thomas)
```

```
Even more documention(Thomas)
Nextstep support(Jacek)
Aix support(David)
pginterface manual page(Bruce)
shared libraries all have version numbers
merged all OS-specific shared library defines into one file
smarter TCL/TK configuration checking(Billy)
smarter perl configuration(Brook)
configure uses supplied install-sh if no install script found(Tom)
new Makefile.shlib for shared library configuration(Tom)
```

## 15.8. Release 6.3.2

This is a bugfix release for 6.3.x. Refer to the release notes for v6.3 for a more complete summary of new features.

Summary:

- Repairs automatic configuration support for some platforms, including Linux, from breakage inadvertently introduced in v6.3.1.
- Correctly handles function calls on the left side of BETWEEN and LIKE clauses.

A dump/restore is NOT required for those running 6.3 or 6.3.1. A ’make distclean’, ’make’, and ’make install’ is all that is required. This last step should be performed while the postmaster is not running. You should re-link any custom applications that use Postgres libraries.

For upgrades from pre-v6.3 installations, refer to the installation and migration instructions for v6.3.

### 15.8.1. Detailed Change List

```
Changes
-----
Configure detection improvements for tcl/tk(Brook Milligan, Alvin)
Manual page improvements(Bruce)
BETWEEN and LIKE fix(Thomas)
fix for psql \connect used by pg_dump(Oliver Elphick)
New odbc driver
pgaccess, version 0.86
qsort removed, now uses libc version, cleanups(Jeroen)
fix for buffer over-runs detected(Maurice Gittens)
fix for buffer overrun in libpgtcl(Randy Kunkee)
fix for UNION with DISTINCT or ORDER BY(Bruce)
gettimeofday configure check(Doug Winterburn)
Fix "indexes not used" bug(Vadim)
docs additions(Thomas)
Fix for backend memory leak(Bruce)
libreadline cleanup(Erwan MAS)
Remove DISTDIR(Bruce)
Makefile dependency cleanup(Jeroen van Vianen)
ASSERT fixes(Bruce)
```

## 15.9. Release 6.3.1

Summary:

- Additional support for multi-byte character sets.
- Repair byte ordering for mixed-endian clients and servers.

- Minor updates to allowed SQL syntax.
- Improvements to the configuration autodetection for installation.

A dump/restore is NOT required for those running 6.3. A ’make distclean’, ’make’, and ’make install’ is all that is required. This last step should be performed while the postmaster is not running. You should re-link any custom applications that use Postgres libraries.

For upgrades from pre-v6.3 installations, refer to the installation and migration instructions for v6.3.

## 15.9.1. Detailed Change List

```
Changes
-----
ecpg cleanup/fixes, now version 1.1(Michael Meskes)
pg_user cleanup(Bruce)
large object fix for pg_dump and tclsh (alvin)
LIKE fix for multiple adjacent underscores
fix for redefining builtin functions(Thomas)
ultrix4 cleanup
upgrade to pg_access 0.83
updated CLUSTER manual page
multi-byte character set support, see doc/README.mb(Tatsuo)
configure -with-pgport fix
pg_ident fix
big-endian fix for backend communications(Kataoka)
SUBSTR() and substring() fix(Jan)
several jdbc fixes(Peter)
libpgtcl improvements, see libptcl/README(Randy Kunkee)
Fix for "Datasize = 0" error(Vadim)
Prevent \do from wrapping(Bruce)
Remove duplicate Russian character set entries
Sunos4 cleanup
```

```
Allow optional TABLE keyword in LOCK and SELECT INTO(Thomas)
CREATE SEQUENCE options to allow a negative integer(Thomas)
Add "PASSWORD" as an allowed column identifier(Thomas)
Add checks for UNION target fields(Bruce)
Fix Alpha port(Dwayne Bailey)
Fix for text arrays containing quotes(Doug Gibson)
Solaris compile fix(Albert Chin-A-Young)
Better identify tcl and tk libs and includes(Bruce)
```

## 15.10. Release 6.3

There are *many* new features and improvements in this release. Here is a brief, incomplete summary:

- Many new SQL features, including full SQL92 subselect capability (everything is here but target-list subselects).
- Support for client-side environment variables to specify time zone and date style.
- Socket interface for client/server connection. This is the default now so you may need to start postmaster with the “-i” flag.
- Better password authorization mechanisms. Default table permissions have changed.
- Old-style “time travel” has been removed. Performance has been improved.

**Nota:** Bruce Momjian wrote the following notes to introduce the new release.

There are some general 6.3 issues that I want to mention. These are only the big items that can not be described in one sentence. A review of the detailed changes list is still needed.

First, we now have subselects. Now that we have them, I would like to mention that without subselects, SQL is a very limited language. Subselects are a major feature, and you should review your code for places where subselects provide a better solution for your queries. I think you will find that there are more uses for subselects than you may think. Vadim has put us on the big SQL map with subselects, and fully functional ones too. The only thing you can’t do with subselects is to use them in the target list.

Second, 6.3 uses unix domain sockets rather than TCP/IP by default. To enable connections from other machines, you have to use the new postmaster -i option, and of course edit pg_hba.conf. Also, for this reason, the format of pg_hba.conf has changed.

Third, char() fields will now allow faster access than varchar() or text. Specifically, the text and varchar() have a penalty for access to any columns after the first column of this type. char() used to also have this access penalty, but it no longer does. This may suggest that you redesign some of your tables, especially if you have short character columns that you have defined as varchar() or text. This and other changes make 6.3 even faster than earlier releases.

We now have passwords definable independent of any Unix file. There are new SQL USER commands. See the pg_hba.conf manual page for more information. There is a new table, pg_shadow, which is used to store user information and user passwords, and it by default only SELECT-able by the postgres super-user. pg_user is now a view of pg_shadow, and is SELECT-able by PUBLIC. You should keep using pg_user in your application without changes.

User-created tables now no longer have SELECT permission to PUBLIC by default. This was done because the ANSI standard requires it. You can of course GRANT any permissions you want after the table is created. System tables continue to be SELECT-able by PUBLIC.

We also have real deadlock detection code. No more sixty-second timeouts. And the new locking code implements a FIFO better, so there should be less resource starvation during heavy use.

Many complaints have been made about inadequate documenation in previous releases. Thomas has put much effort into many new manuals for this release. Check out the doc/ directory.

For performance reasons, time travel is gone, but can be implemented using triggers (see pgsql/contrib/spi/README). Please check out the new \d command for types, operators, etc. Also, views have their own permissions now, not based on the underlying tables, so permissions on them have to be set separately. Check /pgsql/interfaces for some new ways to talk to Postgres.

This is the first release that really required an explanation for existing users. In many ways, this was necessary because the new release removes many limitations, and the work-arounds people were using are no longer needed.

## 15.10.1. Migration to v6.3

A dump/restore using pg_dump or pg_dumpall is required for those wishing to migrate data from any previous release of Postgres.

## 15.10.2. Detailed Change List

```
Bug Fixes
------
Fix binary cursors broken by MOVE implementation(Vadim)
Fix for tcl library crash(Jan)
Fix for array handling, from Gerhard Hintermayer
Fix acl error, and remove duplicate pqtrace(Bruce)
Fix psql \e for empty file(Bruce)
Fix for textcat on varchar() fields(Bruce)
Fix for DBT Sendproc (Zeugswetter Andres)
Fix vacuum analyze syntax problem(Bruce)
Fix for international identifiers(Tatsuo)
Fix aggregates on inherited tables(Bruce)
Fix substr() for out-of-bounds data
```

```
Fix for select 1=1 or 2=2, select 1=1 and 2=2, and select sum(2+2)(Bruce)
Fix notty output to show status result.  -q option still turns it off(Bruce)
Fix for count(*), aggs with views and multiple tables and sum(3)(Bruce)
Fix cluster(Bruce)
Fix for PQtrace start/stop several times(Bruce)
Fix a variety of locking problems like newer lock waiters getting
lock before older waiters, and having readlock people not share
locks if a writer is waiting for a lock, and waiting writers not
getting priority over waiting readers(Bruce)
Fix crashes in psql when executing queries from external files(James)
Fix problem with multiple order by columns, with the first one having
NULL values(Jeroen)
Use correct hash table support functions for float8 and int4(Thomas)
Re-enable JOIN= option in CREATE OPERATOR statement (Thomas)
Change precedence for boolean operators to match expected behavior(Thomas)
Generate elog(ERROR) on over-large integer(Bruce)
Allow multiple-argument functions in constraint clauses(Thomas)
Check boolean input literals for 'true','false','yes','no','1','0'
and throw elog(ERROR) if unrecognized(Thomas)
Major large objects fix
Fix for GROUP BY showing duplicates(Vadim)
Fix for index scans in MergeJion(Vadim)

Enhancements
--------
Subselects with EXISTS, IN, ALL, ANY keywords (Vadim, Bruce, Thomas)
New User Manual(Thomas, others)
Speedup by inlining some frequently-called functions
Real deadlock detection, no more timeouts(Bruce)
Add SQL92 "constants" CURRENT_DATE, CURRENT_TIME, CURRENT_TIMESTAMP,
CURRENT_USER(Thomas)
Modify constraint syntax to be SQL92-compliant(Thomas)
Implement SQL92 PRIMARY KEY and UNIQUE clauses using indices(Thomas)
Recognize SQL92 syntax for FOREIGN KEY. Throw elog notice(Thomas)
Allow NOT NULL UNIQUE constraint clause (each allowed separately before)(Tho
Allow Postgres-style casting ("::") of non-constants(Thomas)
Add support for SQL3 TRUE and FALSE boolean constants(Thomas)
```

```
Support SQL92 syntax for IS TRUE/IS FALSE/IS NOT TRUE/IS NOT FALSE(Thomas)
Allow shorter strings for boolean literals (e.g. "t", "tr", "tru")(Thomas)
Allow SQL92 delimited identifiers(Thomas)
Implement SQL92 binary and hexadecimal string decoding (b'10' and x'1F')(Tho
Support SQL92 syntax for type coercion of literal strings
(e.g. "DATETIME 'now'")(Thomas)
Add conversions for int2, int4, and OID types to and from text(Thomas)
Use shared lock when building indices(Vadim)
Free memory allocated for an user query inside transaction block after
this query is done, was turned off in <= 6.2.1(Vadim)
New SQL statement CREATE PROCEDURAL LANGUAGE(Jan)
New Postgres Procedural Language (PL) backend interface(Jan)
Rename pg_dump -H option to -h(Bruce)
Add Java support for passwords, European dates(Peter)
Use indices for LIKE and ~, !~ operations(Bruce)
Add hash functions for datetime and timespan(Thomas)
Time Travel removed(Vadim, Bruce)
Add paging for \d and \z, and fix \i(Bruce)
Add Unix domain socket support to backend and to frontend library(Goran)
Implement CREATE DATABASE/WITH LOCATION and initlocation utility(Thomas)
Allow more SQL92 and/or Postgres reserved words as column identifiers(Thomas
Augment support for SQL92 SET TIME ZONE...(Thomas)
SET/SHOW/RESET TIME ZONE uses TZ backend environment variable(Thomas)
Implement SET keyword = DEFAULT and SET TIME ZONE DEFAULT(Thomas)
Enable SET TIME ZONE using TZ environment variable(Thomas)
Add PGDATESTYLE environment variable to frontend and backend initialization(
Add PGTZ, PGCOSTHEAP, PGCOSTINDEX, PGRPLANS, PGGEQO
frontend library initialization environment variables(Thomas)
Regression tests time zone automatically set with "setenv PGTZ PST8PDT"(Thom
Add pg_description table for info on tables, columns, operators, ty-
pes, and
aggregates(Bruce)
Increase 16 char limit on system table/index names to 32 characters(Bruce)
Rename system indices(Bruce)
Add 'GERMAN' option to SET DATESTYLE(Thomas)
Define an "ISO-style" timespan output format with "hh:mm:ss" fields(Thomas)
Allow fractional values for delta times (e.g. '2.5 days')(Thomas)
```

```
Validate numeric input more carefully for delta times(Thomas)
Implement day of year as possible input to date_part()(Thomas)
Define timespan_finite() and text_timespan() functions(Thomas)
Remove archive stuff(Bruce)
Allow for a pg_password authentication database that is separate from
the system password file(Todd)
Dump ACLs, GRANT, REVOKE permissions(Matt)
Define text, varchar, and bpchar string length functions(Thomas)
Fix Query handling for inheritance, and cost computations(Bruce)
Implement CREATE TABLE/AS SELECT (alternative to SELECT/INTO)(Thomas)
Allow NOT, IS NULL, IS NOT NULL in constraints(Thomas)
Implement UNIONs for SELECT(Bruce)
Add UNION, GROUP, DISTINCT to INSERT(Bruce)
varchar() stores only necessary bytes on disk(Bruce)
Fix for BLOBs(Peter)
Mega-Patch for JDBC...see README_6.3 for list of changes(Peter)
Remove unused "option" from PQconnectdb()
New LOCK command and lock manual page describing deadlocks(Bruce)
Add new psql \da, \dd, \df, \do, \dS, and \dT commands(Bruce)
Enhance psql \z to show sequences(Bruce)
Show NOT NULL and DEFAULT in psql \d table(Bruce)
New psql .psqlrc file startup(Andrew)
Modify sample startup script in contrib/linux to show syslog(Thomas)
New types for IP and MAC addresses in contrib/ip_and_mac(TomH)
Unix system time conversions with date/time types in contrib/unixdate(Thomas
Update of contrib stuff(Massimo)
Add Unix socket support to DBD::Pg(Goran)
New python interface (PyGreSQL 2.0)(D'Arcy)
New frontend/backend protocol has a version number, network byte order(Phil)
Security features in pg_hba.conf enhanced and documented, many cleanups(Phil
CHAR() now faster access than VARCHAR() or TEXT
ecpg embedded SQL preprocessor
Reduce system column overhead(Vadmin)
Remove pg_time table(Vadim)
Add pg_type attribute to identify types that need length (bpchar, varchar)
Add report of offending line when COPY command fails
Allow VIEW permissions to be set separately from the underlying tables.
```

```
For security, use GRANT/REVOKE on views as appropriate(Jan)
Tables now have no default GRANT SELECT TO PUBLIC.  You must
explicitly grant such permissions.
Clean up tutorial examples(Darren)

Source Tree Changes
-------------
Add new html development tools, and flow chart in /tools/backend
Fix for SCO compiles
Stratus computer port Robert Gillies
Added support for shlib for BSD44_derived & i386_solaris
Make configure more automated(Brook)
Add script to check regression test results
Break parser functions into smaller files, group together(Bruce)
Rename heap_create to heap_create_and_catalog, rename heap_creatr
to heap_create()(Bruce)
Sparc/Linux patch for locking(TomS)
Remove PORTNAME and reorganize port-specific stuff(Marc)
Add optimizer README file(Bruce)
Remove some recursion in optimizer and clean up some code there(Bruce)
Fix for NetBSD locking(Henry)
Fix for libptcl make(Tatsuo)
AIX patch(Darren)
Change IS TRUE, IS FALSE, ... to expressions using "=" rather than
function calls to istrue() or isfalse() to allow optimization(Thomas)
Various fixes NetBSD/Sparc related(TomH)
Alpha linux locking(Travis,Ryan)
Change elog(WARN) to elog(ERROR)(Bruce)
FAQ for FreeBSD(Marc)
Bring in the PostODBC source tree as part of our standard distribution(Marc)
A minor patch for HP/UX 10 vs 9(Stan)
New pg_attribute.atttypmod for type-specific info like varchar length(Bruce)
Unixware patches(Billy)
New i386 'lock' for spin lock asm(Billy)
Support for multiplexed backends is removed
Start an OpenBSD port
Start an AUX port
```

```
Start a Cygnus port
Add string functions to regression suite(Thomas)
Expand a few function names formerly truncated to 16 characters(Thomas)
Remove un-needed malloc() calls and replace with palloc()(Bruce)
```

# 15.11. Release 6.2.1

v6.2.1 is a bug-fix and usability release on v6.2.

Summary:

- Allow strings to span lines, per SQL92.
- Include example trigger function for inserting user names on table updates.

This is a minor bug-fix release on v6.2. For upgrades from pre-v6.2 systems, a full dump/reload is required. Refer to the v6.2 release notes for instructions.

## 15.11.1. Migration from v6.2 to v6.2.1

This is a minor bug-fix release. A dump/reload is not required from v6.2, but is required from any release prior to v6.2.

In upgrading from v6.2, if you choose to dump/reload you will find that avg(money) is now calculated correctly. All other bug fixes take effect upon updating the executables.

Another way to avoid dump/reload is to use the following SQL command from psql to update the existing system table:

```
update pg_aggregate set aggfinalfn = 'cash_div_flt8'
 where aggname = 'avg' and aggbasetype = 790;
```

This will need to be done to every existing database, including template1.

### 15.11.2. Detailed Change List

```
Changes in this release
---------------
Allow TIME and TYPE column names(Thomas)
Allow larger range of true/false as boolean values(Thomas)
Support output of "now" and "current"(Thomas)
Handle DEFAULT with INSERT of NULL properly(Vadim)
Fix for relation reference counts problem in buffer manager(Vadim)
Allow strings to span lines, like ANSI(Thomas)
Fix for backward cursor with ORDER BY(Vadim)
Fix avg(cash) computation(Thomas)
Fix for specifying a column twice in ORDER/GROUP BY(Vadim)
Documented new libpq function to return affected rows, PQcmdTuples(Bruce)
Trigger function for inserting user names for INSERT/UPDATE(Brook Mi-
lligan)
```

## 15.12. Release 6.2

A dump/restore is required for those wishing to migrate data from previous releases of Postgres.

## 15.12.1. Migration from v6.1 to v6.2

This migration requires a complete dump of the 6.1 database and a restore of the database in 6.2.

Note that the pg_dump and pg_dumpall utility from 6.2 should be used to dump the 6.1 database.

## 15.12.2. Migration from v1.x to v6.2

Those migrating from earlier 1.* releases should first upgrade to 1.09 because the COPY output format was improved from the 1.02 release.

## 15.12.3. Detailed Change List

```
Bug Fixes
------
Fix problems with pg_dump for inheritance, sequences, archive tables(Bruce)
Fix compile errors on overflow due to shifts, unsigned, and bad prototypes
 from Solaris(Diab Jerius)
Fix bugs in geometric line arithmetic (bad intersection calculations)(Thomas
Check for geometric intersections at endpoints to avoid rounding ugliness(Th
Catch non-functional delete attempts(Vadim)
Change time function names to be more consistent(Michael Reifenberg)
Check for zero divides(Michael Reifenberg)
Fix very old bug which made tuples changed/inserted by a commnd
visible to the command itself (so we had multiple update of
updated tuples, etc)(Vadim)
Fix for SELECT null, 'fail' FROM pg_am (Patrick)
SELECT NULL as EMPTY_FIELD now allowed(Patrick)
Remove un-needed signal stuff from contrib/pginterface
Fix OR (where x != 1 or x isnull didn't return tuples with x NULL) (Vadim)
Fix time_cmp function (Vadim)
Fix handling of functions with non-attribute first argument in
```

```
WHERE clauses (Vadim)
Fix GROUP BY when order of entries is different from order
in target list (Vadim)
Fix pg_dump for aggregates without sfunc1 (Vadim)

Enhancements
--------
Default genetic optimizer GEQO parameter is now 8(Bruce)
Allow use parameters in target list having aggregates in functions(Vadim)
Added JDBC driver as an interface(Adrian & Peter)
pg_password utility
Return number of tuples inserted/affected by INSERT/UPDATE/DELETE etc.(Vadim
Triggers implemented with CREATE TRIGGER (SQL3)(Vadim)
SPI (Server Programming Interface) allows execution of queries inside
C-functions (Vadim)
NOT NULL implemented (SQL92)(Robson Paniago de Miranda)
Include reserved words for string handling, outer joins, and unions(Thomas)
Implement extended comments ("/* ... */") using exclusive states(Thomas)
Add "//" single-line comments(Bruce)
Remove some restrictions on characters in operator names(Thomas)
DEFAULT and CONSTRAINT for tables implemented (SQL92)(Vadim & Thomas)
Add text concatenation operator and function (SQL92)(Thomas)
Support WITH TIME ZONE syntax (SQL92)(Thomas)
Support INTERVAL unit TO unit syntax (SQL92)(Thomas)
Define types DOUBLE PRECISION, INTERVAL, CHARACTER,
and CHARACTER VARYING (SQL92)(Thomas)
Define type FLOAT(p) and rudimentary DECIMAL(p,s), NUMERIC(p,s) (SQL92)(Thom
Define EXTRACT(), POSITION(), SUBSTRING(), and TRIM() (SQL92)(Thomas)
Define CURRENT_DATE, CURRENT_TIME, CURRENT_TIMESTAMP (SQL92)(Thomas)
Add syntax and warnings for UNION, HAVING, INNER and OUTER JOIN (SQL92)(Thom
Add more reserved words, mostly for SQL92 compliance(Thomas)
Allow hh:mm:ss time entry for timespan/reltime types(Thomas)
Add center() routines for lseg, path, polygon(Thomas)
Add distance() routines for circle-polygon, polygon-polygon(Thomas)
Check explicitly for points and polygons contained within polygons
using an axis-crossing algorithm(Thomas)
Add routine to convert circle-box(Thomas)
```

```
Merge conflicting operators for different geometric data types(Thomas)
Replace distance operator "<===>" with "<->"(Thomas)
Replace "above" operator "!^" with ">^" and "below" operator "!|" with "<^"(
Add routines for text trimming on both ends, substring, and string position(
Added conversion routines circle(box) and poly(circle)(Thomas)
Allow internal sorts to be stored in memory rather than in files(Bruce & Vad
Allow functions and operators on internally-identical types to succeed(Bruce
Speed up backend startup after profiling analysis(Bruce)
Inline frequently called functions for performance(Bruce)
Reduce open() calls(Bruce)
psql:  Add PAGER for \h and \?,\C fix
Fix for psql pager when no tty(Bruce)
New entab utility(Bruce)
General trigger functions for referential integrity (Vadim)
General trigger functions for time travel (Vadim)
General trigger functions for AUTOINCREMENT/IDENTITY feature (Vadim)
MOVE implementation (Vadim)

Source Tree Changes
-------------
HPUX 10 patches (Vladimir Turin)
Added SCO support, (Daniel Harris)
mkLinux patches (Tatsuo Ishii)
Change geometric box terminology from "length" to "width"(Thomas)
Deprecate temporary unstored slope fields in geometric code(Thomas)
Remove restart instructions from INSTALL(Bruce)
Look in /usr/ucb first for install(Bruce)
Fix c++ copy example code(Thomas)
Add -o to psql manual page(Bruce)
Prevent relname unallocated string length from being copied into database(Br
Cleanup for NAMEDATALEN use(Bruce)
Fix pg_proc names over 15 chars in output(Bruce)
Add strNcpy() function(Bruce)
remove some (void) casts that are unnecessary(Bruce)
new interfaces directory(Marc)
Replace fopen() calls with calls to fd.c functions(Bruce)
Make functions static where possible(Bruce)
```

```
enclose unused functions in #ifdef NOT_USED(Bruce)
Remove call to difftime() in timestamp support to fix SunOS(Bruce & Thomas)
Changes for Digital Unix
Portability fix for pg_dumpall(Bruce)
Rename pg_attribute.attnvals to attdisbursion(Bruce)
"intro/unix" manual page now "pgintro"(Bruce)
"built-in" manual page now "pgbuiltin"(Bruce)
"drop" manual page now "drop_table"(Bruce)
Add "create_trigger", "drop_trigger" manual pages(Thomas)
Add constraints regression test(Vadim & Thomas)
Add comments syntax regression test(Thomas)
Add PGINDENT and support program(Bruce)
Massive commit to run PGINDENT on all *.c and *.h files(Bruce)
Files moved to /src/tools directory(Bruce)
SPI and Trigger programming guides (Vadim & D'Arcy)
```

# 15.13. Release 6.1.1

## 15.13.1. Migration from v6.1 to v6.1.1

This is a minor bug-fix release. A dump/reload is not required from v6.1, but is required from any release prior to v6.1. Refer to the release notes for v6.1 for more details.

## 15.13.2. Detailed Change List

```
Changes in this release
---------------
fix for SET with options (Thomas)
allow pg_dump/pg_dumpall to preserve ownership of all tables/objects(Bruce)
```

```
new psql \connect option allows changing usernames without changing database
fix for initdb -debug option(Yoshihiko Ichikawa))
lextest cleanup(Bruce)
hash fixes(Vadim)
fix date/time month boundary arithmetic(Thomas)
fix timezone daylight handling for some ports(Thomas, Bruce, Tatsuo)
timestamp overhauled to use standard functions(Thomas)
other code cleanup in date/time routines(Thomas)
psql’s \d now case-insensitive(Bruce)
psql’s backslash commands can now have trailing semicolon(Bruce)
fix memory leak in psql when using \g(Bruce)
major fix for endian handling of communication to server(Thomas, Tatsuo)
Fix for Solaris assembler and include files(Yoshihiko Ichikawa)
allow underscores in usernames(Bruce)
pg_dumpall now returns proper status, portability fix(Bruce)
```

## 15.14. Release 6.1

The regression tests have been adapted and extensively modified for the v6.1 release of Postgres.

Three new data types (datetime, timespan, and circle) have been added to the native set of Postgres types. Points, boxes, paths, and polygons have had their output formats made consistant across the data types. The polygon output in misc.out has only been spot-checked for correctness relative to the original regression output.

Postgres v6.1 introduces a new, alternate optimizer which uses *genetic* algorithms. These algorithms introduce a random behavior in the ordering of query results when the query contains multiple qualifiers or multiple tables (giving the optimizer a choice on order of evaluation). Several regression tests have been modified to explicitly order the results, and hence are insensitive to optimizer choices. A few regression tests are for

data types which are inherently unordered (e.g. points and time intervals) and tests involving those types are explicitly bracketed with **set geqo to 'off'** and **reset geqo**.

The interpretation of array specifiers (the curly braces around atomic values) appears to have changed sometime after the original regression tests were generated. The current `./expected/*.out` files reflect this new interpretation, which may not be correct!

The float8 regression test fails on at least some platforms. This is due to differences in implementations of pow() and exp() and the signaling mechanisms used for overflow and underflow conditions.

The "random" results in the random test should cause the "random" test to be "failed", since the regression tests are evaluated using a simple diff. However, "random" does not seem to produce random results on my test machine (Linux/gcc/i686).

## 15.14.1. Migration to v6.1

This migration requires a complete dump of the 6.0 database and a restore of the database in 6.1.

Those migrating from earlier 1.* releases should first upgrade to 1.09 because the COPY output format was improved from the 1.02 release.

## 15.14.2. Detailed Change List

```
Bug Fixes
------
packet length checking in library routines
lock manager priority patch
check for under/over flow of float8(Bruce)
multi-table join fix(Vadim)
SIGPIPE crash fix(Darren)
large object fixes(Sven)
allow btree indexes to handle NULLs(Vadim)
timezone fixes(D'Arcy)
```

```
select SUM(x) can return NULL on no rows(Thomas)
internal optimizer, executor bug fixes(Vadim)
fix problem where inner loop in < or <= has no rows(Vadim)
prevent re-commuting join index clauses(Vadim)
fix join clauses for multiple tables(Vadim)
fix hash, hashjoin for arrays(Vadim)
fix btree for abstime type(Vadim)
large object fixes(Raymond)
fix buffer leak in hash indices (Vadim)
fix rtree for use in inner scan (Vadim)
fix gist for use in inner scan, cleanups (Vadim, Andrea)
avoid unnecessary local buffers allocation (Vadim, Massimo)
fix local buffers leak in transaction aborts (Vadim)
fix file manager memmory leaks, cleanups (Vadim, Massimo)
fix storage manager memmory leaks (Vadim)
fix btree duplicates handling (Vadim)
fix deleted tuples re-incarnation caused by vacuum (Vadim)
fix SELECT varchar()/char() INTO TABLE made zero-length fields(Bruce)
many psql, pg_dump, and libpq memory leaks fixed using Purify (Igor)

Enhancements
--------
attribute optimization statistics(Bruce)
much faster new btree bulk load code(Paul)
BTREE UNIQUE added to bulk load code(Vadim)
new lock debug code(Massimo)
massive changes to libpg++(Leo)
new GEQO optimizer speeds table multi-table optimization(Martin)
new WARN message for non-unique insert into unique key(Marc)
update x=-3, no spaces, now valid(Bruce)
remove case-sensitive identifier handling(Bruce,Thomas,Dan)
debug backend now pretty-prints tree(Darren)
new Oracle character functions(Edmund)
new plaintext password functions(Dan)
no such class or insufficient privilege changed to distinct messages(Dan)
new ANSI timestamp function(Dan)
new ANSI Time and Date types (Thomas)
```

```
move large chunks of data in backend(Martin)
multi-column btree indexes(Vadim)
new SET var TO value command(Martin)
update transaction status on reads(Dan)
new locale settings for character types(Oleg)
new SEQUENCE serial number generator(Vadim)
GROUP BY function now possible(Vadim)
re-organize regression test(Thomas,Marc)
new optimizer operation weights(Vadim)
new psql \z grant/permit option(Marc)
new MONEY data type(D'Arcy,Thomas)
tcp socket communication speed improved(Vadim)
new VACUUM option for attribute statistics, and for certain columns (Vadim)
many geometric type improvements(Thomas,Keith)
additional regression tests(Thomas)
new datestyle variable(Thomas,Vadim,Martin)
more comparison operators for sorting types(Thomas)
new conversion functions(Thomas)
new more compact btree format(Vadim)
allow pg_dumpall to preserve database ownership(Bruce)
new SET GEQO=# and R_PLANS variable(Vadim)
old (!GEQO) optimizer can use right-sided plans (Vadim)
typechecking improvement in SQL parser(Bruce)
new SET, SHOW, RESET commands(Thomas,Vadim)
new \connect database USER option
new destroydb -i option (Igor)
new \dt and \di psql commands (Darren)
SELECT "\n" now escapes newline (A. Duursma)
new geometry conversion functions from old format (Thomas)

Source tree changes
-------------
new configuration script(Marc)
readline configuration option added(Marc)
OS-specific configuration options removed(Marc)
new OS-specific template files(Marc)
no more need to edit Makefile.global(Marc)
```

```
re-arrange include files(Marc)
nextstep patches (Gregor Hoffleit)
removed WIN32-specific code(Bruce)
removed postmaster -e option, now only postgres -e option (Bruce)
merge duplicate library code in front/backends(Martin)
now works with eBones, international Kerberos(Jun)
more shared library support
c++ include file cleanup(Bruce)
warn about buggy flex(Bruce)
DG-UX, Ultrix, Irix, AIX portability fixes
```

## 15.15. Release v6.0

A dump/restore is required for those wishing to migrate data from previous releases of Postgres.

### 15.15.1. Migration from v1.09 to v6.0

This migration requires a complete dump of the 1.09 database and a restore of the database in 6.0.

### 15.15.2. Migration from pre-v1.09 to v6.0

Those migrating from earlier 1.* releases should first upgrade to 1.09 because the COPY output format was improved from the 1.02 release.

## 15.15.3. Detailed Change List

```
Bug Fixes
------
ALTER TABLE bug - running postgress process needs to re-read table definitio
Allow vacuum to be run on one table or entire database(Bruce)
Array fixes
Fix array over-runs of memory writes(Kurt)
Fix elusive btree range/non-range bug(Dan)
Fix for hash indexes on some types like time and date
Fix for pg_log size explosion
Fix permissions on lo_export()(Bruce)
Fix unitialized reads of memory(Kurt)
Fixed ALTER TABLE ... char(3) bug(Bruce)
Fixed a few small memory leaks
Fixed EXPLAIN handling of options and changed full_path option name
Fixed output of group acl permissions
Memory leaks (hunt and destroy with tools like Purify(Kurt)
Minor improvements to rules system
NOTIFY fixes
New asserts for run-checking
Overhauled parser/analyze code to properly report errors and increa-
se speed
Pg_dump -d now handles NULL's properly(Bruce)
Prevent SELECT NULL from crashing server (Bruce)
Properly report errors when INSERT ... SELECT columns did not match
Properly report errors when insert column names were not correct
Psql \g filename now works(Bruce)
Psql fixed problem with multiple statements on one line with mul-
tiple outputs
Removed duplicate system oid's
SELECT * INTO TABLE . GROUP/ORDER BY gives unlink error if table exists(Bruc
Several fixes for queries that crashed the backend
Starting quote in insert string errors(Bruce)
Submitting an empty query now returns empty status, not just " " query(Bruce

Enhancements
```

```
--------
Add EXPLAIN manual page(Bruce)
Add UNIQUE index capability(Dan)
Add hostname/user level access control rather than just hostname and user
Add synonym of != for <>(Bruce)
Allow "select oid,* from table"
Allow BY,ORDER BY to specify columns by number, or by non-alias table.column
Allow COPY from the frontend(Bryan)
Allow GROUP BY to use alias column name(Bruce)
Allow actual compression, not just reuse on the same page(Vadim)
Allow installation-configuration option to auto-add all local users(Bryan)
Allow libpq to distinguish between text value ” and null(Bruce)
Allow non-postgres users with createdb privs to destroydb’s
Allow restriction on who can create C functions(Bryan)
Allow restriction on who can do backend COPY(Bryan)
Can shrink tables, pg_time and pg_log(Vadim & Erich)
Change debug level 2 to print queries only, changed debug heading layout(Bru
Change default decimal constant representation from float4 to float8(Bruce)
European date format now set when postmaster is started
Execute lowercase function names if not found with exact case
Fixes for aggregate/GROUP processing, allow ’select sum(func(x),sum(x+y) fro
Gist now included in the distrubution(Marc)
Idend authentication of local users(Bryan)
Implement BETWEEN qualifier(Bruce)
Implement IN qualifier(Bruce)
Libpq has PQgetisnull()(Bruce)
Libpq++ improvements
New options to initdb(Bryan)
Pg_dump allow dump of oid’s(Bruce)
Pg_dump create indexes after tables are loaded for speed(Bruce)
Pg_dumpall dumps all databases, and the user table
Pginterface additions for NULL values(Bruce)
Prevent postmaster from being run as root
Psql \h and \? is now readable(Bruce)
Psql allow backslashed, semicolons anywhere on the line(Bruce)
Psql changed command prompt for lines in query or in quotes(Bruce)
Psql char(3) now displays as (bp)char in \d output(Bruce)
```

```
Psql return code now more accurate(Bryan?)
Psql updated help syntax(Bruce)
Re-visit and fix vacuum(Vadim)
Reduce size of regression diffs, remove timezone name difference(Bruce)
Remove compile-time parameters to enable binary distributions(Bryan)
Reverse meaning of HBA masks(Bryan)
Secure Authentication of local users(Bryan)
Speed up vacuum(Vadim)
Vacuum now had VERBOSE option(Bruce)

Source tree changes
-------------
All functions now have prototypes that are compared against the calls
Allow asserts to be disabled easly from Makefile.global(Bruce)
Change oid constants used in code to #define names
Decoupled sparc and solaris defines(Kurt)
Gcc -Wall compiles cleanly with warnings only from unfixable constructs
Major include file reorganization/reduction(Marc)
Make now stops on compile failure(Bryan)
Makefile restructuring(Bryan, Marc)
Merge bsdi_2_1 to bsdi(Bruce)
Monitor program removed
Name change from Postgres95 to PostgreSQL
New config.h file(Marc, Bryan)
PG_VERSION now set to 6.0 and used by postmaster
Portability additions, including Ultrix, DG/UX, AIX, and Solaris
Reduced the number of #define's, centeralized #define's
Remove duplicate OIDS in system tables(Dan)
Remove duplicate system catalog info or report mismatches(Dan)
Removed many os-specific #define's
Restructured object file generation/location(Bryan, Marc)
Restructured port-specific file locations(Bryan, Marc)
Unused/uninialized variables corrected
```

# 15.16. Release v1.09

Sorry, we stopped keeping track of changes from 1.02 to 1.09. Some of the changes listed in 6.0 were actually included in the 1.02.1 to 1.09 releases.

# 15.17. Release v1.02

## 15.17.1. Migration from v1.02 to v1.02.1

Here is a new migration file for 1.02.1. It includes the ’copy’ change and a script to convert old ascii files.

> **Nota:** The following notes are for the benefit of users who want to migrate databases from postgres95 1.01 and 1.02 to postgres95 1.02.1.
>
> If you are starting afresh with postgres95 1.02.1 and do not need to migrate old databases, you do not need to read any further.

In order to upgrade older postgres95 version 1.01 or 1.02 databases to version 1.02.1, the following steps are required:

1. Start up a new 1.02.1 postmaster
2. Add the new built-in functions and operators of 1.02.1 to 1.01 or 1.02 databases. This is done by running the new 1.02.1 server against your own 1.01 or 1.02 database and applying the queries attached at the end of thie file. This can be done easily through psql. If your 1.01 or 1.02 database is named "testdb" and you have cut the commands from the end of this file and saved them in addfunc.sql:

   ```
   % psql testdb -f addfunc.sql
   ```

Those upgrading 1.02 databases will get a warning when executing the last two statements in the file because they are already present in 1.02. This is not a cause for concern.

## 15.17.2. Dump/Reload Procedure

If you are trying to reload a pg_dump or text-mode ’copy tablename to stdout’ generated with a previous version, you will need to run the attached sed script on the ASCII file before loading it into the database. The old format used ’.’ as end-of-data, while ’\.’ is now the end-of-data marker. Also, empty strings are now loaded in as ” rather than NULL. See the copy manual page for full details.

```
sed ’s/^\.$/\\./g’ <in_file >out_file
```

If you are loading an older binary copy or non-stdout copy, there is no end-of-data character, and hence no conversion necessary.

```
- following lines added by agc to reflect the case-insensitive
- regexp searching for varchar (in 1.02), and bpchar (in 1.02.1)
create operator ~* (leftarg = bpchar, rightarg = text, procedure = texticreg
create operator !~* (leftarg = bpchar, rightarg = text, procedure = texticre
create operator ~* (leftarg = varchar, rightarg = text, procedure = texticre
create operator !~* (leftarg = varchar, rightarg = text, procedu-
re = texticregexne);
```

## 15.17.3. Detailed Change List

```
Source code maintenance and development
 * worldwide team of volunteers
```

```
 * the source tree now in CVS at ftp.ki.net

Enhancements
 * psql (and underlying libpq library) now has many more options for
   formatting output, including HTML
 * pg_dump now output the schema and/or the data, with many fixes to
   enhance completeness.
 * psql used in place of monitor in administration shell scripts.
   monitor to be depreciated in next release.
 * date/time functions enhanced
 * NULL insert/update/comparison fixed/enhanced
 * TCL/TK lib and shell fixed to work with both tck7.4/tk4.0 and tcl7.5/tk4.

Bug Fixes (almost too numerous to mention)
 * indexes
 * storage management
 * check for NULL pointer before dereferencing
 * Makefile fixes

New Ports
 * added SolarisX86 port
 * added BSDI 2.1 port
 * added DGUX port
```

# 15.18. Release v1.01

## 15.18.1. Migration from v1.0 to v1.01

The following notes are for the benefit of users who want to migrate databases from postgres95 1.0 to postgres95 1.01.

If you are starting afresh with postgres95 1.01 and do not need to migrate old databases, you do not need to read any further.

In order to postgres95 version 1.01 with databases created with postgres95 version 1.0, the following steps are required:

1. Set the definition of NAMEDATALEN in src/Makefile.global to 16 and OIDNAMELEN to 20.
2. Decide whether you want to use Host based authentication.

    a. If you do, you must create a file name "pg_hba" in your top-level data directory (typically the value of your $PGDATA). src/libpq/pg_hba shows an example syntax.

    b. If you do not want host-based authentication, you can comment out the line

    ```
    HBA = 1
    ```

    in src/Makefile.global

    Note that host-based authentication is turned on by default, and if you do not take steps A or B above, the out-of-the-box 1.01 will not allow you to connect to 1.0 databases.

3. Compile and install 1.01, but DO NOT do the initdb step.
4. Before doing anything else, terminate your 1.0 postmaster, and backup your existing $PGDATA directory.
5. Set your PGDATA environment variable to your 1.0 databases, but set up path up so that 1.01 binaries are being used.
6. Modify the file $PGDATA/PG_VERSION from 5.0 to 5.1
7. Start up a new 1.01 postmaster
8. Add the new built-in functions and operators of 1.01 to 1.0 databases. This is done by running the new 1.01 server against your own 1.0 database and applying the

queries attached and saving in the file 1.0_to_1.01.sql. This can be done easily through psql. If your 1.0 database is name "testdb":

```
% psql testdb -f 1.0_to_1.01.sql
```

and then execute the following commands (cut and paste from here):

```
- add builtin functions that are new to 1.01

create function int4eqoid (int4, oid) returns bool as 'foo'
language 'internal';
create function oideqint4 (oid, int4) returns bool as 'foo'
language 'internal';
create function char2icregexeq (char2, text) returns bool as 'foo'
language 'internal';
create function char2icregexne (char2, text) returns bool as 'foo'
language 'internal';
create function char4icregexeq (char4, text) returns bool as 'foo'
language 'internal';
create function char4icregexne (char4, text) returns bool as 'foo'
language 'internal';
create function char8icregexeq (char8, text) returns bool as 'foo'
language 'internal';
create function char8icregexne (char8, text) returns bool as 'foo'
language 'internal';
create function char16icregexeq (char16, text) returns bool as 'foo'
language 'internal';
create function char16icregexne (char16, text) returns bool as 'foo'
language 'internal';
create function texticregexeq (text, text) returns bool as 'foo'
language 'internal';
create function texticregexne (text, text) returns bool as 'foo'
language 'internal';

- add builtin functions that are new to 1.01

create operator = (leftarg = int4, rightarg = oid, procedure = int4eqoid)
create operator = (leftarg = oid, rightarg = int4, procedure = oideqint4)
create operator ~* (leftarg = char2, rightarg = text, procedure = char2ic
```

```
create operator !~* (leftarg = char2, rightarg = text, procedure = char2i
create operator ~* (leftarg = char4, rightarg = text, procedure = char4ic
create operator !~* (leftarg = char4, rightarg = text, procedure = char4i
create operator ~* (leftarg = char8, rightarg = text, procedure = char8ic
create operator !~* (leftarg = char8, rightarg = text, procedure = char8i
create operator ~* (leftarg = char16, rightarg = text, procedure = char16
create operator !~* (leftarg = char16, rightarg = text, proce-
dure = char16icregexne);
create operator ~* (leftarg = text, rightarg = text, procedure = texticre
create operator !~* (leftarg = text, rightarg = text, procedure = tex-
ticregexne);
```

## 15.18.2. Detailed Change List

```
Incompatibilities:
 * 1.01 is backwards compatible with 1.0 database provided the user
   follow the steps outlined in the MIGRATION_from_1.0_to_1.01 file.
   If those steps are not taken, 1.01 is not compatible with 1.0 database.

Enhancements:
 * added PQdisplayTuples() to libpq and changed monitor and psql to use it
 * added NeXT port (requires SysVIPC implementation)
 * added CAST .. AS ... syntax
 * added ASC and DESC keywords
 * added 'internal' as a possible language for CREATE FUNCTION
   internal functions are C functions which have been statically linked
   into the postgres backend.
 * a new type "name" has been added for system identifiers (table names,
   attribute names, etc.)  This replaces the old char16 type.   The
   of name is set by the NAMEDATALEN #define in src/Makefile.global
 * a readable reference manual that describes the query language.
 * added host-based access control.  A configuration file ($PGDATA/pg_hba)
   is used to hold the configuration data.  If host-based access control
```

```
   is not desired, comment out HBA=1 in src/Makefile.global.
 * changed regex handling to be uniform use of Henry Spencer's re-
gex code
   regardless of platform.  The regex code is included in the distribution
 * added functions and operators for case-insensitive regular expressions.
   The operators are ~* and !~*.
 * pg_dump uses COPY instead of SELECT loop for better performance

Bug fixes:
 * fixed an optimizer bug that was causing core dumps when
   functions calls were used in comparisons in the WHERE clause
 * changed all uses of getuid to geteuid so that effective uids are used
 * psql now returns non-zero status on errors when using -c
 * applied public patches 1-14
```

# 15.19. Release v1.0

## 15.19.1. Detailed Change List

```
Copyright change:
 * The copyright of Postgres 1.0 has been loosened to be freely modifiable
   and modifiable for any purpose.  Please read the COPYRIGHT file.
   Thanks to Professor Michael Stonebraker for making this possible.

Incompatibilities:
 *  date formats have to be MM-DD-YYYY (or DD-MM-YYYY if you're using
   EUROPEAN STYLE).  This follows SQL-92 specs.
 *  "delimiters" is now a keyword

Enhancements:
```

```
 *  sql LIKE syntax has been added
 *  copy command now takes an optional USING DELIMITER specification.
   delimiters can be any single-character string.
 *  IRIX 5.3 port has been added.
   Thanks to Paul Walmsley and others.
 *  updated pg_dump to work with new libpq
 *  \d has been added psql
   Thanks to Keith Parks
 *  regexp performance for architectures that use POSIX regex has been
   improved due to caching of precompiled patterns.
   Thanks to Alistair Crooks
 *  a new version of libpq++
   Thanks to William Wanders

Bug fixes:
 *  arbitrary userids can be specified in the createuser script
 *  \c to connect to other databases in psql now works.
 *  bad pg_proc entry for float4inc() is fixed
 *  users with usecreatedb field set can now create databases without
   having to be usesuper
 *  remove access control entries when the entry no longer has any
   permissions
 *  fixed non-portable datetimes implementation
 *  added kerberos flags to the src/backend/Makefile
 *  libpq now works with kerberos
 *  typographic errors in the user manual have been corrected.
 *  btrees with multiple index never worked, now we tell you they don't
   work when you try to use them
```

# 15.20. Postgres95 Beta 0.03

## 15.20.1. Detailed Change List

```
Incompatible changes:
 * BETA-0.3 IS INCOMPATIBLE WITH DATABASES CREATED WITH PREVIOUS VERSIONS
   (due to system catalog changes and indexing structure changes).
 * double-quote (") is deprecated as a quoting character for string literals
   you need to convert them to single quotes (').
 * name of aggregates (eg. int4sum) are renamed in accordance with the
   SQL standard (eg. sum).
 * CHANGE ACL syntax is replaced by GRANT/REVOKE syntax.
 * float literals (eg. 3.14) are now of type float4 (instead of float8 in
   previous releases); you might have to do typecasting if you de-
pend on it
   being of type float8.  If you neglect to do the typecasting and you assig
   a float literal to a field of type float8, you may get incorrect values
   stored!
 * LIBPQ has been totally revamped so that frontend applications
   can connect to multiple backends
 * the usesysid field in pg_user has been changed from int2 to int4 to
   allow wider range of Unix user ids.
 * the netbsd/freebsd/bsd o/s ports have been consolidated into a
   single BSD44_derived port.  (thanks to Alistair Crooks)

SQL standard-compliance (the following details changes that makes postgres95
more compliant to the SQL-92 standard):
 * the following SQL types are now built-in: smallint, int(eger), float, rea
   char(N), varchar(N), date and time.

   The following are aliases to existing postgres types:
smallint -> int2
integer, int -> int4
float, real  -> float4
   char(N) and varchar(N) are implemented as truncated text types. In
```

```
   addition, char(N) does blank-padding.
 * single-quote (’) is used for quoting string literals; ” (in ad-
dition to
   \’) is supported as means of inserting a single quote in a string
 * SQL standard aggregate names (MAX, MIN, AVG, SUM, COUNT) are used
   (Also, aggregates can now be overloaded, i.e. you can define your
   own MAX aggregate to take in a user-defined type.)
 * CHANGE ACL removed. GRANT/REVOKE syntax added.
   - Privileges can be given to a group using the "GROUP" keyword.
For example:
GRANT SELECT ON foobar TO GROUP my_group;
The keyword ’PUBLIC’ is also supported to mean all users.

Privileges can only be granted or revoked to one user or group
at a time.

"WITH GRANT OPTION" is not supported.  Only class owners can change
access control
   - The default access control is to to grant users readonly access.
     You must explicitly grant insert/update access to users.  To change
     this, modify the line in
src/backend/utils/acl.h
     that defines ACL_WORLD_DEFAULT

Bug fixes:
 * the bug where aggregates of empty tables were not run has been fi-
xed. Now,
   aggregates run on empty tables will return the initial conditions of the
   aggregates. Thus, COUNT of an empty table will now properly re-
turn 0.
   MAX/MIN of an empty table will return a tuple of value NULL.
 * allow the use of \; inside the monitor
 * the LISTEN/NOTIFY asynchronous notification mechanism now work
 * NOTIFY in rule action bodies now work
 * hash indices work, and access methods in general should perform better.
   creation of large btree indices should be much faster.  (thanks to Paul
   Aoki)
```

```
Other changes and enhancements:
 * addition of an EXPLAIN statement used for explaining the query execution
   plan (eg. "EXPLAIN SELECT * FROM EMP" prints out the execution plan for
   the query).
 * WARN and NOTICE messages no longer have timestamps on them. To turn on
   timestamps of error messages, uncomment the line in
   src/backend/utils/elog.h:
/* define ELOG_TIMESTAMPS */
 * On an access control violation, the message
"Either no such class or insufficient privilege"
   will be given.  This is the same message that is returned when
   a class is not found.  This dissuades non-privileged users from
   guessing the existence of privileged classes.
 * some additional system catalog changes have been made that are not
   visible to the user.

libpgtcl changes:
 * The -oid option has been added to the "pg_result" tcl command.
   pg_result -oid returns oid of the last tuple inserted.   If the
   last command was not an INSERT, then pg_result -oid returns "".
 * the large object interface is available as pg_lo* tcl commands:
   pg_lo_open, pg_lo_close, pg_lo_creat, etc.

Portability enhancements and New Ports:
 * flex/lex problems have been cleared up.  Now, you should be able to use
   flex instead of lex on any platforms.  We no longer make assump-
tions of
   what lexer you use based on the platform you use.
 * The Linux-ELF port is now supported.  Various configuration ha-
ve been
   tested:  The following configuration is known to work:
kernel 1.2.10, gcc 2.6.3, libc 4.7.2, flex 2.5.2, bison 1.24
   with everything in ELF format,

New utilities:
 * ipcclean added to the distribution
```

```
   ipcclean usually does not need to be run, but if your backend crashes
   and leaves shared memory segments hanging around, ipcclean will
   clean them up for you.

New documentation:
 * the user manual has been revised and libpq documentation added.
```

# 15.21. Postgres95 Beta 0.02

## 15.21.1. Lista Detallada de Cambios

```
Cambios incompatibles:
 * La declaracion SQL para crear una base de datos en 'CREATE DA-
TABASE' en lugar
   de 'CREATEDB'. De modo similar, el borrado de una base de datos en 'DROP
TABASE' en lugar
   de 'DESTROYDB'. Sin embargo los nombres de los ejecutables 'crea-
tedb' y 'destroydb'
   permanecen igual.


Nuevas herramientas:
 * pgperl - una interfaz Perl (4.036) para Postgres95
 * pg_dump - una utlidad para volcar una base de datos postgres en un ficher
fichero de script conteniedo los comandos de creacion. Los fiche-
ros script estan
en formato ASCII y pueden ser usados para reconstruir una base de da-
tos, incluso
en otras maquinas y otras arquitecturas. (Tambien es bueno para convertir
una base de datos Postgres 4.2 a base de datos Postgres95.)
```

```
Los siguientes portes han sido incorporados en postgres95-beta-0.02:
 * el porte a NetBSD por Alistair Crooks
 * el porte a AIX por Mike Tung
 * el porte a Windows NT por Jon Forrest (hay mas gente pero aun no han he-
cho nada)
 * el porte a Linux ELF por Brian Gallew

Los siguientes errores han sido corregidos en postgres95-beta-0.02:
 * lineas nuevas no acaban en escape en COPY OUT y problemas con COPY OUT cu
   la primera linea es un '.'
 * no se puede escribir un "retur" para utilizar el id del usuario por omi-
sion en "createuser"
 * SELECT DISTINCT en tablas grandes aborta
 * Problemas en la instalacion en Linux
 * monitor no acepta el uso de 'localhost' como PGHOST
 * psql core dumps when doing \c or \l
 * the "pgtclsh" target missing from src/bin/pgtclsh/Makefile
 * libpgtcl has a hard-wired default port number
 * SELECT DISTINCT INTO TABLE se cuelga
 * CREATE TYPE no acepta 'variable' como longitud interna ("internallenght")
 * resultados incorrectos utilizando mas de 1 agregado en una SE-
LECT
```

## 15.22. Postgres95 Beta 0.01

Version inicial.

# 15.23. Tiempos Resultantes

Estos son los tiempos resultantes de ejecutar los test de regresion con los comandos

```
% cd src/test/regress
% make all
% time make runtest
```

Los tiempos bajo Linux 2.0.27 parecen tener una variacion de aproximadamente 5% de ejecucion a ejecicion, presumiblemente debido a vaguedades de planificacion en los sistemas multitareas.

## 15.23.1. v6.5

Como ha sido el caso para versiones precedentes, los tiempos entre versiones no son directamente comparables puesto que se han añadido nuevos test de regresion. En general, la v6.5 es mas rapida que versiones precedentes. releases.

Tiempo con `fsync()` desabilitado:

```
  Tiempo   Sistema
  02:00  Dual Pentium Pro 180, 224MB, UW-SCSI, Linux 2.0.36, gcc 2.7.2.3 -
O2 -m486
  04:38  Sparc Ultra 1 143MHz, 64MB, Solaris 2.6
```

Tiempos con `fsync()` habilitado:

```
  Tiempo Sistema
  04:21  Dual Pentium Pro 180, 224MB, UW-SCSI, Linux 2.0.36, gcc 2.7.2.3 -
O2 -m486
```

Para el sistema Linux anterior, la utilizacion de discos UW-SCSI mejores que los (viejos) IDE conducen a una mejora de un 50% en la velocidad de los test de regresion.

## 15.23.2. v6.4beta

Los tiempos para esta version no son directamente comparables a aquellos de versiones precedentes puesto que algunos test de regresion adicionales han sido incluidos. No obstante, en general, la v6.4 puede ser levemente mas rapida que versiones precedentes (gracias, Bruce!).

```
  Tiempo   Sistema
  02:26  Dual Pentium Pro 180, 96MB, UW-SCSI, Linux 2.0.30, gcc 2.7.2.1 -
O2 -m486
```

## 15.23.3. v6.3

Los tiempos para esta version no son directamente comparables a aquellos de versiones precedentes puesto que algunos test de regresion adicionales han sido incluidos y algunos test obsoletos incluyendo los tiempos de viaje han sido eliminados. No obstante, en general, la v6.3 es sustancialmente mas rapida que versiones precedentes (gracias, Bruce!).

```
  Tiempo   Sistema
  02:30  Dual Pentium Pro 180, 96MB, UW-SCSI, Linux 2.0.30, gcc 2.7.2.1 -
O2 -m486
  04:12  Dual Pentium Pro 180, 96MB, EIDE, Linux 2.0.30, gcc 2.7.2.1 -
O2 -m486
```

## 15.23.4. v6.1

```
  Tiempo   Sistema
  06:12  Pentium Pro 180, 32MB, EIDE, Linux 2.0.30, gcc 2.7.2 -O2 -
m486
  12:06  P-100, 48MB, Linux 2.0.29, gcc
  39:58  Sparc IPC 32MB, Solaris 2.5, gcc 2.7.2.1 -O -g
```

# Bibliografía

Selección de referencias y lecturas sobre SQL y Postgres.

## Libros de referencia sobre SQL

*The Practical SQL Handbook* , Bowman et al, 1993 , *Using Structured Query Language* , 3, Judity Bowman, Sandra Emerson, y Marcy Damovsky, 0-201-44787-8, 1996, Addison-Wesley, 1997.

*A Guide to the SQL Standard* , Date and Darwen, 1997 , *A user's guide to the standard database language SQL* , 4, C. J. Date y Hugh Darwen, 0-201-96426-0, 1997, Addison-Wesley, 1997.

*An Introduction to Database Systems* , Date, 1994 , 6, C. J. Date, 1, 1994, Addison-Wesley, 1994.

*Understanding the New SQL* , Melton and Simon, 1993 , *A complete guide*, Jim Melton y Alan R. Simon, 1-55860-245-3, 1993, Morgan Kaufmann, 1993.

**Abstract**

Accessible reference for SQL features.

*Principles of Database and Knowledge : Base Systems* , Ullman, 1988 , Jeffrey D. Ullman, 1, Computer Science Press , 1988 .

# Documentación específica sobre PostgreSQL

*The PostgreSQL Administrator's Guide* , The Administrator's Guide , Editado por Thomas Lockhart, 1998-10-01, The PostgreSQL Global Development Group.

*The PostgreSQL Developer's Guide* , The Developer's Guide , Editado por Thomas Lockhart, 1998-10-01, The PostgreSQL Global Development Group.

*The PostgreSQL Programmer's Guide* , The Programmer's Guide , Editado por Thomas Lockhart, 1998-10-01, The PostgreSQL Global Development Group.

*The PostgreSQL Tutorial Introduction* , The Tutorial , Editado por Thomas Lockhart, 1998-10-01, The PostgreSQL Global Development Group.

*The PostgreSQL User's Guide* , The User's Guide , Editado por Thomas Lockhart, 1998-10-01, The PostgreSQL Global Development Group.

*Enhancement of the ANSI SQL Implementation of PostgreSQL* , Simkovics, 1998 , Stefan Simkovics, O.Univ.Prof.Dr.. Georg Gottlob, November 29, 1998, Department of Information Systems, Vienna University of Technology .

Discurre sobre la historia y la sintaxis de SQL y describe la adición de las construcciones INTERSECT y EXCEPT en Postgres. Preparado como "Master's Thesis" con ayuda de O.Univ.Prof.Dr. Georg Gottlob y Univ.Ass. Mag. Katrin Seyr en Vienna University of Technology.

*The Postgres95 User Manual* , Yu and Chen, 1995 , A. Yu y J. Chen, The POSTGRES Group , Sept. 5, 1995, University of California, Berkeley CA.

# Procedimientos y Articulos

*Partial indexing in POSTGRES: research project* , Olson, 1993 , Nels Olson, 1993, UCB Engin T7.49.1993 O676, University of California, Berkeley CA.

*A Unified Framework for Version Modeling Using Production Rules in a Database System* , Ong and Goh, 1990 , L. Ong y J. Goh, April, 1990, ERL Technical Memorandum M90/33, University of California, Berkeley CA.

*The Postgres Data Model* , Rowe and Stonebraker, 1987 , L. Rowe y M. Stonebraker, Sept. 1987, VLDB Conference, Brighton, England, 1987.

*Generalized partial indexes (http://simon.cs.cornell.edu/home/praveen/papers/partindex.de95.ps.Z)* , , P. Seshadri y A. Swami, March 1995, Eleventh International Conference on Data Engineering, 1995, Cat. No.95CH35724, IEEE Computer Society Press.

*The Design of Postgres* , Stonebraker and Rowe, 1986 , M. Stonebraker y L. Rowe, May 1986, Conference on Management of Data, Washington DC, ACM-SIGMOD, 1986.

*The Design of the Postgres Rules System*, Stonebraker, Hanson, Hong, 1987 , M. Stonebraker, E. Hanson, y C. H. Hong, Feb. 1987, Conference on Data Engineering, Los Angeles, CA, IEEE, 1987.

*The Postgres Storage System* , Stonebraker, 1987 , M. Stonebraker, Sept. 1987, VLDB Conference, Brighton, England, 1987.

*A Commentary on the Postgres Rules System* , Stonebraker et al, 1989, M. Stonebraker, M. Hearst, y S. Potamianos, Sept. 1989, Record 18(3), SIGMOD, 1989.

*The case for partial indexes (DBMS) (http://s2k-ftp.CS.Berkeley.EDU:8000/postgres/papers/ERL-M89-17.pdf)* , Stonebraker, M, 1989b, M. Stonebraker, Dec. 1989, Record 18(no.4):4-11, SIGMOD, 1989.

*The Implementation of Postgres* , Stonebraker, Rowe, Hirohama, 1990 , M. Stonebraker, L. A. Rowe, y M. Hirohama, March 1990, Transactions on Knowledge and Data Engineering 2(1), IEEE.

*On Rules, Procedures, Caching and Views in Database Systems* , Stonebraker et al, ACM, 1990 , M. Stonebraker y et al, June 1990, Conference on Management of Data, ACM-SIGMOD.