ADMINISTRACIÓN DE BASE DE DATOS CON POSTGRESQL. LABORATORIO 5. RESPALDO DE LA BD Y EXPORTACIÓN DE DATOS ENTRE DIFERENTES DBMS

—

Luis Álvarez Oval • loval@unach.mx
Christian Mauricio Castillo Estrada • cmce@unach.mx
Aron De La Cruz Vázquez • aron.cruz@unach.mx

Facultad De Contaduría Pública C-IV UNACH

Para citar este artículo:
Álvarez, L., Castillo, C. y De la Cruz, A.(2017) Administración de base de datos con postgreSQL. Laboratorio 5. Respaldo de la BD y exportación de datos entre diferentes DBMS. *Espacio I+D Innovación más Desarrollo 6* (13) 145-176. Recuperado de http://www.espacioimasd.unach.mx/suplemento/espacioimasd_espanol_13.pdf

RESUMEN

La serie de laboratorios de Administración de Bases de Datos con PostgreSQL, muestra de forma práctica la administración de este tipo de sistemas, el cual tiene un amplio uso en la industria de desarrollo de software. Mientras que las bases de datos es la herramienta que requieren todas las empresas que necesitan almacenar la información que generan, y es en este tipo de sistemas donde se guarda ésta. De ahí la importancia de entender y aplicar los conceptos de administración estándares que se usan en la industria. Se usa el sistema PostgreSQL debido a que ofrece los mecanismos que tienen otros sistemas similares pero de carácter propietario. PostgreSQL se ofrece bajo una licencia PostgresSQL, lo que permite desde el punto de vista del propietario de un sistema de información evitar el pago de costosas licencias por el uso de una base de datos.

Palabras clave.

Administración de Base de Datos, SQL, Programación de procedimientos almacenados, postgreSQL.

Esta quinta entrega de una serie de seis laboratorios de Administración de Base de Datos (ABD) enseña el respaldo de una base de datos y la exportación de datos entre diferentes sistemas de administración de base de datos relacionales (DBMS). Para este laboratorio es necesario que el lector disponga de dos computadoras, una de ellas ejecutando el sistema operativo Windows y la otra con Linux. Para la transferencia de archivos entre ambos sistemas operativos estaremos usando el protocolo de transferencia de archivos (FTP), por lo que será necesario instalar el servidor FTP en el equipo que usa Linux.

Los laboratorios se han diseñado para proporcionar los conceptos y la experiencia necesarios para conocer detalladamente el sistema, se aprovecha la función de "copiar y pegar" que ofrece el sistema operativo para disminuir el esfuerzo del lector en la preparación del ambiente de trabajo y en la solución de los problemas. En la sección denominada "trabajo adicional" se requiere que el lector aplique la experiencia obtenida en la solución de problemas relacionados con el tema central del laboratorio. La sección de conceptos básicos muestra la sintaxis de los comandos y da algunas explicaciones del uso de los mismos, este material ha sido tomado del Manual de usuario del sistema PostgreSQL el cual está disponible en la página oficial (https://www.postgresql.org/docs/9.3/static/) de la herramienta, en algunos casos se ha tomado del sitio oficial en Español. En esta misma sección se describen algunos comandos y aplicaciones del sistema operativo Linux cuyas descripciones de uso y/o sintaxis se han tomado de los manuales de usuario del sistema y se han complementado con publicaciones del sitio Wikipedia (es.wikipedia.org). Los conceptos básicos se aplican en torno al mismo proyecto que usaremos en esta serie: "Universidad ACME", el cual es producto de la imaginación del autor, así como la solución práctica de los problemas planteados. Los libros que se ofrecen en la sección de referencias, sirven como consulta para apoyar algunos de los conceptos que se aplican en la solución práctica de problemas de administración de base de datos.

Estos laboratorios se han preparado para procurar experiencia práctica a los estudiantes de la materia Administración de Base de Datos de la Licenciatura en Sistemas Computacionales que se ofrece en la Facultad de Contaduría Pública (FCP) del Campus IV de la Universidad Autónoma de Chiapas (UNACH). En la FCP tenemos al menos 14 años de experiencia en el uso de PostgreSQL en las aulas, proyectos de investigación y en sistemas que se han implementado para la automatización de las actividades cotidianas de la FCP. Como producto de esa experiencia académica e industrial se han obtenido estos laboratorios que se usan en las aulas para capacitar a nuestros estudiantes. También se tiene noticia de que son una fuente de consulta para egresados que laboran en el sector empresarial.

Como se ha mencionado previamente la herramienta tiene características y lenguajes de programación estándar que ofrecen sistemas propietarios, por lo que los ejemplos fácilmente pueden ser aplicados en otros sistemas de bases de datos del mercado, o pueden ser referencia para aplicar los conceptos en proyectos industriales. Por lo que puedan servir como consulta a profesionales de las Ciencias de la Computación.

OBJETIVO

El lector aprenderá a usar los procedimientos de respaldo, además de cómo migrar datos entre diferentes Sistemas de Administración de Bases de Datos.

PRERREQUISITOS

Se espera que el lector tenga experiencia previa en el uso y conversión de diagramas Entidad-Relación (E-R), los temas asociados al Diseño de Base de Datos no se cubren en este documento. También se espera que el lector tenga conocimientos de programación en cualquier lenguaje de programación y si necesita información adicional del PLPGSQL, se sugiere que visite el sitio: http://www.postgresql.org/docs/9.3/static/plpgsql.html, o busque esta información en el libro PostgreSQL (2003)[1] de los autores Susan y Korry Douglas.

Asimismo se espera que el lector tenga experiencia en la conexión de redes locales, es necesario que se configuren el sistema de transferencia de archivos (FTP), el cliente en Windows y el servidor en Linux. Finalmente, es necesario instalar la base de datos PostgreSQL versión 9.3 sobre el sistema operativo Windows o Linux, verifique los requerimientos para instalación en la página oficial de la herramienta: www. postgresql. org El sistema puede descargarse del sitio Web:

http://www.enterprisedb.com/products-services-training/pgdownload#windows

Si tiene alguna duda con respecto a PostgreSQL, le recomiendo visitar el sitio oficial con información publicada en idioma español: http://www. postgresql.org.es/primeros_pasos.

Partes de la que se compone este laboratorio:

1. Proyecto a desarrollar
2. Conceptos básicos

1 (ISBN: 0672327562).

3. Preparación del ambiente de trabajo
4. Problemática a resolver
5. Trabajo adicional
6. Referencias

1. PROYECTO A DESARROLLAR

El ejercicio que se va a realizar consiste en un proyecto que describe el problema de una empresa dedicada a la prestación de servicios educativos: después de leer el texto se genera el diagrama E-R con la solución a este problema, se continúa con la creación de las tablas y población de las tablas, para finalmente trabajar con los permisos de grupos y usuarios.

Proyecto Universidad ACME

En UACME, se ofrecen dos tipos de cursos en el periodo especial de verano, en el cual se imparten cursos de verano y cursos extracurriculares. Los primeros son materias que un alumno regular que estudia una carrera cursa en este periodo, se le permite adelantar hasta dos materias; mientras que los segundos son cursos especiales de capacitación que se ofrecen a alumnos regulares como estudiantes o profesionistas externos.

Los docentes de la UACME, son los únicos a los que se les permite impartir estos cursos, por los cuales recibe un pago adicional, se les paga de acuerdo a un tabulador que indica el costo de la hora de estos cursos de acuerdo al nivel académico del docente. El pago se genera a partir del alta del curso y solo se permite expedir un cheque por cada curso. Además los estudiantes deben acudir a pagar adicionalmente al costo del semestre por asistir a ellos.

UACME tiene dos departamentos que intervienen en la administración de los cursos:

A) Departamento de Administración (DA) y B) Departamento de Control Escolar (DCE). Corresponde al DA, efectuar el pago a los docentes y los cobros a los alumnos. El DA es dirigido por el C.P. Ávila y es auxiliado por el Sr. Cancino. Mientras que el DCE, es dirigido por el Lic. Barroso y auxiliado por los Sres. Tirado, Martínez, Aquino y Ramos y es en este donde se decide cuales cursos se imparten en el periodo, quién los imparte, y se aceptan las solicitudes de los alumnos. Un caso especial, es el de los Profesores, ya que el DA es quién les puede modificar el sueldo quincenal, mientras que el DCE ni siquiera puede visualizar éste. Lo curioso radica en que, es el DCE quién acepta los docentes y los registra en el sistema, pero es el DA donde se captura el sueldo. Importante es para la administración de la UACME que esta política se aplique al pie de la letra, y que sea implementado directamente sobre la DB.

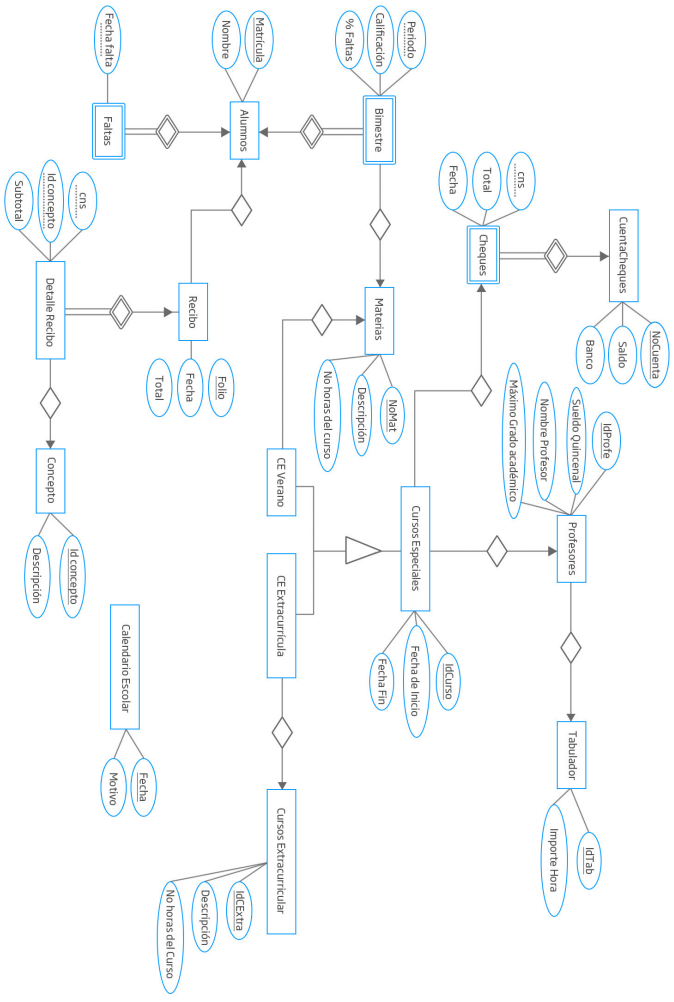
A continuación se describe detalladamente las tablas a las cuales tiene acceso el personal de cada

Tablas a las que se le permite el acceso al personal de la Secretaría Administrativa: CuentaCheques, Cheque, Tabulador, Profesores, Concepto, Recibo, y DetalleRecibo.

Como casos especiales este departamento podrá acceder a consultar las tablas de Cursos Especiales, Cursos Especiales Verano, Cursos Especiales Extracurriculares, Cursos Extracurriculares y Materias. Explícitamente no se les permite modificar ningún campo o registro.

Tablas a las que se le permite el acceso al personal de la Secretaría Escolar: CursosEspeciales, CursosExtracurricular, Materias, CEVerano, CEExtracurricula, Alumnos, Bimestre, Faltas, CalendarioEscolar.

Figura 1. Diagrama E/R que resuelve el problema anterior.



2. CONCEPTOS BÁSICOS.

COPY— Copia datos entre archivos y tablas

Sintaxis

```
COPY [ BINARY ] table [ WITH OIDS ]
FROM { 'filename' | stdin }
[ [USING] DELIMITERS 'delimiter' ]
[ WITH NULL AS 'null string' ]
COPY [ BINARY ] table [ WITH OIDS ]
TO { 'filename' | stdout }
[ [USING] DELIMITERS 'delimiter' ]
[ WITH NULL AS 'null string' ]
```

Entradas

- BINARY - Cambia el comportamiento del formato de campos, forzando a todos los datos a almacenarse o leerse como objetos binarios, en lugar de como texto.
- *Table* - El nombre de una tabla existente.
- WITH OIDS - Copia el identificador de objeto interno único (OID) para cada fila.
- *Filename* - La ruta absoluta en formato Unix del archivo de entrada o salida.
- *Stdin* - Específica que la entrada viene de un conducto o terminal.
- *Stdout* - Específica que la salida va a un conducto o terminal.
- *Delimiter* - Un carácter que delimita los campos de entrada o salida.
- *null print* - Una cadena para representar valores NULL. El valor por defecto es "\N" (backslash-N), por razones históricas. Puede preferir, por ejemplo, una cadena vacía.

Nota: En una copia de entrada, cualquier dato que coincida con esta cadena será almacenado como un valor NULL, por lo que debería asegurarse de usar la misma cadena que usó para la copia de salida.

Salidas

- COPY - La copia se completó satisfactoriamente.
- ERROR: reason - La copia falló por la razón indicada en el mensaje de error.

Descripción

COPY mueve datos entre tablas de Postgres y archivos del sistema de archivos estándar. **COPY** indica al servidor Postgres que lea o escriba de o a un archivo. El archivo ha de ser directamente visible para el servidor, y el nombre completo ha de especificarse desde el punto de vista del servidor. Si se especifica stdin o stdout, los datos van de la aplicación cliente al servidor (o viceversa).

Notas

La palabra clave BINARY obliga a que todos los datos se almacenen o lean como objetos binarios en lugar de como texto. Esto es algo más rápido que el comportamiento normal de COPY pero el resultado no es generalmente portable, y los archivos generados son algo más grandes aunque este es un factor que depende de los datos en sí. Por defecto, cuando se copia un texto se usa un tabulador ("\t") como delimitador. El delimitador puede cambiarse por cualquier otro carácter empleando la palabra clave USING DELIMITERS. Los caracteres dentro de los campos de datos que resulten coincidir con el delimitador serán encerrados entre comillas. Ha de hacerse primero un *select access* en cualquier tabla cuyos valores sean leídos por **COPY**, e *insert or update access* en la tabla en la que se vayan a insertar los valores. El servidor necesita los permisos Unix adecuados sobre cualquier archivo que vaya a leerse o escribirse con este comando. La palabra clave USING DELIMITERS especifica un carácter que se usará para delimitar entre columnas. Si se especifican varios caracteres en la cadena delimitadora, solo se usará el primer caracter.

Sugerencia: No confunda **COPY** con la instrucción **copy** de psql.

- **COPY** no invoca regla ni acciones por defecto en las columnas. Sin embargo, puede invocar procedimientos disparados.

- **COPY** detiene las operaciones en el primer error. Esto no produce problemas en el caso de

- **COPY FROM**, pero el destino, por supuesto, será parcialmente modificado en el caso de un **COPY TO**.

- **VACUUM** puede usarse para limpiar tras una copia fallida.

Debido a que el directorio de trabajo del servidor de Postgres no es normalmente el mismo que el directorio de trabajo del usuario, el resultado de copiar el archivo "foo" (sin añadir información de la ruta) puede dar lugar a resultados inesperados para el usuario inadvertido. En este caso, en lugar de foo, acabamos con $PGDATA/foo. Por lo general, debería usarse la ruta completa tal como se vería desde el servidor, al especificar los archivos a copiar.

Los archivos usados como argumentos para **COPY** deben residir o ser accesible por parte de la máquina servidor de base de datos, en los discos locales o en un sistema de archivos de red. Cuando se emplea una conexión TCP/IP, y se especifica un archivo objetivo, dicho archivo se escribirá en la máquina donde se esté ejecutando el servidor, no en la máquina del usuario.

FORMATOS DE ARCHIVO

Formato de Texto

Cuando se usa **COPY TO** sin la opción BINARY, el archivo generado tendrá cada fila (instancia) en una sola línea, con cada una de las columnas (atributo) separada por el carácter delimitador. Los caracteres delimitadores internos (los caracteres internos que coincidan con el delimitador) se precederán del carácter barra atrás ("\"). Los valores de atributo son cadenas de texto generados por la función de salida asociada con cada uno de los tipos de atributo. La función de salida para un tipo no debería tratar de generar el carácter barra atrás; éste será generado por el comando **COPY**.

El formato para cada instancia es:

```
<attr1><separator><attr2><separator>...<separator><attrn><newline>
```

El identificador se sitúa en el principio de la línea, cuando se especifica WITH OID, si **COPY** envía su salida a la salida estándar en lugar de a un archivo, enviará una barra invertida ("\") y un punto, seguidos de un carácter de salto de línea en una línea separada, cuando termina su salida. Similarmente, si **COPY** está leyendo de una salida estándar, esperará una barra invertida y un punto; seguidos por un fin de línea, como los tres primeros caracteres de una línea para indicar el fin del archivo. Sin embargo, **COPY** terminará (y a continuación terminará la aplicación servidor) si se encuentra un EOF antes de que se encuentre esta cadena que indica el fin de archivo. El carácter barra invertida tiene otros significados especiales. Un carácter barra invertida literal se representa como dos barras consecutivas ("\\". El carácter tabulador se representa con una barra invertida y un tabulador. El carácter fin de línea se representa como una barra invertida y un fin de línea. Cuando se cargan datos de test no generados por PostgreSQL

necesitará convertir el carácter barra invertida en un par de barras para asegurar que se carguen adecuadamente. (La secuencia "\N" siempre se interpretará como una barra invertida y un carácter "N", por compatibilidad. La solución más general es "\\N".)

Binary Format

En el caso de **COPY BINARY**, los primeros cuatro bytes del archivo será el número de instancias en el archivo. Si el número es cero, el comando **COPY BINARY** leerá hasta que se encuentre el fin del archivo. En otro caso, dejará de leer cuando se lean ese número de instancias. Los restantes datos en el archivo se ignorarán.

CONTENIDOS DE UN ARCHIVO BINARIO DE COPY

Alineación de datos binarios

Sobre equipos Sun-3s, los atributos de 2 bytes se alinean en grupos de cuatro bytes. Los atributos de caracteres se alinean en grupos de un solo byte. En la mayoría de las otras máquinas, todos los atributos mayores de un byte se alinean en grupos de cuatro bytes. Nótese que los atributos de longitud variable vienen precedidos de la longitud del atributo; las matrices son simplemente cadenas continuas del elemento tipo de la matriz.

Uso

El siguiente ejemplo copia una tabla a la salida estándar, usando una barra vertical como delimitador de campo:

```
COPY country TO stdout USING DELIMITERS '|';
```

Para copiar datos de un archivo Unix a la tabla "country":

```
COPY country FROM '/usr1/proj/bray/sql/country_data';
```

Aquí un ejemplo de datos adecuados para ser copiados a una tabla desde stdin (dado que tienen la secuencia de terminación en la última línea):

```
AF AFGHANISTAN
AL ALBANIA
DZ ALGERIA
...
```

```
ZM ZAMBIA
ZW ZIMBABWE
\.
```

Compatibilidad

SQL92

No existe la sentencia **COPY** en SQL 92.

pg_dump — Extrae una base de datos Postgres a un archivo de script

Sintáxis

```
pg_dump [ base_de_datos ]
pg_dump [ -h huésped ] [ -p puerto ]
[ -t tabla ]
[ -a ] [ -c ] [ -d ] [ -D ] [ -n ] [ -N ]
[ -o ] [ -s ] [ -u ] [ -v ] [ -x ]
[ base_de_datos ]
```

Entrada

pg_dump acepta los siguientes argumentos de la línea de comando:

Argumento	**Descripción**
base_de_datos	Especifica el nombre de la base de datos que se va a extraer. *base_de_datos* tiene como estándar el valor de la variable de entorno USER.
-a	Vuelca sólo los datos, no el esquema (las definiciones).
-c	Limpia el esquema antes de crearlo.
-d	Vuelca los datos como propios insertos de cadenas.
-D	Vuelca la data como insertos con nombres de atributos.
-n	Suprime las dobles comillas de los identificadores, a menos que sean absolutamente necesarias. Esto puede causar problemas al cargar la misma si esta data volcada contiene palabras reservadas usadas por los identificadores. Esta era la conducta estándar en pg_dump pre-v6.4.
-N	Incluye comillas dobles en los identificadores.
-o	Vuelca los identificadores de objetos (oid) para cada tabla.
-s	Vuelca solo el esquema (las definiciones), no los datos.
-t tabla	Vuelca los datos para la *tabla* únicamente.
-u	Usa autenticación por medio de clave de acceso. Pide un nombre de usuario y clave de acceso.
-v	Epecifica el modo verbose (parlanchín).

-x	Evita el volcado de ACLs (comandos grant/revoke) y la información de propiedad de la tabla.
-h huésped	Especifica el nombre del huésped de la máquina en la cual se está ejecutando el postmaster. El estándar es usar un socket de dominio local Unix en vez de una conexión IP.
-p puerto	Especifica el puerto de Internet TCP/IP o extensión de archivo socket de dominio local Unix en el cual postmaster está esperando que se efectúen conexiones. En número estándar de puerto es 5432, o el valor de la variable de ambiente PGPORT (si está establecida).
-u	Usa autenticación con clave de acceso. Pide *nombre_de_usuario* y *clave_de_acceso*.

Salida

pg_dump creará un archivo o escribirá a stdout.

La conexión con la base de datos 'template1' falló. ConnectDB() falló: ¿Está el postmaster ejecutándose y aceptando conexiones en el 'Socket de UNIX' en el puerto 'puerto'?

pg_dump no pudo conectarse al proceso postmaster en el huésped y puerto especificados. Si ve usted este mensaje, verifique que postmaster se esté ejecutando en el huésped indicado, y que usted especificó el puerto correcto. Si su site usa algún sistema de autenticación, verifique que usted tiene las credenciales de autenticación requeridas.

La conexión con la base de datos 'base_de_datos' falló. FATAL 1: SetUserId: el usuario 'nombre_de_usuario' no está en 'pg_shadow'. Usted no posee una entrada válida en la relación pg_shadow y no le será permitido tener acceso a Postgres. Contacte a su administrador de Postgres.

dumpSequence(tabla): SELECT falló Usted carece del permiso para leer la base de datos. Contacte a su administrador de site Postgres.

Nota: pg_dump ejecuta internamente las directivas **SELECT**. Si tiene problemas ejecutando pg_dump, verifique que puede seleccionar la información de la base de datos mediante el uso de, por ejemplo, psql.

Descripción

pg_dump es un utilitario para volcar una base de datos Postgres en un archivo de script conteniendo comandos de consulta. Los archivos de script son en formato de texto y pueden ser usados para reconstruir la base de datos, incluso en otras máquinas y con otras arquitecturas. pg_dump producirá las consultas necesarias para regenerar todos los tipos definidos por el usuario, funciones, tablas, índices, agregados, y operadores. Adicionalmente, todos

los datos son copiados en formato de texto el cual puede ser nuevamente copiado, también puede ser importado a herramientas para su edición.

pg_dump es útil para verter el contenido de una base de datos que se vaya a mudar de una instalación de Postgres a otra. Después de ejecutar pg_dump, se debe examinar el script de salida a ver si contiene alguna advertencia, especialmente a la luz de las limitaciones citadas en la parte inferior.

Notas

pg_dump tiene pocas limitaciones. Las limitaciones surgen principalmente de la dificultad para extraer ciertas meta-informaciones de los catálogos del sistema.

- pg_dump no entiende los índices parciales. La razón es la misma citada anteriormente; los predicados de los índices parciales se almacenan como planos.
- pg_dump no maneja objetos grandes. Los objetos grandes son ignorados y se debe lidiar con ellos de forma manual.

Uso

Para volcar una base de datos del mismo nombre que el usuario:

```
% pg_dump > db.out
```

Para volver a cargar esta base de datos:

```
% psql -e base_de_datos < db.out
```

pg_dumpall —Extrae todas las bases de datos Postgres en un archivo de script.

Sintáxis

```
pg_dumpall [connection-option...] [option...]
```

Entradas

pg_dumpall acepta los siguientes argumentos de la línea de órdenes:

Argumento	Descripción
-a	Dump only the data, not the schema (data definitions).
-c	Incluye comandos para limpiar (drop) bases de datos antes de recrearlas. Los comandos DROP para roles y espacios de tablas son agregados también.
-f filename	Enviar la salida a un archivo especificado. Si este se omite, la salida estándar es usada.
-g	Vaciar solo objetos globales (roles y espacios de tablas), no las bases de datos.
-i	Una opción obsoleta que ahora es ignorada.
-o	Vacía los identificadores de objetos como parte de los datos de cada tabla. Use esta opción si su aplicación referencia las columnas OID de alguna manera (por ejemplo en una restricción de llave foránea). De otra manera, esta opción no debe ser usada.
-O	No emite comandos para establecer la propiedad de los objetos para que coincidan con la base de datos original. Por omisión, pg_dumpall ejecuta las instrucciones ALTER OWNER o SET SESSION AUTHORIZATION para establecer la propiedad de los elementos de esquema creados. Estas instrucciones fallarán cuando se ejecute el script, a menos que sea iniciado por un superusuario (o el mismo usuario que posee todos los objetos del script). Para hacer un script que pueda ser restaurado por cualquier usuario, pero que le dará a ese usuario la propiedad de todos los objetos, especifique -O.
-r	Vacía solo roles, no las bases de datos o los espacios de tablas.
-s	Vacía solo las definiciones de objetos (esquemas), no los datos.
-S	Especifica el nombre del superusuario a usar cuando se deshabiliten los disparadores. Esto es relevante solo si –disable-triggers es usado. (Normalmente, es mejor dejar esto fuera, y en su lugar iniciar el script resultante como superusuario.)
-t	Vacía solo espacios de tablas, no bases de datos ni roles.
-v	Especifica el modo verbose. Esto causará que pg_dumpall muestre las horas de inicio / parada en el archivo de vaciado, y los mensajes de progreso a error estándar. También habilitará la salida detallada en pg_dump.
-V	Imprime la versión de pg_dumpall y sale.
-x	Evita el vaciado de los privilegios de acceso (comandos grant/revoke).
--binary-upgrade	Esta opción es para uso de utilidades de actualización in situ. Su uso para otros fines no es recomendable ni está soportado. El comportamiento de la opción puede cambiar en versiones futuras sin previo aviso.
--column-inserts --attribute-inserts	Vaciar datos como comandos INSERT con nombres de columna explícitos (INSERT INTO tabla (columna, ...) VALUES ...). Esto hará que la restauración sea muy lenta; Es principalmente útil para hacer vaciados que se pueden cargar en bases de datos no PostgreSQL.
--disable-dollar-quoting	Esta opción desactiva el uso de expresiones para los cuerpos de función y obliga a que se expresen utilizando SQL sintaxis de cadena estándar.
--disable-triggers	Esta opción sólo es relevante cuando se crea un vaciado sólo de datos. Indica a pg_dumpall que incluya comandos para deshabilitar temporalmente los disparadores en las tablas de destino cuando los datos se vuelven a cargar. Use esto si tiene comprobaciones de integridad referencial u otros disparadores en las tablas que no desea invocar durante la recarga de datos. Actualmente los comandos emitidos para --disable-triggers deben hacerse como superusuario. Por lo tanto, también debe especificar un nombre de superusuario con -S, o preferiblemente tenga cuidado de iniciar el script resultante como superusuario.

--inserts	Vaciar datos como comandos INSERT (en lugar de COPY). Esto hará que la restauración sea muy lenta; Es principalmente útil para hacer vaciados que se pueden cargar en bases de datos no PostgreSQL. Tenga en cuenta que la restauración puede fallar completamente si ha alterado el orden de las columnas. La opción --column-inserts es más segura, aunque aún más lenta.
--lock-wait-timeout= timeout	No espera para siempre para adquirir los cierres de tablas compartidas al inicio del vaciado. En su lugar, falla si no puede bloquear una tabla dentro del tiempo de espera especificado. El tiempo de espera se puede especificar en cualquiera de los formatos aceptados por SET statement_timeout. Los valores permitidos varían dependiendo de la versión del servidor de la que se está vaciando, pero un número entero de milisegundos es aceptado por todas las versiones desde 7.3. Esta opción se ignora cuándo se descarga desde un servidor anterior al 7.3.
--no-security-labels	No vaciar las etiquetas de seguridad.
--no-tablespaces	No produce comandos para crear espacios de tabla ni selecciona espacios de tabla para objetos. Con esta opción, todos los objetos se crearán en el espacio de tabla que sea el predeterminado durante la restauración.
--no-logged-table-data	No vuelca el contenido de tablas no registradas. Esta opción no tiene ningún efecto sobre si o no las definiciones de tabla (esquema) son volcadas; sólo suprime el volcado de los datos de la tabla.
--quote-all-identifiers	Fuerza el expresar todos los identificadores. Esta opción se recomienda cuando se descarga una base de datos de un servidor cuya versión principal de PostgreSQL es diferente de la de pg_dumpall o cuando la salida está destinada a ser cargada en un servidor de una versión principal diferente. De forma predeterminada, pg_dumpall expresa sólo los identificadores que son palabras reservadas en su propia versión principal. Esto a veces resulta en problemas de compatibilidad al tratar con servidores de versiones que pueden tener conjuntos ligeramente diferentes de palabras reservadas. El uso de --quote-all-identifiers evita estos problemas, al precio de un script de volcado más difícil de leer.
--use-set-session-authorization	Ejecuta los comandos SET SESSION AUTHORIZATION que son estándar de SQL en lugar de los comandos ALTER OWNER para determinar la propiedad del objeto. Esto hace que el volcado sea más compatible con los estándares, pero dependiendo del historial de los objetos en el volcado, puede no restaurarse correctamente.
-? --help	Muestra la ayuda de los argumentos de línea de comandos de pg_dumpall, y sale de ejecución

Las siguientes opciones de línea de comandos controlan los parámetros de conexión a la base de datos.

Argumento	Descripción
-d connstr --dbname=connstr	Especifica los parámetros utilizados para conectarse al servidor, como una cadena de conexión. Consulte la Sección 31.1.1 para obtener más información. La opción es llamada --dbname por consistencia con otras aplicaciones cliente, pero debido a que pg_dumpall necesita conectarse a muchas bases de datos, el nombre de la base de datos en la cadena de conexión será ignorado. Utilice la opción -l para especificar el nombre de la base de datos utilizada para volcar objetos globales y para descubrir qué otras bases de datos deben volcarse.
-h host --host=host	Especifica el nombre del anfitrión del equipo en el que se ejecuta el servidor de base de datos. Si el valor comienza con una barra, se utiliza como directorio para el socket de dominio Unix. El valor predeterminado se toma de la variable de entorno PGHOST, si se establece, de lo contrario se intenta una conexión de socket de dominio Unix.

-l dbname --database=dbname	Especifica el nombre de la base de datos a conectarse para volcar objetos globales y descubrir qué otras bases de datos se deben volcar. Si no se especifica, se utilizará la base de datos postgres, y si no existe, se utilizará template1.
-p port --port=port	Especifica el puerto TCP o la extensión de archivo de socket de dominio Unix local en la que el servidor está escuchando conexiones. Por omisión a la variable de entorno PGPORT, si está establecida, o a un valor predeterminado compilado.
-U username --username=username	El nombre del usuario con el que se conecta.
-w --no-password	Nunca ejecute una solicitud de contraseña. Si el servidor requiere autenticación de contraseña y una contraseña no está disponible por otros medios, tal como un archivo .pgpass, el intento de conexión fallará. Esta opción puede ser útil en trabajos por lotes y scripts en los que ningún usuario está presente para ingresar una contraseña.
-W --password	Forza a pg_dumpall a pedir una contraseña antes de conectarse a una base de datos. Esta opción nunca es esencial, ya que pg_dumpall solicitará automáticamente una contraseña si el servidor requiere autenticación de contraseña. Sin embargo, pg_dumpall desperdiciará un intento de conexión descubriendo que el servidor desea una contraseña. En algunos casos vale la pena escribir -W para evitar el intento de conexión adicional. Tenga en cuenta que la solicitud de contraseña volverá a aparecer para cada base de datos que se va a vaciar. Por lo general, es mejor configurar un archivo ~ / .pgpass que confiar en la introducción manual de contraseñas.
--role=rolename	Especifica un nombre de rol que se utilizará para crear el volcado. Esta opción hace que pg_dumpall ejecute un comando SET ROLE rolename después de conectarse a la base de datos. Es útil cuando el usuario autenticado (especificado por -U) carece de los privilegios necesarios para pg_dumpall, pero puede cambiar a un rol con los derechos necesarios. Algunas instalaciones tienen una política contra iniciar sesión directamente como un superusuario y el uso de esta opción permite que se efectúen descargas sin violar la política.

Salida

pg_dumpall creará un archivo o escribirá a stdout.

La conexión a la base de datos 'template1' falló. ConnectDB() falló: ¿Está postmaster ejecutándose y aceptando conexiones en el 'Socket UNIX' en el puerto 'puerto'? pg_dumpall no pudo conectarse al proceso postmaster en la máquina y puerto especificados. Si ve usted este mensaje, verifique que postmaster esté ejecutándose correctamente en el huésped y puerto que usted especificó. Si su lugar de trabajo usa algún sistema de autenticación verifique que usted ha obtenido las credenciales de autenticación.

La conexión a la base de datos 'base_de_datos' falló. FATAL 1: SetUserId: el usuario 'nombre_de_usuario' no está en 'pg_shadow' Usted no tiene una entrada válida en la relación pg_shadow y no le será permitido el acceso a Postgres. Contacte con su administrador Postgres.

dumpSequence(tabla): SELECT falló

No tiene permiso para leer la base de datos. Contacte a su administrador Postgres.

Nota: pg_dumpall ejecuta internamente directivas **SELECT**. Si tiene problemas ejecutando pg_dumpall, asegúrese de que puede consultar información de la base de datos usando, por ejemplo, psql.

Descripción

pg_dumpall se diseñó para volcar todas las bases de datos Postgres en un archivo. También vuelca la tabla pg_shadow, la cual es global para todas las bases de datos. pg_dumpall incluye en este archivo las órdenes correctas para crear automáticamente cada una de las bases de datos volcadas antes de cargar los datos. pg_dumpall toma todas las opciones de pg_dump pero -f, -t y *base_de_datos* deberían ser omitidos. Refiérase a pg_dump para más información con respecto a esta otra utilidad.

Uso

Para volcar todas las bases de datos:

```
% pg_dumpall > db.out
```

Sugerencia: Puede usar la mayoría de las opciones de pg_dump con pg_dumpall.

Para volver a cargar esta base de datos:

```
% psql -e template1 < db.out
```

Sugerencia: Puede usar la mayoría de las opciones de psql cuando vuelva a cargarlas.

Cron

En el sistema operativo Unix, cron es un administrador regular de procesos en segundo plano (demonio) que ejecuta procesos o scripts a intervalos regulares (por ejemplo, cada minuto, día, semana o mes). Los procesos que deben ejecutarse y la hora en la que deben hacerlo se especifican en el archivo crontab. Cron se podría definir como el "*equivalente*" a Tareas *Programadas de Windows*. Los usuarios habilitados para crear su archivo crontab se especifican en el archivo cron.allow. De manera análoga, los que no lo tienen permitido figuran en /etc/cron.d/cron.deny, o /etc/cron.deny,

dependiendo de la versión de Unix. Esta sección puede consultarse en el sitio: https://es.wikipedia.org/wiki/Cron_(Unix)

FORMATO DEL ARCHIVO CRONTAB

Archivo crontab de ejemplo:

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
# run-parts
01 * * * * root nice -n 19 run-parts /etc/cron.hourly
50 0 * * * root nice -n 19 run-parts /etc/cron.daily
22 4 * * 0 root nice -n 19 run-parts /etc/cron.weekly
42 4 1 * * root nice -n 19 run-parts /etc/cron.monthly
```

Para agregar, quitar o modificar tareas, hay que editar el crontab. Esto se hace con la orden *crontab -e*, que abrirá el editor definido en la variable de entorno **EDITOR** y cargará el archivo crontab correspondiente al usuario que está logueado. Cada vez que se ejecuta el **crontab**, se envía un mensaje al usuario que aparece en la variable de entorno **MAILTO**, si está habilitado, indicándole la tarea realizada.

Sintaxis

El formato de configuración de cron es muy sencillo.

- El símbolo Numeral "#" es un comentario, todo lo que se encuentre después de ese carácter no será ejecutado por cron.
- El momento de ejecución se especifica de acuerdo con la siguiente tabla:

    1. Minutos: (0-59)
    2. Horas: (0-23)
    3. Días: (1-31)
    4. Mes: (1-12)
    5. Día de la semana: (0-6), siendo 1=Lunes, 2=Martes,... 6=sábado y 0=Domingo

Para especificar todos los valores posibles de una variable se utiliza un asterisco (*).

- La última columna corresponde al path absoluto del binario o script que se quiere ejecutar.

Ejemplos

Por ejemplo:

```
30 10 * * 1 /usr/bin/who >> /home/quien.tex
```

Ejecuta la orden who todos los lunes a las 10:30 y guarda la salida en el archivo «***quien.tex***». Para especificar dos o más valores en cada variable, estas deben estar separadas por comas, siguiendo con el ejemplo anterior:

```
0,30 * * * 1 /usr/bin/who >> /home/quien.tex
```

Ejecuta la orden who todos los lunes cada media hora y guarda la salida en el archivo «***quien.tex***».

Si queremos que se ejecute cada 15 minutos sería:

```
0,15,30,45 * * * * /usr/bin/who >> /home/quien.tex
```

o

```
*/15 * * * * /usr/bin/who >> /home/quien.tex
```

En este ejemplo veremos cómo pasarle más de un comando al cron y de paso como puede programarse una descarga:

```
30 21 * * * cd /media/sda7/isos;wget http://mipagina.com/archivo a
descarga.miarchivo
```

Este otro es para programar el apagado del equipo. En este caso todos los sábados a las 21.30

```
30 21 * * 6 /sbin/shutdown -h now
```

A continuación se listan protocolos que se encuentran incrustados en el sistema operativo Linux y que sirven para dar solución a la problemática presentada en este documento, la información vertida se ha tomado del sitio: es.wikipedia.org/wiki

PROTOCOLO DE TRANSFERENCIA DE ARCHIVOS (FTP)

FTP (siglas en inglés de File Transfer Protocol - Protocolo Transferencias Archivos) en informática, es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP, basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo. El Servicio FTP es ofrecido por la capa de Aplicación del modelo de capas de red TCP/IP al usuario, utilizando normalmente el puerto de red 20 y el 21. Un problema básico de FTP es que está pensado para ofrecer la máxima velocidad en la conexión, pero no la máxima seguridad, ya que todo el intercambio de información, desde el *login* y *password* del usuario en el servidor hasta la transferencia de cualquier archivo, se realiza en texto plano sin ningún tipo de cifrado, con lo que un posible atacante puede capturar este tráfico, acceder al servidor, o apropiarse de los archivos transferidos. Para solucionar este problema son de gran utilidad aplicaciones como scp y sftp, incluidas en el paquete SSH, que permiten transferir archivos pero cifrando todo el tráfico.

EL MODELO FTP

El intérprete de protocolo (PI) de usuario, inicia la conexión de control en el puerto 21. Las órdenes FTP estándar las genera el PI de usuario y se transmiten al proceso servidor a través de la conexión de control. Las respuestas estándar se envían desde el PI del servidor al PI de usuario por la conexión de control como respuesta a las órdenes. Estas órdenes FTP especifican parámetros para la conexión de datos (puerto de datos, modo de transferencia, tipo de representación y estructura) y la naturaleza de la operación sobre el sistema de archivos (almacenar, recuperar, añadir, borrar, etc.). El proceso de transferencia de datos (DTP) de usuario u otro proceso en su lugar, debe esperar a que el servidor inicie la conexión al puerto de datos especificado (puerto 20 en modo activo o estándar) y transferir los datos en función de los parámetros que se hayan especificado.

La comunicación entre cliente y servidor es independiente del sistema de archivos utilizado en cada ordenador, de manera que no importa que sus sistemas operativos sean distintos, porque las entidades que se comunican entre sí son los PI y los DTP, que usan el mismo protocolo estandarizado: el FTP.

También hay que destacar que la conexión de datos es bidireccional, es decir, se puede usar simultáneamente para enviar y para recibir, y no tiene por qué existir todo el tiempo que dura la conexión FTP.

SERVIDOR FTP

Un servidor FTP es un programa especial que se ejecuta en un equipo servidor normalmente conectado a Internet (aunque puede estar conectado a otros tipos de redes, LAN, MAN, etc.). Su función es permitir el intercambio de datos entre diferentes servidores/ordenadores. Por lo general, los programas servidores FTP no suelen encontrarse en los ordenadores personales, por lo que un usuario normalmente utilizará el FTP para conectarse remotamente a uno y así intercambiar información con él.

CLIENTE FTP

Cuando un navegador no está equipado con la función FTP, o si se quiere cargar archivos en un ordenador remoto, se necesitará utilizar un programa cliente FTP. Un cliente FTP es un programa que se instala en el ordenador del usuario, y que emplea el protocolo FTP para conectarse a un servidor FTP y transferir archivos, ya sea para descargarlos o para subirlos. Para utilizar un cliente FTP, se necesita conocer el nombre del archivo, el ordenador en que reside (servidor, en el caso de descarga de archivos), el ordenador al que se quiere transferir el archivo (en caso de querer subirlo nosotros al servidor), y la carpeta en la que se encuentra.

Algunos clientes de FTP básicos en modo consola vienen integrados en los sistemas operativos, incluyendo Windows, DOS, Linux y Unix. Sin embargo, hay disponibles clientes con opciones añadidas e interfaz gráfica. Aunque muchos navegadores tienen ya integrado FTP, es más confiable a la hora de conectarse con servidores FTP no anónimos utilizar un programa cliente.

GUÍA DE COMANDOS FTP

Cuando un cliente intenta comunicarse con el servidor FTP, se puede especificar la dirección de este último en la línea de comandos; pero si no se hace el cliente entra en modo de intérprete de comandos y espera instrucciones del usuario, a continuación se ofrece un resumen de los comandos que le pueden ser útiles para el trabajo de este laboratorio y son parte del Manual de usuario que Linux Ubuntu ofrece (http://manpages.ubuntu.com/manpages/trusty/man1/tnftp.1.html):

COMANDO Y ARGUMENTOS	ACCIÓN QUE REALIZA
open *servidor*	Inicia una conexión con un servidor ftp
close o **disconnect**	Finaliza una conexión ftp sin cerrar el programa cliente
bye o **quit**	Finaliza una conexión ftp y la sesión de trabajo con el programa cliente

cd *directorio*	Cambia el directorio de trabajo en el servidor
delete *archivo*	Borra un archivo en el servidor
mdelete *patrón*	Borra múltiples archivos basado en un patrón que se aplica al nombre.
dir	Muestra el contenido del directorio en el que estamos en el servidor.
get archivo	Obtiene un archivo.
mget *archivos*	Obtiene múltiples archivos.
hash	Activa la impresión de caracteres # a medida que se transfieren archivos, a modo de barra de progreso.
lcd *directorio*	Cambia el directorio de trabajo local.
ls	Muestra el contenido del directorio en el servidor.
prompt	Activa/desactiva la confirmación por parte del usuario de la ejecución de comandos. Por ejemplo al borrar múltiples archivos.
put *archivos*	Envía un archivo al directorio activo del servidor.
mput *archivos*	Envía múltiples archivos.
pwd	Muestra el directorio activo en el servidor.
rename *archivos*	Cambia el nombre a un archivo en el servidor.
rmdir *directorio*	Elimina un directorio en el servidor si ese directorio está vacío.
status	Muestra el estado actual de la conexión.
bin o **binary**	Activa el modo de transferencia binario.
ascii	Activa el modo de transferencia en modo texto ASCII.
!	Permite salir a línea de comandos temporalmente sin cortar la conexión.
exit	Regresar al Shell del sistema operativo.
? *nombre de comando*	Muestra la información relativa al comando.
? o **help**	Muestra una lista de los comandos disponibles.
append *nombre del archivo*	Continúa una descarga que se ha cortado previamente.
bell	Activa/desactiva la reproducción de un sonido cuando ha terminado cualquier proceso de transferencia de archivos.
glob	Activa/desactiva la visualización de nombres largos de nuestro PC.
literal	Con esta orden se pueden ejecutar comandos del servidor de forma remota. Para saber los disponibles se utiliza: **literal help**
mkdir	Crea el directorio indicado de forma remota.
quote	Hace la misma función que **literal**.
send *nombre del archivo*	Envía el archivo indicado al directorio activo del servidor.
user	Para cambiar nuestro nombre de usuario y contraseña sin necesidad de salir de la sesión ftp.

Ejemplo de uso: Ejecución (desde mi equipo) de un proceso de descarga de archivos del equipo 16.20.1.1

(servidor) a mi equipo local:

```
loval@OvalBComm: $ Ftp -i 16.20.1.1
# bin
# lcd /usr/local/Respaldos/
```

```
# get BDUACME20080810
# quit
```

RESPALDO DE LA BD

Las bases de datos de PostgreSQL deben ser respaldadas de forma regular, ya que esta contiene datos valiosos. El procedimiento es esencialmente simple, y es importante tener una comprensión básica de las técnicas subyacentes. Existen tres enfoques fundamentales para respaldar una base de datos en PostgreSQL, cada una tiene fortalezas y debilidades:

- Pg_dump de SQL.
- Respaldo a nivel de sistema de archivos.
- Respaldos en línea.

En el presente documento nos enfocamos el uso del comando pg_dump para efectuar dicho respaldo.

Microsoft propone una estrategia para asegurar la disponibilidad de los datos (https://msdn.microsoft.com/es-es/library/bb972245.aspx) cuando se usa la herramienta que ellos fabrican: SQL Server, esta misma estrategia aplica para todas las bases de datos. Esta se presenta a continuación:

Los elementos básicos de una estrategia de disponibilidad de datos son los siguientes:

- Planear el futuro.
- Comprender los registros de transacciones y saber cómo utilizarlos para restaurar datos.
- Realizar copias de seguridad de la base de datos.
- Restaurar los datos.

Creación de una estrategia de copia de seguridad de base de datos

Las copias de seguridad de la base de datos son una parte fundamental en la creación de esta estrategia; sin una estrategia de copias de seguridad efectiva podríamos encontrarnos en una situación en la que tengamos una base de datos corrupta pero no las suficientes copias de seguridad para restaurarla.

Los tipos de fallas que podrían ocurrir son las siguientes, entre otras:

- Datos inválidos del usuario.
- Fallo en el disco duro.
- Fallo en el servidor.

Para evitar perderlo todo a causa de un fallo, sigue las siguientes recomendaciones:

- Realiza copias de seguridad con frecuencia (esto depende del uso de la base de datos).
- Mantén copias de seguridad completas fuera del sitio.
- Realiza comprobaciones de consistencia con cierta frecuencia.
- Administra tus copias de seguridad con efectividad.

MODELOS DE RECUPERACIÓN

SQL soporta los siguientes 3 modelos de recuperación:

A) Recuperación completa: Es el modelo más completo; si se produce un fallo en el disco duro, te permite recuperar la base hasta el momento justo del fallo o en cualquier momento en el tiempo. Para poder lograr esto se registran todas las operaciones, lo que hace que el registro crezca demasiado ya que las operaciones masivas también se registran. Esta es una característica muy poderosa cuando se tiene una base de datos 24 x 7; nos permite asegurar que se pierda la menor cantidad de modificaciones posible.

B) Registro masivo: Es una copia de seguridad completa. No obstante, si falla el disco duro, puede recuperarse con el modelo de copia masiva pero no te permite recuperar la base hasta cualquier momento en el tiempo.

C) Recuperación Simple: Es el modelo más sencillo de todos, ocupa el menor espacio en disco y es el que ocupa menos recursos del sistema, pero también lo expone a mayores pérdidas de datos; este modelo no nos permite recuperar hasta cualquier momento en el tiempo ni hasta el momento del fallo.

Todos estos modelos tienen ventajas y desventajas; determinar cuál es el mejor de ellos depende de sus requerimientos individuales. Por ejemplo, una base de datos que tenga muchas transacciones y necesite recuperarse completamente lo más pronto posible se beneficiaría con el modelo de recuperación completa; por el contrario, una base de datos que haya tenido muchas actualizaciones masivas y no necesite recuperar las transacciones individuales de los usuarios podría utilizar el modelo de registro masivo; por último, el modelo simple se utiliza en aplicaciones que no sean cruciales o en aplicaciones en desarrollo.

El respaldo de las bases de datos se hace fácilmente, antes de realizarlo hay que tomar en cuenta lo siguiente:

- Para respaldar toda la base de datos, sin que no haya nadie trabajando en ella, es necesario detener postgres.
- Para asegurarse de que no hay nadie trabajando es necesario apagar el servicio postgres o hacerlo fuera del horario laboral. El procedimiento de abajo describe los pasos para realizar el respaldo con la consola del servidor, es decir desde el equipo en el que está el sistema.

Nota: Para realizarlo desde una estación de trabajo, es necesario conectarse al servidor con aplicaciones como "VNC", "Rdesktop" o "Secure Shell".

Procedimiento manual de respaldo de postgres:

Abrir una ventana de Terminal. Una vez abierta la terminal, cambiarse a usuario "root" ingresando el comando "su" y haciendo clic en la tecla "Enter" para ejecutarlo. Cuando la solicite, ingresar la contraseña de "root" y hacer clic en la tecla "Enter". Una vez que se encuentra como "superusuario" o "root", hay que detener el servidor "postgres". Se ingresa la instrucción: */etc/init.d/ postgresql stop*

Cuando despliegue el siguiente renglón o el "prompt" (representado por un signo de número "#"), ya habrá detenido el postgres. Después de detenerlo hay que levantarlo con la instrucción: */etc/init.d/postgresql start*. Cuando despliegue el siguiente renglón o el "prompt" (representado por un signo de número "#"), ya habrá levantado el postgres.

Ahora, levantado el postgres, se puede ejecutar la instrucción para respaldar la base de datos. Si no existe, conviene crear un directorio donde colocar los archivos con los respaldos. Si existe, se puede ir a ese directorio con la instrucción: ***cd nombre_y_ruta_del _directorio*** Por ejemplo: ***cd /usr/ local/Respaldos/*** más "Enter".

Una forma de saber en qué directorio se encuentra, es ingresando la instrucción "pwd" más "Enter"; esto desplegará el nombre y la ruta del directorio.

Después de colocarse en el directorio donde se va a generar el respaldo, se ingresa la instrucción para realizarlo. Esta instrucción es la siguiente:

```
pg_dump -C -u uacme > BDUACME20080810 + "Enter".
```

El sistema la solicita que ingrese la contraseña. Al ingresarla, los caracteres escritos no se despliegan. Después de ingresar la contraseña, se hace clic en la tecla "Enter". El cursor se coloca en el siguiente renglón y comienza a producirse el archivo de respaldo de la base de datos. Mientras el cursor no

pase al siguiente renglón, no se debe ingresar nada, pues está generando el respaldo. El tiempo que demora en hacerlo depende del tamaño de la base de datos. Cuando termine, el cursor se coloca en el siguiente renglón. Si se desea verificar que haya producido el archivo, se puede ingresar una de las siguientes instrucciones:

```
"ls -la nombre_del_archivo"
"ls -la primeras_letras_del_nombre_del_archivo asterisco"
```

Por ejemplo: `"ls -la nombre_del_archivo" + "Enter".`

Esto listará el archivo recién creado, desplegando además su tamaño y la hora y fecha de creación.

Procedimiento automático de respaldo de postgres:

La idea del respaldo automático es que se ejecute cuando no hay nadie laborando en las oficinas, para lograrlo es necesario auxiliarse del sistema CRON. La política de la UACME es ejecutar el respaldo a la 23:00 horas de la noche, por lo cual es necesario apagar y encender el servidor PostgreSQL antes de la ejecución del respaldo, digamos unos 10 minutos antes se apaga y 5 minutos después lo encendemos.

Pasos a efectuar para el respaldo automático:

Si por casualidad el primer pensamiento es poner las instrucciones para pg_dump en un script, lamento desilusionarlo. Tendríamos un pequeño problema, y es que el comando se quedará frenado pidiéndonos la contraseña del usuario que quiere hacer el backup. Como siempre, tenemos una salida. Nuestro nuevo mejor amigo en este caso será pgpass. Pgpass es una variable de entorno. Este archivo no se crea por defecto, y varía levemente el procedimiento si lo usamos en Linux o en Windows. En ambos casos, el archivo contendrá la misma información:

```
host:puerto:basededatos:usuario:contraseña
```

CASO LINUX

En el ejemplo, el encargado de realizar el backup es el usuario root. Para que funcione, siempre y cuando tengamos los privilegios, vamos dentro de la carpeta /root y creamos el archivo .pgpass. Una forma rápida, y presuponiendo que estamos firmados como root, sería la siguiente.

```
Echo "16.20.1.1:5432:mibase:miusuario:micontraseña" >> ~/.pgpass
```

Ya resuelto el detalle de la contraseña. Ahora, simplemente, creamos un archivo que será el que vamos a programar para que se ejecute automáticamente y nos realice el backup. Dentro de nuestro script pondremos el comando:

```
pg_dump -i -h 16.20.1.1 -p 5432 -U miuser -F c -b -v -f "/home/
oval/backup/uacme.backup" uacme
```

-- Modifique los directorios apropiados a su equipo, asegurese de tener
-- todos los derechos sobre los directorios a usar.

Ahora cuando el script necesite la contraseña para conectarse, la tomará del Pgpass... y asunto resuelto.

Pasos a efectuar en el cron para ejecución automática en Linux:

Es probable que necesite información adicional del editor vi, consulte su manual de linux.

- *Abra una terminal del sistema con el usuario root.*
- *Ejecute el comando crontab -i*
- *Agregue los siguientes renglones (teclee la letra i)*
  *40 22 * * * /**etc/init.d/postgresql stop***
  *45 22 * * * /**etc/init.d/postgresql start***
  *0 23 * * * pg_dump -C -u uacme > /**home/loval/backup**/BDUACME20080810*
- *Grabe los cambios: wq*
- *Modifique la hora de su sistema (cambie la hora a 22:37)*
- *Espere la hora indicada y verifique si el archivo está generado, usando el comando "ls" que se le explicó en el procedimiento de respaldo manual.*

Caso Windows

Para usarlo, debemos crear la carpeta postgresql dentro de c:\documents and settings\(usuario que correrá la tarea)\datos de programa. Dentro de esa nueva carpeta, pondremos el archivo pgpass.conf. Ahora sí, dentro del archivo, podríamos tener los siguientes parámetros:

```
16.20.1.1:5432:mibase:miusuario:micontraseña
```

Pasos a efectuar en la aplicación "Tareas Programadas" para ejecución automática en Windows:

Ejecute la aplicación "tareas programadas" siga al asistente de configuración y agregue los siguientes comandos, en los mismos horarios que se ejecutan en Linux:

```
c:\postgresql stop
/etc/init.d/postgresql start
pg_dump -C -u uacme > c:\respaldos\BDUACME20080810
```

Exportación e Importación de datos.

Un nuevo problema aparece cuando los directivos de las empresas u organismos públicos deciden hacer cambios de plataforma a los sistemas de información, lo cual es un trabajo a lo cual un profesional de la computación se puede enfrentar en su vida profesional.

Exportación de Datos.

Suponga que el rector de UACME ha decidido cambiar la base de datos PostgreSQL por la que se denomina MySQL (www.sun.com/software/products/mysql/getit.jsp), por lo que ha solicitado al departamento de Tecnologías de Información que se ponga en marcha el proyecto pertinente. La base de datos MySQL, va a funcionar sobre un equipo al que se la ha instalado un sistema operativo Linux. Sin embargo, el rector ha sugerido que no está dispuesto a pagar personal adicional para que capture la información del sistema viejo (sobre PostgreSQL) al nuevo (sobre MySQL), por lo que solicita que se busque la forma de hacer un traspaso de información de un sistema a otro.

Pasos a efectuar para Exportar datos (PostgreSQL -> MySQL):
Genere el archivo de datos de texto para las tablas en equipo origen (Windows).

```
-- Cuidado con la ruta donde se almacena el archivo de texto
-- "/area_trabajo/ es equivalente a decir "c:\area_trabajo\"
-- el directorio area_trabajo ya debe de existir
copy profesores to '/area_trabajo/prof.txt' with delimiter ',';
-- replique el comando para cada una de las tablas, cambiando nombre de tabla
-- y el nombre del archivo de texto
```

Verifique que esté instalado el servidor FTP en el equipo destino (Linux-Ubuntu ver7.10)

-- Instale el archivo vsftpd (servidor FTP)
-- verifique los pasos en la página:
-- guia-ubuntu.org/index.php?title=Servidor_de_FTP

Asigne dirección IP estática en el equipo destino (Linux)

-- Use el comando IFCONFIG ó en el menú sistema->configuración->red

Cree una cuenta de usuario con privilegios de administrador en el equipo destino (Linux)

-- Menú sistema->administración->Usuarios y grupos
-- Indique como directorio raíz de esta cuenta /home/<usuario>

Verifique la ruta en el equipo destino (Linux), usando una terminal

```
cd /home/<usuario>
ls -l
```

Cambiarse al directorio"c:\area_trabajo\" en el equipo origen (Windows)

```
cd /area_trabajo
```

Envíe los archivos de texto generados desde el equipo origen (Windows)

```
$ ftp -i 16.20.10.20
# bin
# put prof.txt
..
-- PUT se repite por cada archivo que se envía
# quit
```

Instale el servidor MySQL en el sistema destino (Linux)

-- siga los pasos que le indica Ubuntu (gestor de paquetes)

Ejecute el cliente de MySQL en una terminal.

```
mysql -u root -p
```

Construya la base de datos en el sistema destino (Linux – MySQL).

```
create database uacme;
use uacme;
```

Construya las tablas en el sistema destino (Linux – MySQL).

```
Create Table Profesores (
idprofe int(4) PRIMARY KEY,
idtab int(4),
nombre varchar(40),
maximo varchar(40),
sueldo int(4)
);
```

Cargue los datos en el sistema destino (Linux-MySQL)

Para leer estructura del comando LOAD DATA INFILE leer las páginas http://mirror.hostfuss.com/mysql/doc/refman/5.0/es/load-data.html ó http://dev.mysql.com/doc/refman/5.0/en/load-data.html

```
mysql> LOAD DATA INFILE '/home/<usuario>/prof.txt' INTO TABLE
profesores
FIELDS TERMINATED BY ',';
```

-- Ejecute la carga de cada archivo a su tabla correspondiente

Verifique que todos los registros estén cargados correctamente.

```
Select * from profesores;
```

-- Verifique que cada una de las tablas ha sido cargada correctamente

Importación de datos.

Después de tres años de trabajar con MySQL, el rector ha decidido que el pago de mantenimiento del equipo SunFire es demasiado caro y nuevamente decide a que se regrese a trabajar con PostgreSQL sobre un equipo Linux. Como ya se hizo anteriormente solicita que la información del sistema de información se ejecute de forma automatizada.

Pasos a efectuar para Importar datos:

Genere el archivo de datos de texto para la tabla en sistema origen (Linux-MySQL).

```
SELECT * INTO OUTFILE '/tmp/profesores.txt'
FIELDS TERMINATED BY ',' ENCLOSED BY '' FROM Profesores;
```

-- Aplique para cada una de las tablas del sistema.
-- verifique que el directorio tmp existe

Envíe los archivos de texto generados al equipo destino (Windows).

-- Usando ftp (siga las mismas instrucciones que en la exportación),
-- solo que esta vez será enviada la información en sentido contrario.

Construya las tablas en el sistema destino (Windows - PostgreSQL).

-- Use las tablas que tenemos ya construidas, limpie la información
-- de cada tabla usando DELETE FROM ...
-- o elimine la BD (DROP DATABASE UACME;) y vuelva a construir todas
-- las tablas.

Cargue los datos en el sistema destino (Windows - PostgreSQL).

```
copy profesores from '/area_trabajo/profesores.txt' with delimiter ',';
```

-- Ejecute este comando una vez por cada tabla.

TRABAJO ADICIONAL

- Restaure los archivos generados con el respaldo, usando el comando pg_restore.
- Respalde la tabla de Profesores de la BD uacme.
- Averigüe si los comandos de Importación y Exportación de datos existen en las bases de datos: DB2, Informix, Oracle, y Sybase. De ser posible, consiga los DBMS vía descarga en línea y efectúe las pruebas necesarias.
- Inserte los datos de la tabla Profesores en una hoja de cálculo (Excel de MS Office, Calc de OpenOffice, etc), y busque una estrategia para convertir el formato propietario a un formato de archivo de texto delimitado por comas, el resultado de esta conversión debe de ser transferido a la tabla Profesores de PostgreSQL o MySQL usando los comandos que ya ha practicado.

REFERENCIAS

Douglas Korry y Susan Douglas. (2003) PostgreSQL A comprehensive guide to building, programming and administering Postgre SQL databases. (2nd. Edition). Sams

Elmasri, R.; Navathe, S.B. (2002). Fundamentos de Sistemas de Bases de Datos. 3ª Edición. Addison-Wesley

Garcia-Molina, Jeffrey Ullman y Jennifer Widom. (2008). Database systems: the complete book. Prentice-Hall.

Momjiam Bruce. (2001). PostgreSQL Introduction and Concepts. Boston: Addison-Wesley Logman Publishing Co. Inc.

Silberschatz Abraham, Henry Korth y S. Sudarshan. (2006). Fundamentos de Base de Datos (5a. Ed.). España: McGraw-Hill.

The PostgreSQL Global Development Group. (2015). Manual de postgreSQL. Disponible enwww.postgresql.org.