

Replicación de Bases de Datos

Trabajo coordinado de un conjunto de servidores que aporta alta disponibilidad y *balance de carga*.

Una consulta sobre una base de datos, puede ser resuelta por distintos servidores, según la carga de trabajo de cada uno, se puede obtener un mayor rendimiento en sistemas con alta carga de trabajo.

Esta metodología podría llamarse consulta paralela

Replicación de Bases de Datos

Una misma base de datos podría estar copiada total o parcialmente en distintos SGBD's, a esto lo llamamos una *base de datos replicada*.

Responder a consultas sobre una base de datos replicada incrementa el rendimiento, la consulta podría resolverse por el servidor más próximo al cliente y la capacidad de respuesta se incrementa por la cantidad de servidores disponibles.

Replicación de Bases de Datos

Las consultas sobre una base de datos pueden ser:

read/only (no modifican datos, son simples consultas)

read/write (modifican datos).

Una transacción esta formada por un conjunto de queries read/write que deben ejecutarse de forma atómica.

Una transacción no puede ejecutarse por la mitad porque ello dejaría a la base de datos en un estado inconsistente.

Replicación de Bases de Datos

Es fácil coordinar varios servidores cuando se trata de queries *read/only*.

El problema se presenta en una replicación cuando hay queries *read/write*, ya que, el dato modificado debe ser guardado en todas las copias de la base de datos y se debe obtener el dato de forma consistente (el mismo valor), desde todos los servidores.

Replicación de Bases de Datos

Una solución a este problema es:

Permitir que un solo servidor modifique los datos, *servidor maestro*.

Hay otros servidores (*servidor esclavo*) que replican el dato modificado por el servidor maestro.

Replicación de Bases de Datos

Cuando un servidor esclavo no puede ser conectado hasta que no sea promovido por el maestro, se los denomina *warm standby server*.

Un servidor esclavo que puede ser conectado y solo sirve queries read/only se lo denomina *hot standby server*.

Replicación de Bases de Datos

Las soluciones propuestas a los problemas de replicación pueden clasificarse de distintas formas:

Solución sincrónica es aquella en donde el dato modificado no es considerado *comiteado* hasta que todos los servidores lo hayan comiteado, todos los servidores del cluster devuelven un mismo dato consistente.

Solución asincrónica es aquella que permite una demora entre el commit del dato y su propagación a los demás servidores. Los servidores pueden devolver un dato no consistente, no necesariamente devolverán el mismo valor. Esta última opción se utiliza cuando la solución sincrónica es muy lenta debido al hardware disponible o bien al ancho de banda de la red.

Método de Replicación Unidireccional en Postgres 9.4

La forma mas sencilla de realizar una replica en Postgres es el método de replica Hot StandBy o replica unidireccional, esta consta de un servidor Master o Productivo y otro(s) servidores Esclavos o Stand By que recibirán los datos desde la master. El método se realiza por .ssh donde se hace replicación por rsync o replicación sincrónica.

Método de Replicación Unidireccional en Postgres 9.4

Términos

Cluster: Es el conjunto de máquinas que contienen las bases de datos PostgreSQL que intervienen en la replicación.

Nodo: Se le llama así a cada una de las bases de datos a replicar en la replicación.

Replication set: Es el conjunto de tablas a ser replicadas. En un mismo cluster pueden haber varios sets.

Método de Replicación Unidireccional en Postgres 9.4

Origen: Es el nodo principal (maestro), es el único en el que se puede escribir.

Suscriptores: Son todos los demás nodos en el cluster (esclavos), son los que reciben los datos en la réplica.

Proveedor: Es un nodo suscriptor (esclavo) que sirve como proveedor para un subconjunto de nodos en el cluster (actúa como un nodo origen pero no se permite a ninguna aplicación escribir en él).

Método de Replicación Unidireccional en Postgres 9.4

Requisitos

PostgreSQL >= 7.3.3 (7.4.8 o superior es recomendado).

Verificar que postgres este aceptando conexiones (listen_addresses='*' en postgresql.conf) Definir la estructura del cluster (a cada nodo del cluster se le debe identificar con su IP).

Definir los replication sets (Tablas y claves para las tablas que no tienen un PK). Las tablas relacionadas por un FK deberían estar en el mismo replication set.

Todos los nodos involucrados en la replicación deben estar usando un timezone reconocido por PostgreSQL, se debe setear en el archivo postgresql.conf

Crear la base de datos y el esquema de la base de datos en los nodos.

Método de Replicación Unidireccional en Postgres 9.4

Slony utiliza una aplicación llamada slonik para inicializar, moldear y actualizar las bases de datos en un clúster. Para que slony funcione, slony creará tablas en las bases de datos que participan en la replicación. Slonik ejecuta una serie de comandos en las bases de datos, como crear tablas y modificarlas para que se inicialicen para su uso con slony.

Para que Slonik pueda hacer esto, slonik debe poder conectarse a su base de datos en el host Maestro y el host esclavo.

Método de Replicación Unidireccional en Postgres 9.4

Slony no puede copiar el esquema y los cambios de esquema. Para este efecto, antes de inicializar slony, debemos copiar el DDL (la estructura / esquema) de la base de datos al nodo esclavo.

Método de Replicación Unidireccional en Postgres 9.4

Demonio Slon

Actúa como activador en la base de datos y envía los datos a los nodos esclavos. Para cada nodo en la base de datos, debe haber un demonio slon en ejecución. Para iniciar el demonio slon en el servidor:

```
slon slony_1 "dbname = db host = hostM user = postgres password=clave"
```

Método de Replicación Unidireccional en Postgres 9.4

Implementación Slony-I

Paso 1

Máquina Virtual Maestro archivo pg_hba.conf agregar

#MAESTRO

host	all	all	192.168.1.60/24	md5
------	-----	-----	-----------------	-----

#ESCLAVO

host	all	all	192.168.1.40/24	md5
------	-----	-----	-----------------	-----

#POSTGRES

host	postgres	postgres	192.168.1.60/24	md5
------	----------	----------	-----------------	-----

Máquina Virtual Esclavo archivo pg_hba.conf agregar

#MAESTRO

host	all	all	192.168.1.60/24	md5
------	-----	-----	-----------------	-----

#ESCLAVO

host	all	all	192.168.1.40/24	md5
------	-----	-----	-----------------	-----

#POSTGRES

host	postgres	postgres	192.168.1.40/24	md5
------	----------	----------	-----------------	-----

Método de Replicación Unidireccional en Postgres 9.4 Implementación Slony-I

Paso 2

Verificar que postgres este aceptando conexiones

En el archivo postgresql.conf de la máquina Maestro y la máquina Esclavo agregar en

- Connection Settings –

```
listen_addresses = '*'
```


Método de Replicación Unidireccional en Postgres 9.4 Implementación Slony-I

Paso 3

Ejecutar pgAdmin III en la máquina Maestro y crear una conexión a la IP del Maestro y al puerto de Postgres

Crear una base de Datos y una tabla para realizar un ejemplo.

Ejecutar pgAdmin III en la máquina Esclavo y crear una conexión a la IP del Esclavo y al puerto de Postgres

Crear una base de Datos con sus tablas con el mismo nombre que se uso en el Maestro.

Método de Replicación Unidireccional en Postgres 9.4

Implementación Slony-I

Paso 4

En el Maestro y en el Esclavo con pgAdmin III

en el menú

**Archivo→Opciones→Navegador→Binary Paths→Ruta Slony-I
agregar C:\Program Files\PostgreSQL\9.2\share**

**Archivo→Opciones→Navegador→Binary Paths→Ruta Binaria PG
agregar C:\Program Files\PostgreSQL\9.2\bin**

Método de Replicación Unidireccional en Postgres 9.4 Implementación Slony-I

Paso 5

Configurar Puertos en Maestro y Esclavo

En panel de control de Windows → FireWall de Windows → Configuración Avanzada → Reglas de Entrada → Nueva Regla → Puerto → TCP → Puertos locales específicos 5432 → Permitir la conexión → Dominio Privado Público → Nombre postgres → Finalizar

Método de Replicación Unidireccional en Postgres 9.4

Implementación Slony-I

Paso 6

En el Maestro crear el siguiente script en PostgreSQL -> 9.2 -> bin

```
cluster name=slony_1;
node 1 admin conninfo = 'dbname=ReplicacionPrueba host=192.168.1.60 user=postgres password=xxxxxx';
node 2 admin conninfo = 'dbname=ReplicacionPrueba host=192.168.1.40 user=postgres password=xxxxxx';
init cluster (id=1,comment='nodo maestro');

create set (id=1,origin=1,comment='lista de tablas');

set add table (set id=1,origin=1,id=1,fully qualified name='public.tablareplicacion',comment='mi tabla');

store node (id=2,comment='nodo esclavo',EVENT NODE=1);

store path(server=1,client=2,conninfo='dbname=ReplicacionPrueba host=192.168.1.60 user=postgres
password=xxxxxx');
store path(server=2,client=1,conninfo='dbname=ReplicacionPrueba host=192.168.1.40 user=postgres
password=xxxxxx');

store listen(origin=1,provider=1,receiver=2);
store listen(origin=2,provider=2,receiver=1);
```

Método de Replicación Unidireccional en Postgres 9.4

Implementación Slony-I

Paso 7

En el Esclavo crear el siguiente script en PostgreSQL -> 9.2 -> bin

```
cluster name= slony_1;
```

```
node 1 admin conninfo = 'dbname= ReplicacionPrueba host=192.168.1.60 user=postgres password=suclave';  
node 2 admin conninfo = 'dbname= ReplicacionPrueba host=192.168.1.40 user=postgres password=suclave';
```

```
subscribe set(id=1,provider=1,receiver=2,forward=yes);
```

```
WAIT FOR EVENT (  
    ORIGIN = ALL,  
    CONFIRMED = ALL,  
    WAIT ON = 1  
);
```

Método de Replicación Unidireccional en Postgres 9.4 Implementación Slony-I

Paso 8

Ejecutar el script en el Maestro en el directorio bin de postgresql

C:\...\...\bin>slonik maestro.txt

Ejecutar el script en el Esclavo en el directorio bin de postgresql

C:\...\...\bin>slonik suscriptor.txt

Método de Replicación Unidireccional en Postgres 9.4 Implementación Slony-I

Paso 9


Ejecutar en el Maestro en el directorio bin de postgresql


```
C:\...\...\bin>slon slony_1 “dbname=ReplicacionPrueba user=postgres  
password=clave”
```


Ejecutar en el Esclavo en el directorio bin de postgresql


```
C:\...\...\bin>slon slony_1 “dbname=ReplicacionPrueba user=postgres  
password=clave”
```


Implementación en Máquinas Virtuales


**Windows 7_1**
Corriendo


**debian3**
Apagada

**Windows 7_2**
Apagada


**Huayra-3-2**
Apagada


**Win-10-x64**
Apagada


**debian9_32_1**
Apagada


**Debian 7 (3)**
Apagada


IDE secundario maestro: [Unidad óptica] Vacío
Controlador: Controlador SATA
Puerto SATA 0: Windows-7-aux-disk1.vmdk (Normal, 50,00 GB)


**Audio**
Controlador de anfitrión: Windows DirectSound
Controlador: Audio Intel HD


**Red**


**USB**


**Carpets compartidas**
Carpets compartidas: 1


**Descripción**
192.168.1.60
Firebird 3.0 Flame Robin
Replicación Postgresql - slony-1 (Mestro)


**Windows 7_2**
Apagada

**Huayra-3-2**
Apagada


**Win-10-x64**
Apagada


**debian9_32_1**
Apagada


**Debian 7 (3)**
Apagada


**Debian_4**
Apagada

Controlador de anfitrión: Windows DirectSound
Controlador: Audio Intel HD

**Red**

**USB**

**Carpets compartidas**
Carpets compartidas: 1

**Descripción**
192.168.1.40
SGBD
Firebird (execute procedure en debian3 y debian3_2)
ClienteServidorSwitch (Servidor debian_1)
Replicación Postgresql - Slony-1 (Esclavo)
ClienteServidorMiniTelnet en Csharp

Paso 1 en Windows 7_1 192.168.1.60 Maestro

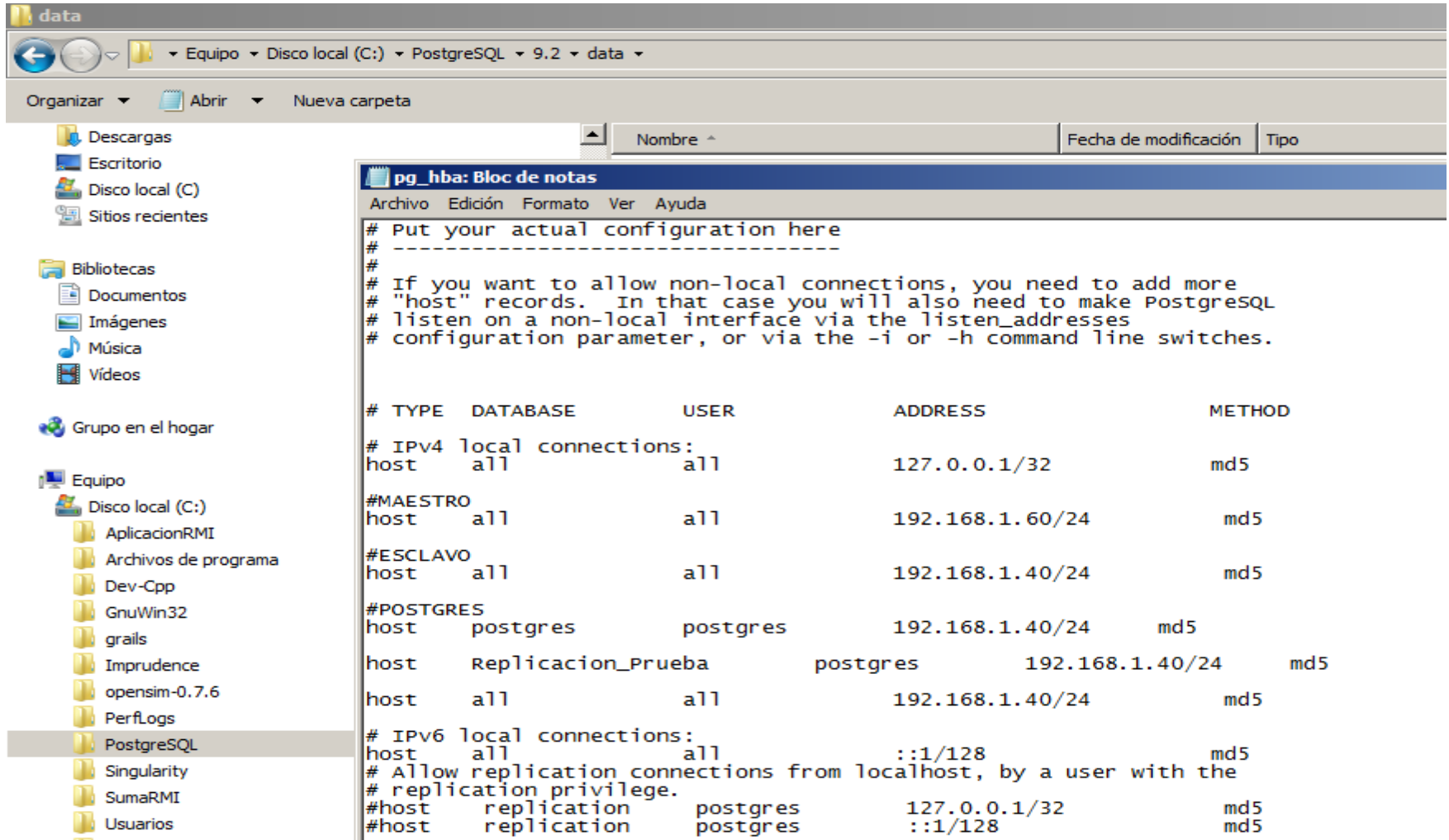
Archivos de programa > PostgreSQL > 9.2 > data > Buscar data

Organizar pg_hba: Bloc de notas

Archivo Edición Formato Ver Ayuda

#	TYPE	DATABASE	USER	ADDRESS	METHOD
# IPv4 local connections:					
host	all		all	127.0.0.1/32	md5
#MAESTRO					
host	all		all	192.168.1.60/24	md5
#ESCLAVO					
host	all		all	192.168.1.40/24	md5
#POSTGRES					
host	postgres		postgres	192.168.1.60/24	md5
#host	all		all	192.168.1.40/24	md5
# IPv6 local connections:					
host	all		all	:::1/128	md5
# Allow replication connections from localhost, by a user with the					
# replication privilege.					
#host	replication		postgres	127.0.0.1/32	md5
#host	replication		postgres	10.0.118.80/32	md5
#host	replication		postgres	10.0.125.102/32	md5

Paso 1 en Windows 7_2 192.168.1.40 Esclavo



The screenshot shows a Windows 7 desktop environment. On the left, a file explorer window is open, displaying the 'data' folder within the 'PostgreSQL 9.2' directory on the local drive (C:). The file explorer's left sidebar shows the 'Equipo' (Computer) section expanded, with the 'PostgreSQL' folder selected. The main pane of the file explorer shows the contents of the 'data' folder, which include a file named 'pg_hba'.

Overlaid on the file explorer is a Notepad application window titled 'pg_hba: Bloc de notas'. The Notepad window displays the contents of the 'pg_hba.conf' file, which is a configuration file for PostgreSQL. The file contains comments and a table of host-based authentication rules. The table has five columns: TYPE, DATABASE, USER, ADDRESS, and METHOD. The rules are as follows:

TYPE	DATABASE	USER	ADDRESS	METHOD
host	all	all	127.0.0.1/32	md5
host	all	all	192.168.1.60/24	md5
host	all	all	192.168.1.40/24	md5
host	postgres	postgres	192.168.1.40/24	md5
host	Replicacion_Prueba	postgres	192.168.1.40/24	md5
host	all	all	192.168.1.40/24	md5
host	all	all	:::1/128	md5
host	replication	postgres	127.0.0.1/32	md5
host	replication	postgres	:::1/128	md5

```

# (change requires restart)

#-----
# CONNECTIONS AND AUTHENTICATION
#-----

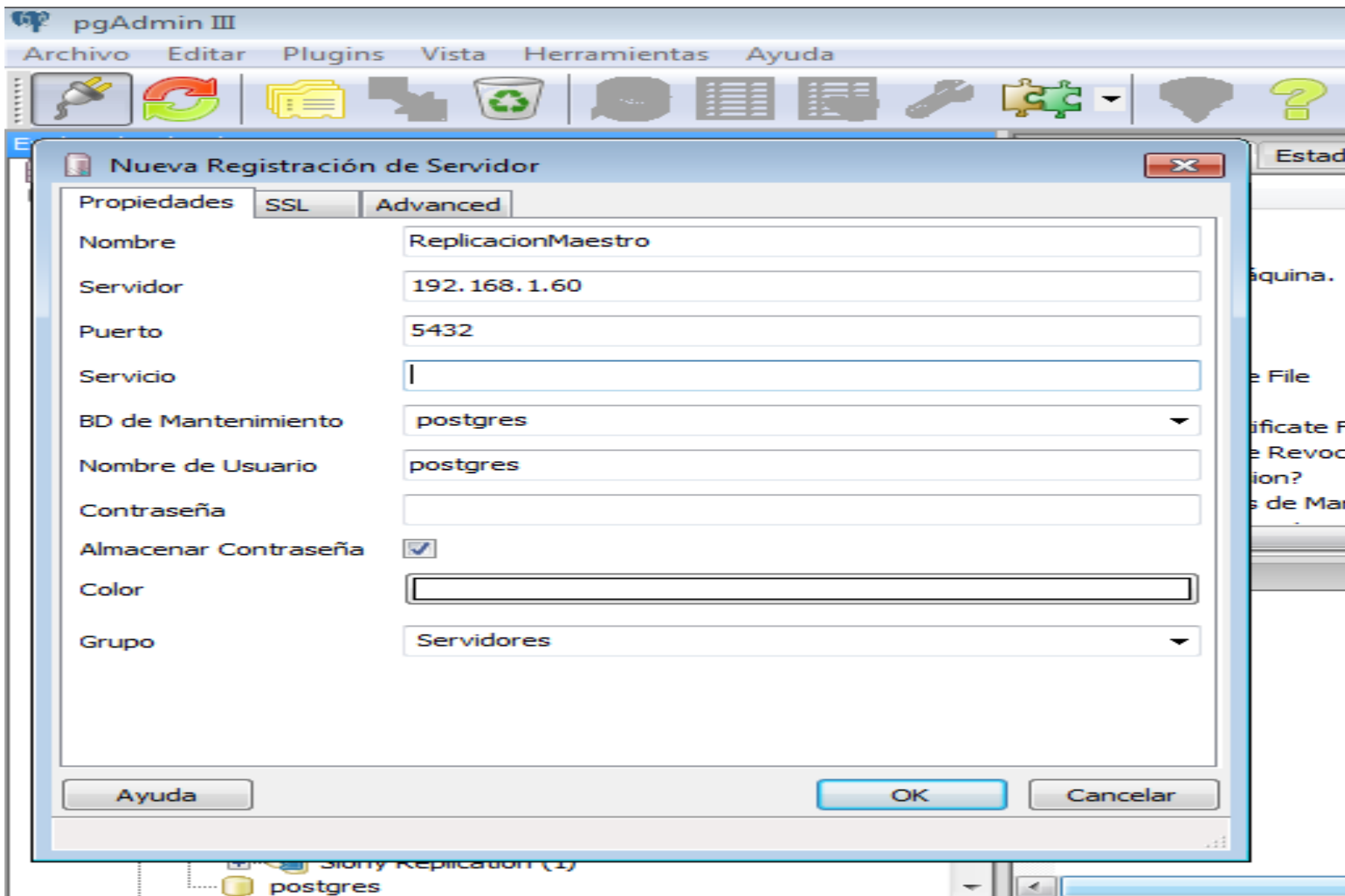
# - Connection Settings -

listen_addresses = '*'          # what IP address(es) to listen on;
                                # comma-separated list of addresses;
                                # defaults to 'localhost'; use '*' for all
                                # (change requires restart)
port = 5432                     # (change requires restart)
max_connections = 100           # (change requires restart)
# Note: Increasing max_connections costs ~400 bytes of shared memory per
# connection slot, plus lock space (see max_locks_per_transaction).
#superuser_reserved_connections = 3 # (change requires restart)
#unix_socket_directory = ''       # (change requires restart)
#unix_socket_group = ''          # (change requires restart)
#unix_socket_permissions = 0777  # begin with 0 to use octal notation
                                # (change requires restart)
#bonjour = off                  # advertise server via Bonjour
                                # (change requires restart)
#bonjour_name = ''              # defaults to the computer name
                                # (change requires restart)

# - Security and Authentication -

#authentication_timeout = 1min   # 1s-600s
#ssl = off                       # (change requires restart)
#ssl_ciphers = 'ALL:!ADH:!LOW:!EXP:!MD5:@STRENGTH' # allowed SSL ciphers
                                # (change requires restart)
#ssl_renegotiation_limit = 512MB # amount of data between renegotiations

```

The screenshot shows the pgAdmin III interface. The left pane, 'Explorador de Objetos', displays the database hierarchy: Grupos de Servidores > Servidores (2) > PostgreSQL 9.2 (localhost:5432) > ReplicacionMaestro (192.168.1.60:5432) > Bases de Datos (2) > ReplicacionPrueba. The 'ReplicacionPrueba' database is selected, showing its contents: Catálogos (3), Extensions (1), Esquemas (1) > public > Tablas (1) > tablareplicacion. The 'tablareplicacion' table is expanded, showing columns 'id' and 'nombre', and other objects like restricciones, índices, reglas, disparadores, funciones disparadoras, and vistas.

The right pane, 'Propiedades', shows the properties of the 'ReplicacionPrueba' database:

Propiedad	Valor
Nombre	ReplicacionPrueba
OID	65883
Propietario	postgres
ACL	
Tablespace	pg_default
Espacio de tabla por defecto	pg_default
Codificado	UTF8
Colación	Spanish_Spain.1252
Tipo caracter	Spanish_Spain.1252
Schema por defecto	public
Tabla ACL por defecto	
Secuencia ACL por defecto	

The bottom pane, 'Panel SQL', contains the following SQL script:

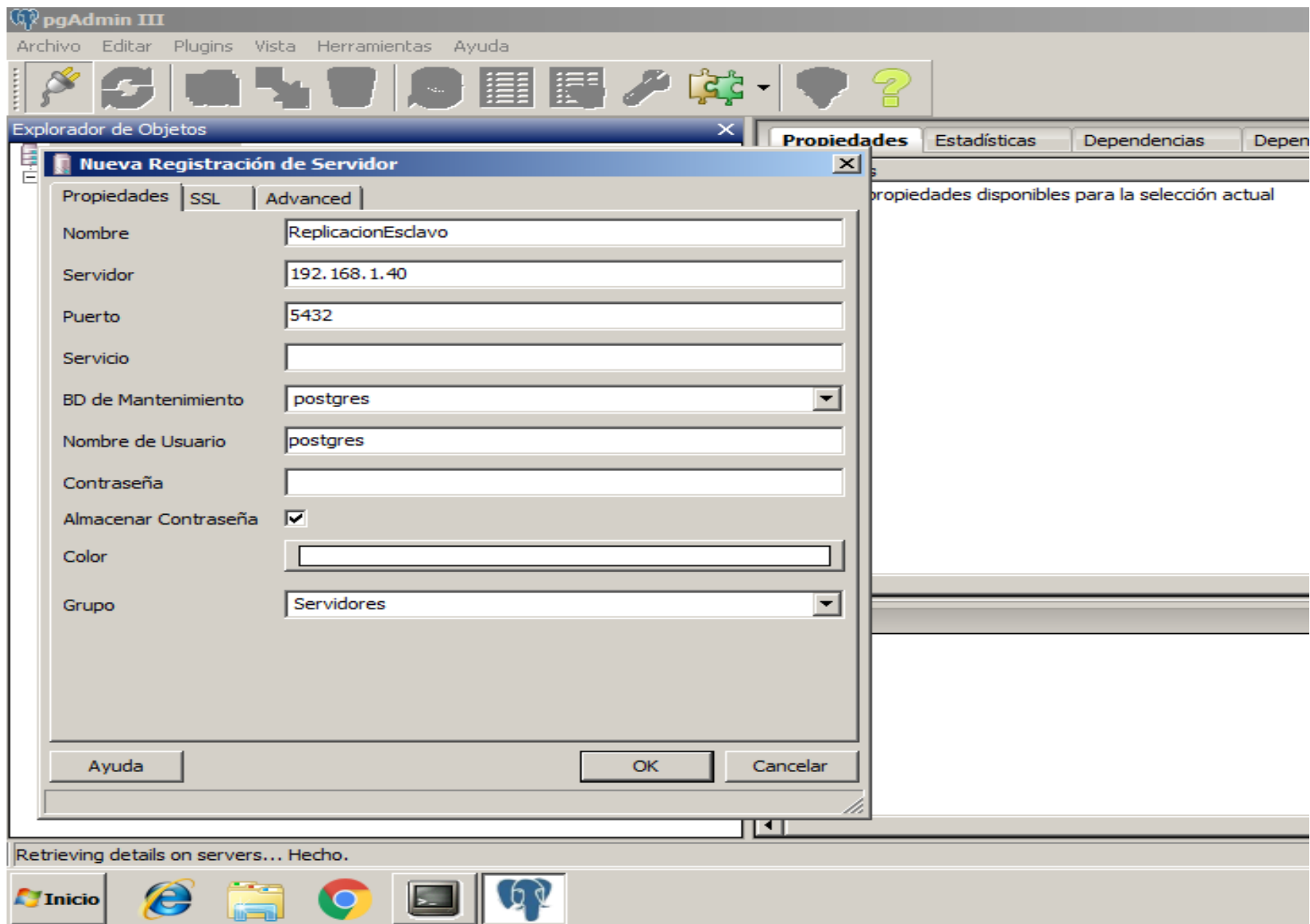
```
-- Database: "ReplicacionPrueba"

-- DROP DATABASE "ReplicacionPrueba";

CREATE DATABASE "ReplicacionPrueba"
  WITH OWNER = postgres
      ENCODING = 'UTF8'
      TABLESPACE = pg_default
      LC_COLLATE = 'Spanish_Spain.1252'
      LC_CTYPE = 'Spanish_Spain.1252'
      CONNECTION LIMIT = -1;
```

The status bar at the bottom indicates 'Retrieving details on database ReplicacionPrueba... Hecho.'

Paso 3-1 en Windows 7_2 192.168.1.40 Esclavo – Crear Conexión



The screenshot displays the pgAdmin III application window. The main pane, titled 'Explorador de Objetos', shows a hierarchical tree of database objects. Under 'ReplicacionEsclavo (192.168.1.40:5432)', the 'Bases de Datos (4)' folder is expanded, showing 'ReplicacionPrueba'. Within this database, the 'public' schema is selected, and its 'Tablas (1)' folder is expanded, showing the 'tablareplicacion' table. The table's structure is visible, including columns 'id' and 'nombre'. The right pane, titled 'Propiedades', shows the properties of the selected object. The 'Panel SQL' at the bottom right contains the following SQL commands:

```
-- Schema: public  
-- DROP SCHEMA public;  
  
CREATE SCHEMA public  
  AUTHORIZATION postgres;  
  
GRANT ALL ON SCHEMA public  
GRANT ALL ON SCHEMA public
```

The status bar at the bottom indicates 'Retrieving details on table tablareplicacion... Hecho.'

Maestro – Crear Ruta Slony-I

