PREVIRED

05/2016

psql

Introducción

- **psql**
  - –Herramienta por línea de comando para la interacción PostgreSQL
  - –Viene preinstalada
  - –Puede ser utilizada como herramienta de scripting
  - –También puede ser utilizado como herramienta de dump
  - –Se puede lanzar desde pgAdmin

- **Variables de Ambiente interesantes**
  - –PGHOST: Host
  - –PGPORT: Puerto
  - –PGUSER: Usuario
  - –PGPASSWORD:Password. En este caso se puede utlizar el archivo de password .psql (A partir de la versión 9.2)
  - –PSQL_HISTORY:Archivo que guarda la histora de comandos ejecutados recientemente. Por omsión es ~/.psql_history
  - –PSQLRC:Archivo de configuración

Interacción

```
utaladriz$ psql
psql (9.5.3)
Type "help" for help.

utaladriz=#utaladriz=# \?
General
\copyright             show PostgreSQL usage and distribution terms
\g [FILE] or ;         execute query (and send results to file or |pipe)  \gset [PREFIX]
execute query and store results in psql variables
\q                     quit psql
\watch [SEC]           execute query every SEC seconds

Help
\? [commands]          show help on backslash commands
\? options             show help on psql command-line options
\? variables           show help on special variables
\h [NAME]              help on syntax of SQL commands, * for all commands

Query Buffer
\e [FILE] [LINE]       edit the query buffer (or file) with external editor
\ef [FUNCNAME [LINE]]  edit function definition with external editor
\p                     show the contents of the query buffer
\r                     reset (clear) the query buffer
\s [FILE]              display history or save it to file
\w FILE                write query buffer to file

Input/Output
\copy ...              perform SQL COPY with data stream to the client host
\echo [STRING]         write string to standard output
```

Modo no interactivo

- **Ejecución de un script**
  - –psql –f archivo_script
- **Ejecución de un comando**
  - –psql –d base_datos –c “DROP TABLE IF EXISTS cliente; CREATE SCHEMA backup;”
- **Ambos comandos pueden ser utilizados como tares batch y ser ejecutados periódicamente si se utiliza una agente que permita agendar trabajos como pgAgent.**
  - –Por supuesto también es factible usar cron

Personalización de psql

- Es factible personalizar psql vía el archuvo de configuración
- En Linux/Unix, el archivo .psqlrc ubicado en el home del usuario
- En Windows el archivo se llama psqlrc.conf y se encuentra en el directorio %APPDATA%\postgresql
  - Usualmente C:\Users\username\AppData\Roaming\postgresql
- Contenido típico:

```
\pset null 'NULL'
\encoding latin1
\set PROMPT1 '%n@%M:%>%x %/# '
\pset pager always
\timing on
\set qstats92 'SELECT usename, datname, left(query,100) || ''...''
As query
FROM pg_stat_activity WHERE state != ''idle'' ;'
```

- Cada comando set debe ir en una línea

Personalización de psql

- Al momento de lanzar psql aparece en pantalla:

```
Null display is "NULL".
Timing is on.
Pager is always used.
psql (9.5.3)
Type "help" for help.
postgres@localhost:5442 postgresql_book#
```

- Algunos comandos solo funcionan en Linux/Unix y no en Windows o viceversa
- Para iniciar psql y no ejecutar psqlrc se debe usar la opción –X
- Para remover una variable se puede usar el comando \unset
  - \unset qstat92

Personalización de psql

- **Sobre el prompt**
  - %n es el usuario conectado
  - %M el servidor
  - %> el puerto
  - %x el estado de la transacción
  - %/ la base de datos
- **Al conectarse a la base de datos STI el prompt se ve de la siguiente manera:**

```
postgres@localhost:5432 sti#
```

- **Al cambiarse a SVI (Comando** `\connect svi`**)**

```
postgres@localhost:5432 svi#
```

- **Al activar el timing** `\timing` **cada vez que ejectuamos un comando muestra el tiempo de ejecución.**

```
SELECT COUNT(*) from pg_tables;
count
--------
73
(1 row)
Time: 18.650 ms
```

Personalización de psql

- **Por omisión AUTOCOMMIT está activado (On)**
  - Cada vez que se ingresa un comando SQL se hace commit
  - Cada comando se ejecuta en su propia transacción
  - Si el comando es exitoso, no se puede revertir (No es factible hacer rollback)
- **Como activar/desactivar AUTOCOMMIT**

```
\set AUTOCOMMIT off
\set AUTOCOMMIT on
```

- **Al momento de salir de psql, se hace automáticamente un rollback**
  - No olvidar hacer COMMIT si el AUTOCOMMIT está desactivado

Personalización de psql

- **En PSQL es factible crear shortcuts o atajos**
  - –Se utiliza el comando \set
- **Ejemplo: Típicamente se usa el comando** `EXPLAIN ANALYZE VERBOSE`**, que permite ver en detalle el análisis de cómo fue resuelto un query.**
  - –Es largo de tipear
  - –Es fácil cometer un error
  - –Solución: Crear un shortcut
  - –`\set eav 'EXPLAIN ANALYZE VERBOSE'`
  - –Luego se puede usar
  - –`:eav`

```
:eav SELECT COUNT(*) FROM pg_tables;
```

Algunos tips

- La flecha arriba y abajo permiten navegar por el historial de comasdasda
- El comando \! permite ejecutar un comando en el sistema operativo
  - Shell
- El comando \watch (Versión 9.3 y posteriores), permite ejecutar un comando periódicamte y observar sus resultados.
  - Ejemplo obtener el tráfico de las conexiones cada 10 segundos:

```
SELECT datname, waiting, query
FROM pg_stat_activity
WHERE state = 'active' AND pid != pg_backend_pid(); \watch 10
```

- Otro ejemplo generar logs

```
SELECT * INTO log_activity FROM pg_stat_activity;
INSERT INTO log_activity SELECT * FROM pg_stat_activity; \watch 5
```

  - El primer comando crea la tabla.
  - El segundo comienza a generar el log cada 5 segundos
- Para termiar el watch CTRLX CTRL-C.

Algunos tips

- **Muchos comandos entregan como resultados listas de objetos**
  - –Ejemplo: Como mostrar el listado de todas las tablas de un catálogo (Schema) que inicien con un prefijo

```
\dt+ pg_catalog.pg_t*
Schema     | Name                      | Type  | Owner    | Size    |
-----------+---------------------------+-------+----------+---------
pg_catalog | pg_tablespace             | table | postgres | 40 kB   |
pg_catalog | pg_trigger                | table | postgres | 16 kB   |
pg_catalog | pg_ts_config              | table | postgres | 40 kB   |
pg_catalog | pg_ts_config_map          | table | postgres | 48 kB   |
pg_catalog | pg_ts_dict                | table | postgres | 40 kB   |
pg_catalog | pg_ts_parser              | table | postgres | 40 kB   |
pg_catalog | pg_ts_template            | table | postgres | 40 kB   |
pg_catalog | pg_type                   | table | postgres | 112 kB  |
```

Algunos tips

- Para obtener los detalles de un objeto

```
\d+ pg_ts_dict
Table "pg_catalog.pg_ts_dict"
Column              | Type| Modifiers | Storage          | Stats target
--------------+------+----------+-----------------+--------------
Dictname            | name| not null  | plain            |
Dictnamespace       | oid | not null  | plain            |
dictowner           | oid | not null  | plain            |
dicttemplate        | oid | not null  | plain            |
dictinitoption      | text|           | extended         |
Indexes:
"pg_ts_dict_dictname_index" UNIQUE, btree (dictname, dictnamespace)
"pg_ts_dict_oid_index" UNIQUE, btree (oid)
Has OIDs: yes
```

Algunos tips

- Para importar y exportar datos desde y hacia archivos de texto existe el utilitario `\copy`
- El delimitador por omisión es el Tab (Es factible cambiarlo)
- El fin de línea separa las filas
- Antes de importar se debe crear una tabla que calce en el número de columnas y tipos de datos.
- La importación sucede en una sola transacción.
  - Es decir, si hay errores, nada se importa
- Una alternativa es crear una tabla con tipos genéricos (Varchar por ejemplo) y luego realizar transformaciones
  - `\connect base_datos`
  - `\cd /archivos/data`
  - `\copy mi_tabala FROM datos.csv CSV`
- CSV indica que el delimitador es la “,”
- Para delimitadores no estándar como |

```
\copy sometable FROM somefile.txt DELIMITER '|';
```

- No confundir el comando `\copy` con el comando SQL COPY

Algunos tips

- También podemos exportar datos

```
\connect base_datos
\copy (SELECT * FROM tabla WHERE s01 ~ E'^[0-9]+' ) TO '/test.tab'
WITH DELIMITER E'\t' CSV HEADER
```

- En este caso el delimitador es el Tab pero se usa CSV HEADER para que agregue el encabezado
- Otro ejemplo

```
\connect base_dato
\copy tabla TO '/datos.csv' WITH CSV HEADER QUOTE '"' FORCE
QUOTE *
```

- Force Quote *, asegura que todas las columnas van con comillas. Por omisión se usa ", para mayor claridad se ha indicado explícitamente el carácter a utilizar como comillas.

Algunos tips

- Es factible importar cosas ejecutando comandos del shell, como dir, ls, curl, etc.

```
\connect base_datos
CREATE TABLE directorio (archivo text);
\copy directorio FROM PROGRAM 'dir C:\projects /b'
```

- psql también es capaz de generar reportes básicos en HTML

```
psql -d base_datos-H -c
"SELECT category, count(*) As num_per_cat
FROM pg_settings
WHERE category LIKE '%Query%'
GROUP BY category
ORDER BY category;" -o test.html
```

category	num_per_cat
Query Tuning / Genetic Query Optimizer	7
Query Tuning / Other Planner Options	5
Query Tuning / Planner Cost Constants	6
Query Tuning / Planner Method Configuration	11
Statistics / Query and Index Statistics Collector	6

(5 rows)

Algunos tips

- Eso es solo la tabla. Para generar un HTML FULL:

```
\o settings_report.html
\T 'cellspacing=0 cellpadding=0'
\qecho '<html><head><style>H2{color:maroon}</style>'
\qecho '<title>PostgreSQL Settings</title></head><body>'
\qecho '<table><tr valign=''top''><td><h2>Planner Settings</h2>'
\x on
\t on
\pset format html
SELECT category, string_agg(name || '=' || setting, E'\n' ORDER BY name) As settings
FROM pg_settings
WHERE category LIKE '%Planner%'
GROUP BY category
ORDER BY category;
\H
\qecho '</td><td><h2>File Locations</h2>'
\x off
\t on
\pset format html
SELECT name, setting FROM pg_settings WHERE category = 'File Locations' ORDER BY
name;
\qecho '<h2>Memory Settings</h2>'
SELECT name, setting, unit FROM pg_settings WHERE category ILIKE '%memory%' ORDER
BY name;
\qecho '</td></tr></table>'
\qecho '</body></html>'
\o
```

Algunos tips

- Redirecciona la salida a un archivo
- Parámetros para la generación de la tabla HTML
- HTML adicional
- Expand mode. Repite los encabezados de columnas para cada fila y cada fila se genera como una fila separada
- Establece que la salida del query será una taba HTML
- string_agg(), se aregó en PostgreSQL 9.0, concatena todas las propiedades de una misma categoría en una columna.
- Desactiva el modo expandido. El segundo y el tercer query genera una fila por fila de la tabla
- per table row.
- Activa tuples mode. Cuando está encendido los encabezados de columna y el contador de filas se omite.

Algunos tips

Planner Settings

category	Query Tuning / Other Planner Options
settings	constraint_exclusion=partition cursor_tuple_fraction=0.1 default_statistics_target=100 from_collapse_limit=8 join_collapse_limit=8
category	Query Tuning / Planner Cost Constants
settings	cpu_index_tuple_cost=0.005 cpu_operator_cost=0.0025 cpu_tuple_cost=0.01 effective_cache_size=16384 random_page_cost=4 seq_page_cost=1
category	Query Tuning / Planner Method Configuration
settings	enable_bitmapscan=on enable_hashagg=on enable_hashjoin=on enable_indexonlyscan=on enable_indexscan=on

File Locations

config_file	C:/projects/pg/pg92edb/data/postgresql.conf
data_directory	C:/projects/pg/pg92edb/data
external_pid_file	
hba_file	C:/projects/pg/pg92edb/data/pg_hba.conf
ident_file	C:/projects/pg/pg92edb/data/pg_ident.conf

Memory Settings

maintenance_work_mem	16384	kB
max_prepared_transactions	0	
max_stack_depth	2048	kB
shared_buffers	4096	8kB
temp_buffers	1024	8kB
track_activity_query_size	1024	
work_mem	1024	kB

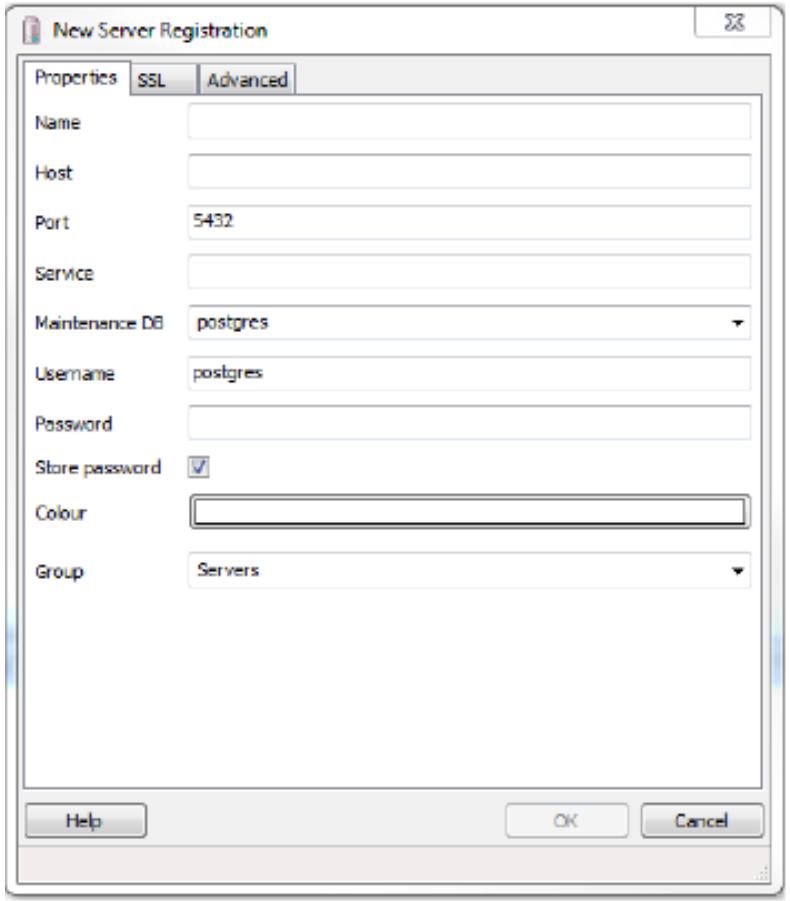
pgAdmin

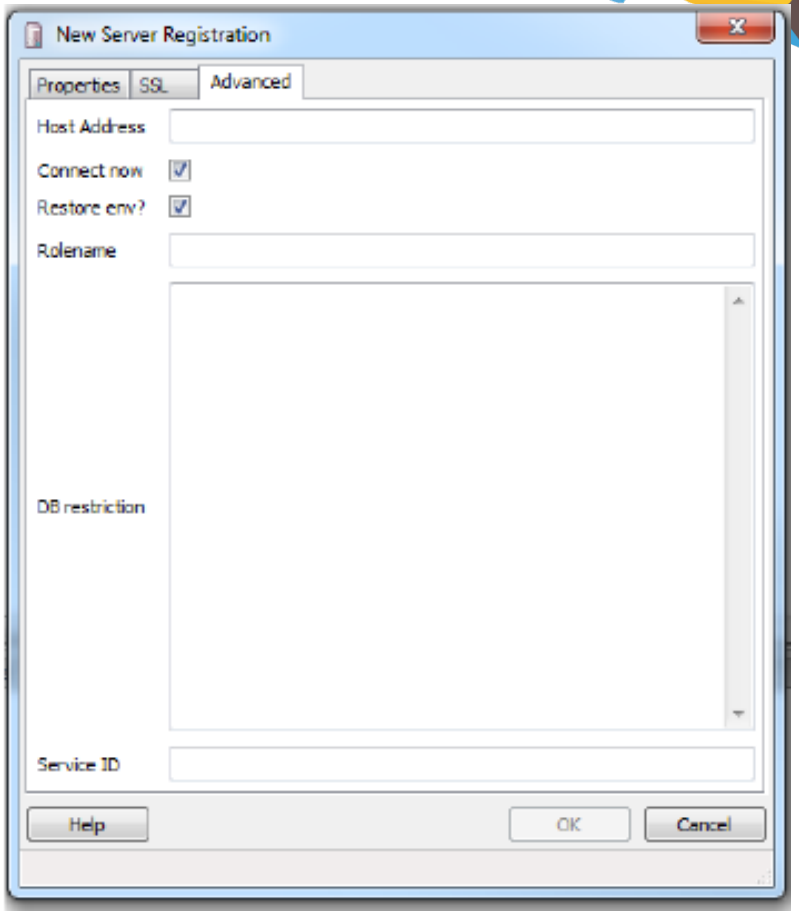
pgAdmin

- **Herramienta gráfica para la administración de PostgreSQL**
- **Habitualmente viene preempaquetada**
- **Carácterísticas**
  - –Explicación gráfica de las consultas
  - –Panel SQL
  - –Editor gráfico para posgresql.conf y pg_hba.conf
  - –Exportación e importación de datos
  - –Wizard de respaldo y restauración
  - –Wizard para otorgar privilegios
  - –pgScript engine
  - –Arquitectura de Plug-in
  - –pgAgent
- **Para habilitar el adminPack en PostgreSQL 9.0 o versiones anteriores, se debe conectar a la base de datos postgres y ejecutar el script share/contrib/adminpack.sql.**
- **Para PostgreSQL 9.1 posterior, conectarse a la base de datos postgres y ejecutar el comando SQL**

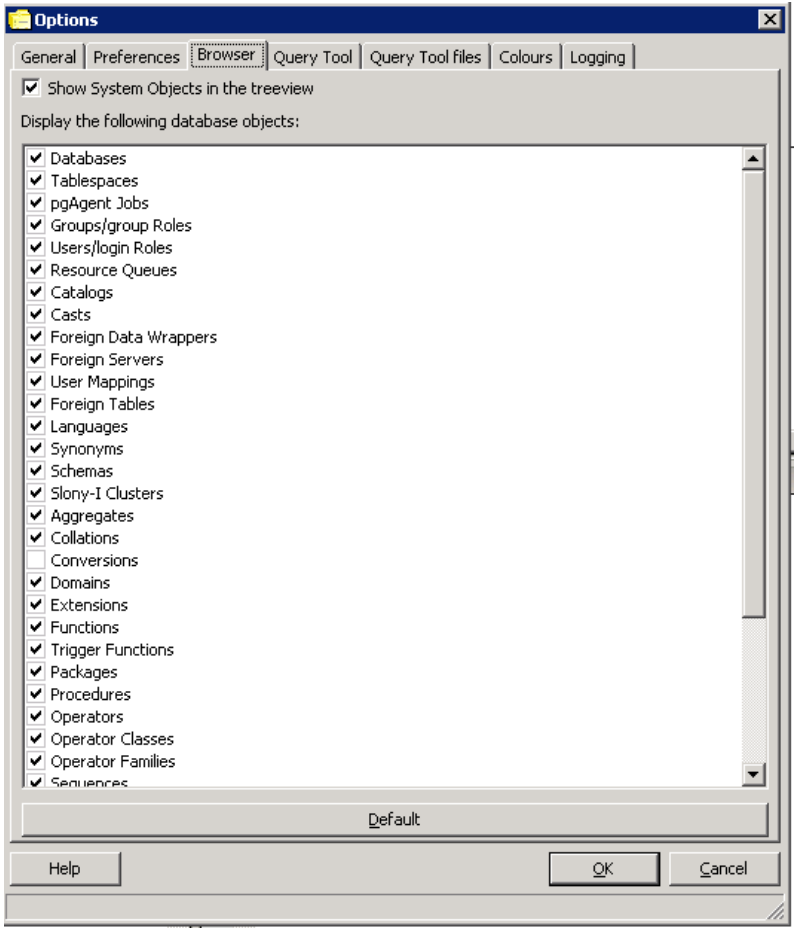
```
CREATE EXTENSION adminpack;
```

Registro de servidor

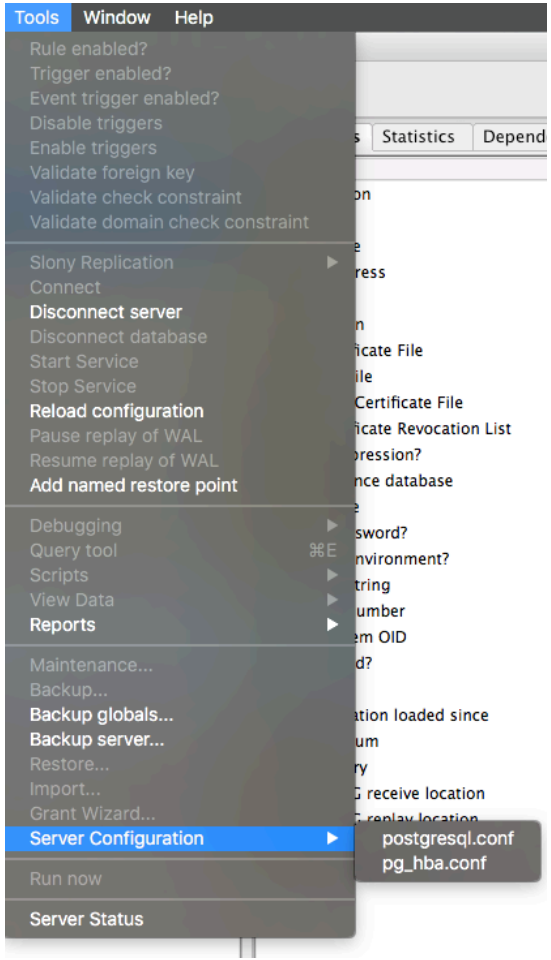


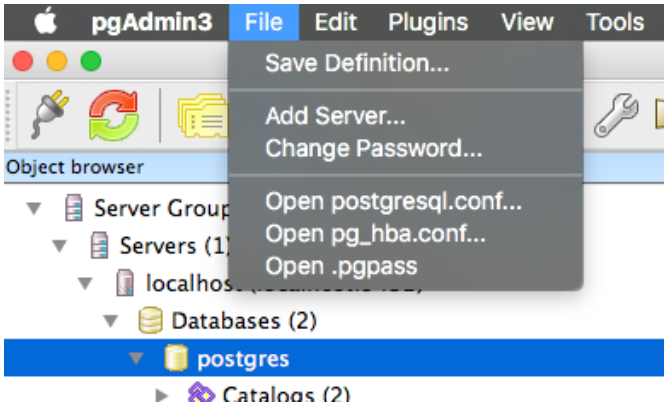


Opciones de visualización

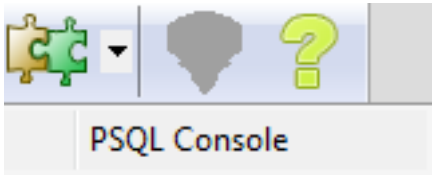


Edición de configuración

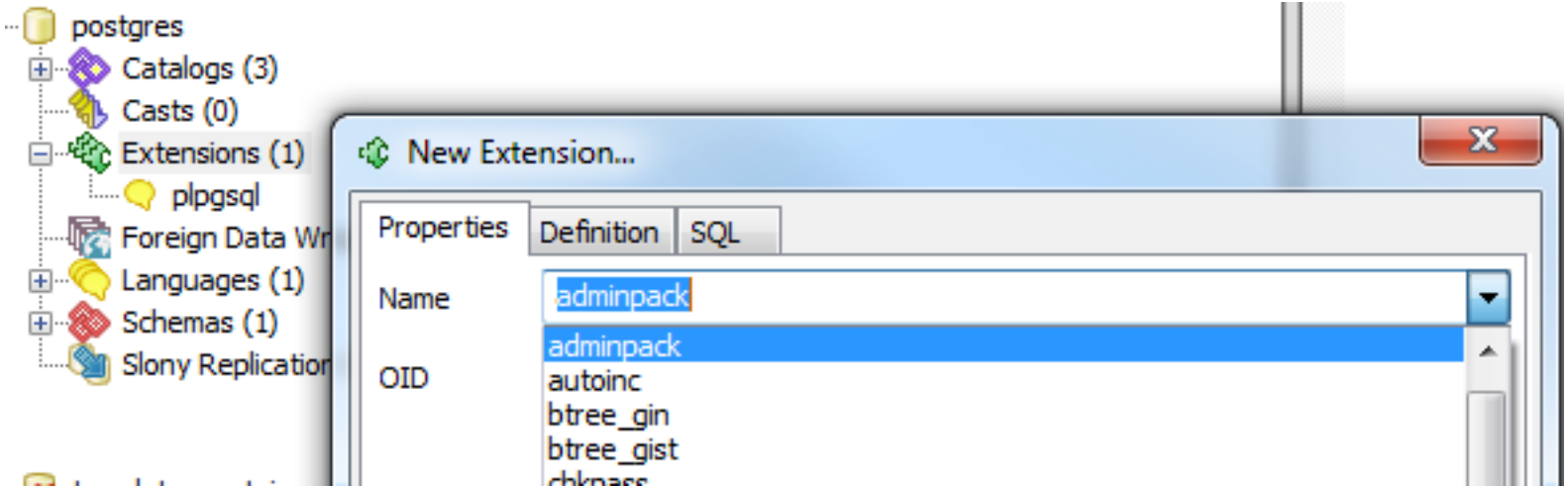




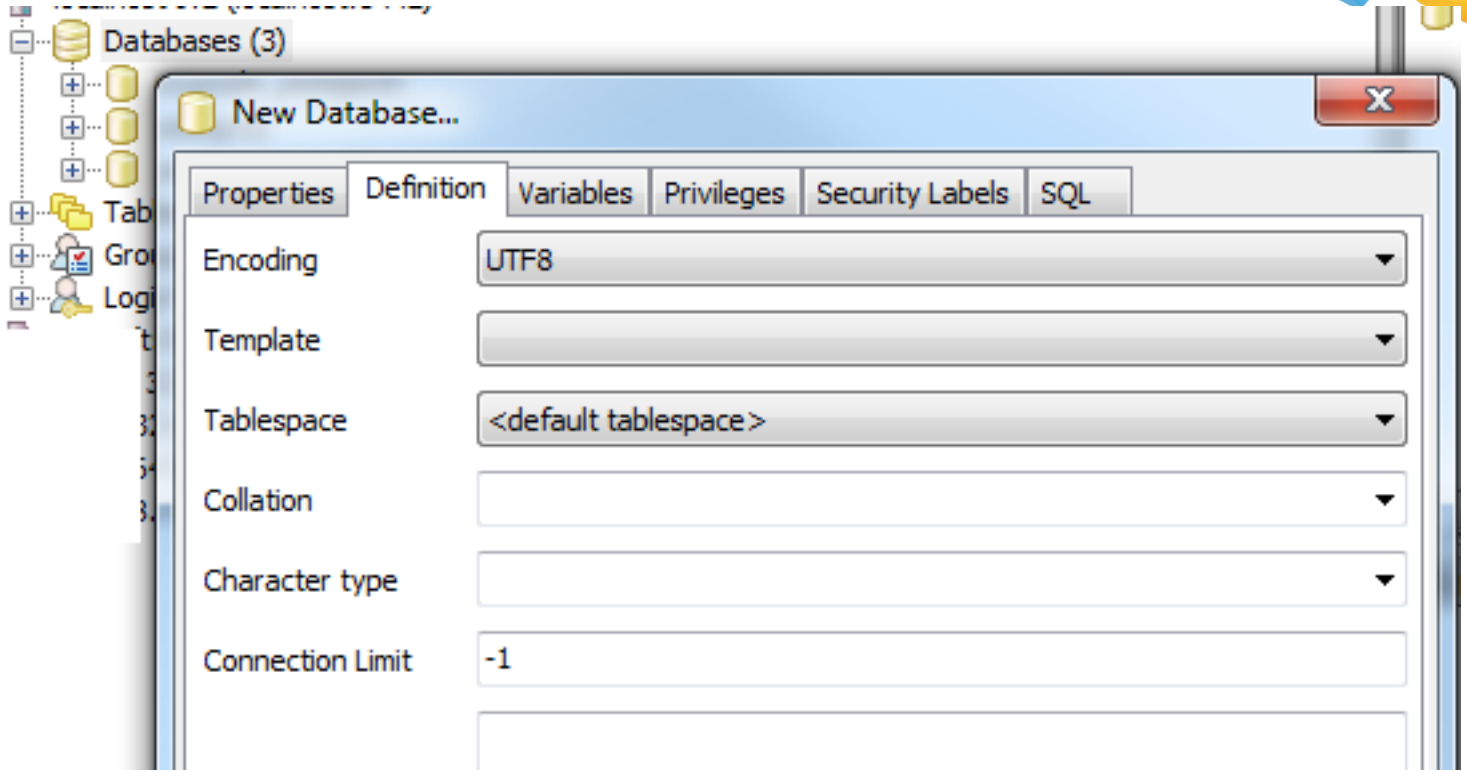
psql



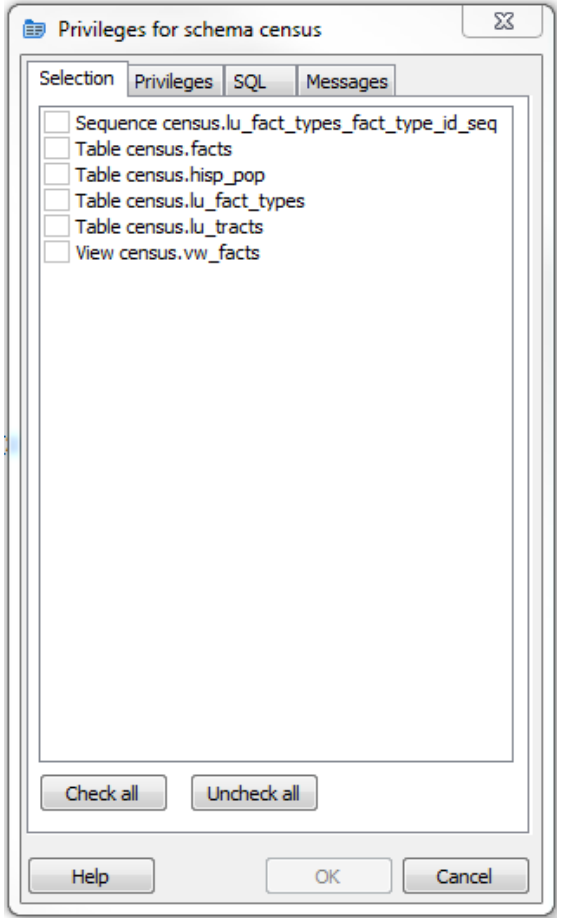
Instalación de extensiones



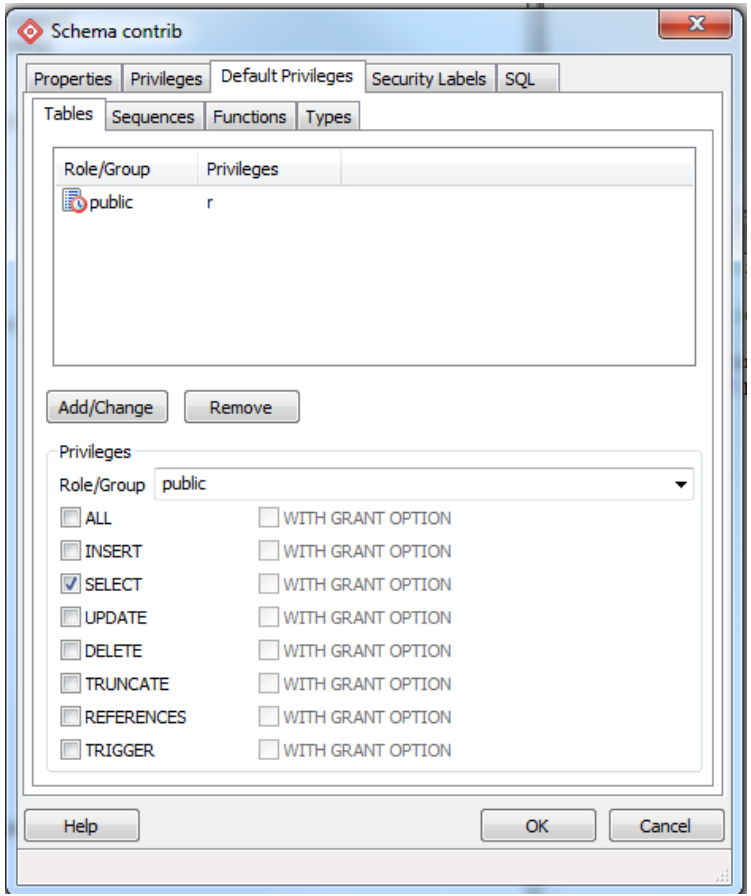
Creación de base de datos



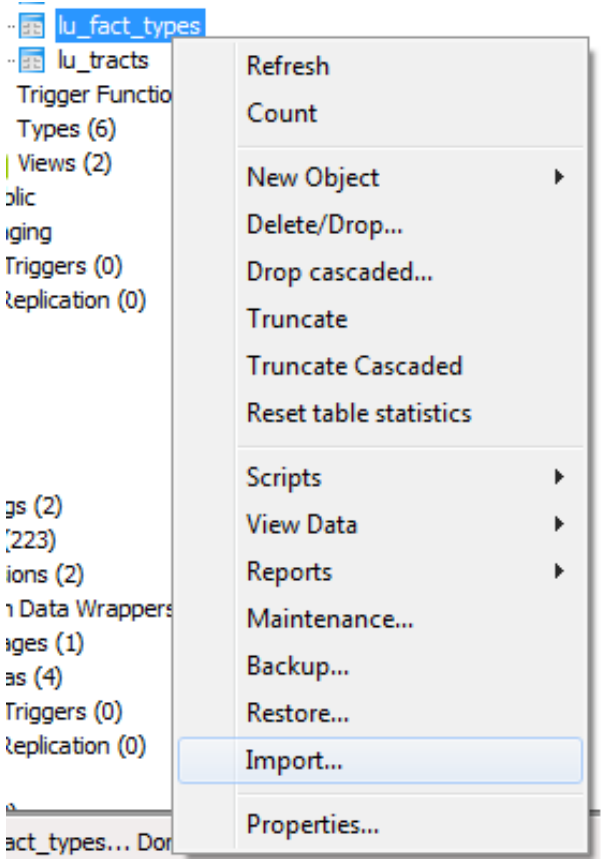
Privilegios

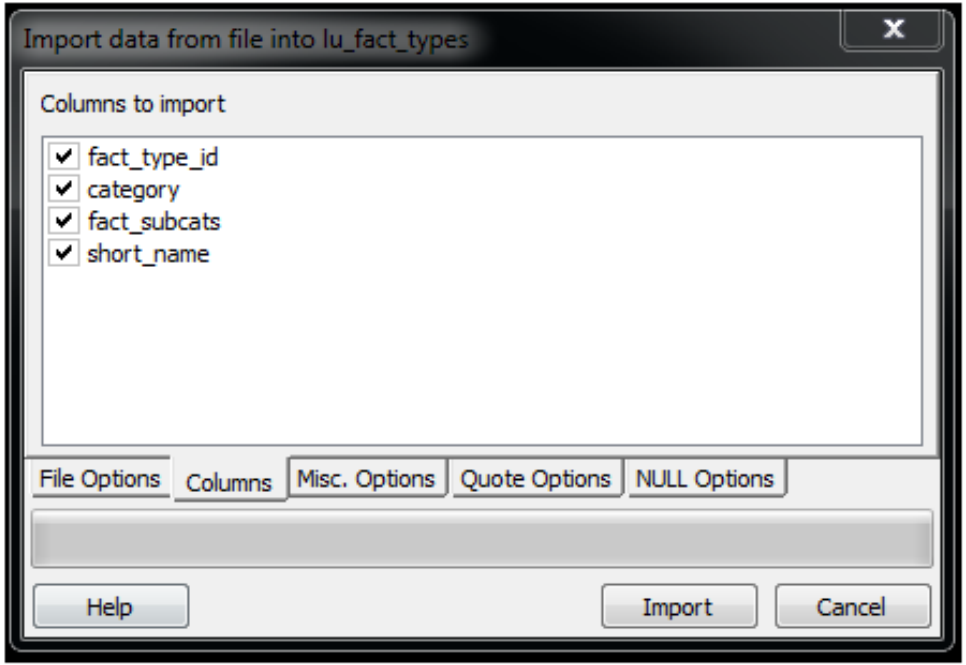


Privilegios



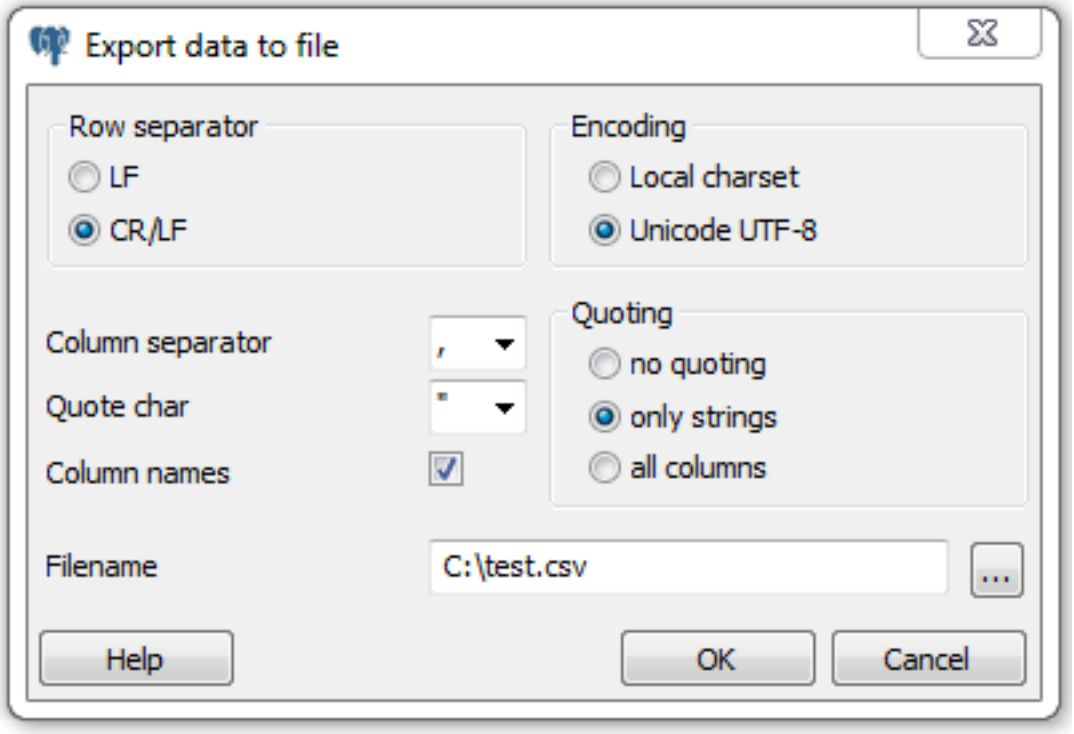
Importación





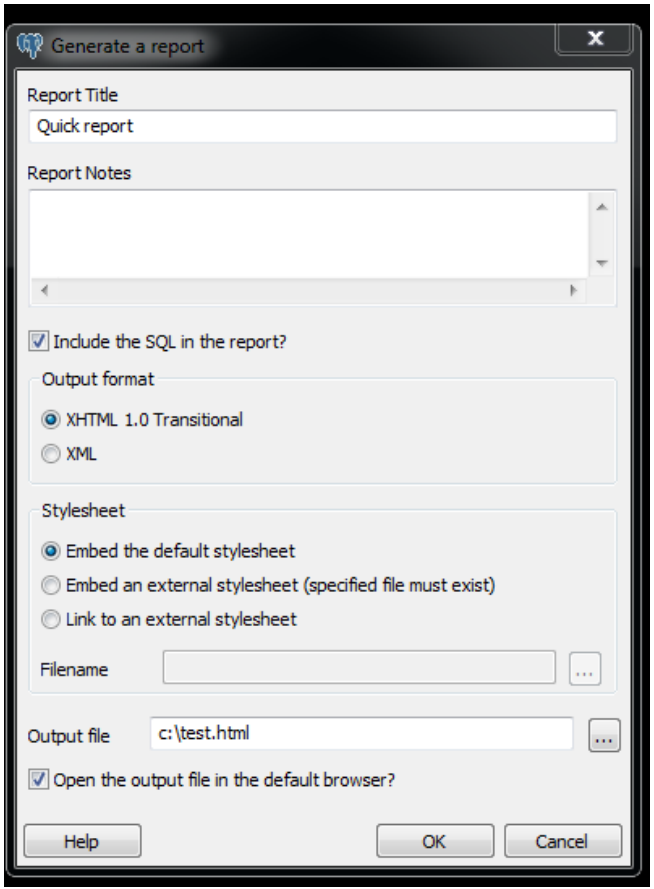
Exportación

- Abrir la ventana query.
- Escribir el query
- Ejecutar
- File→Export.
- Establecer los parámetros

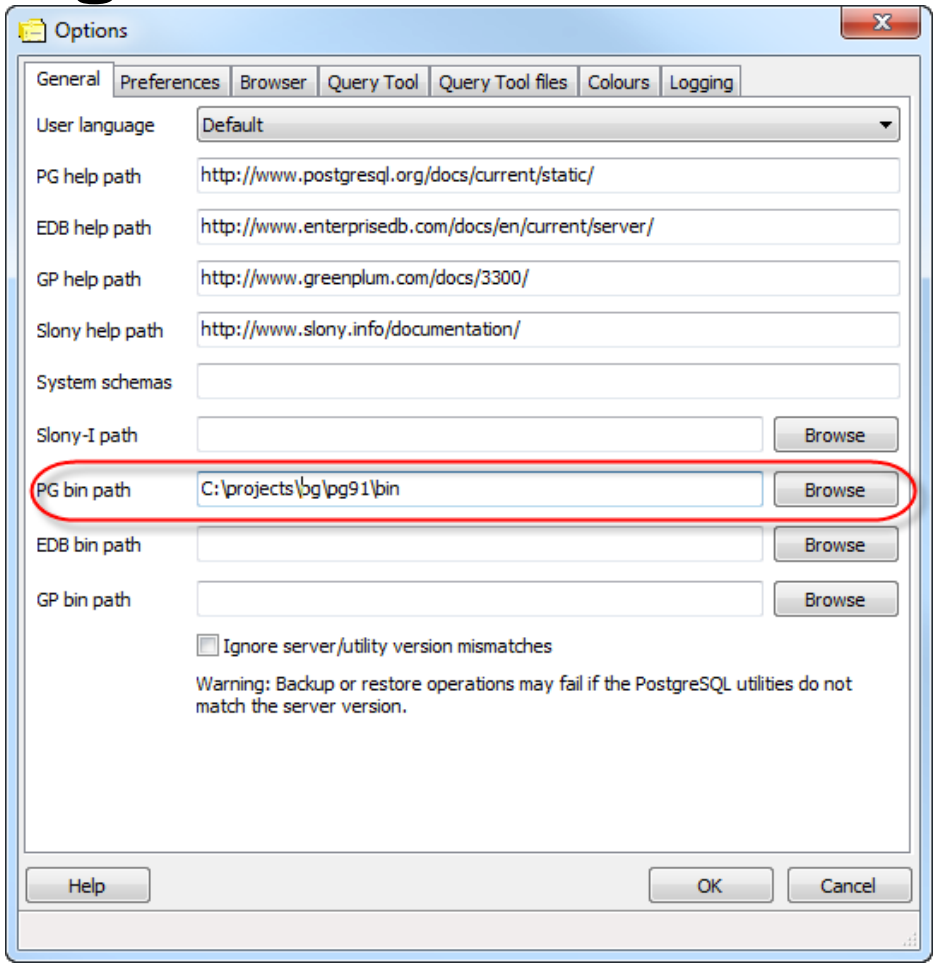


Exportar como HTML y XML

- File->Quick Report

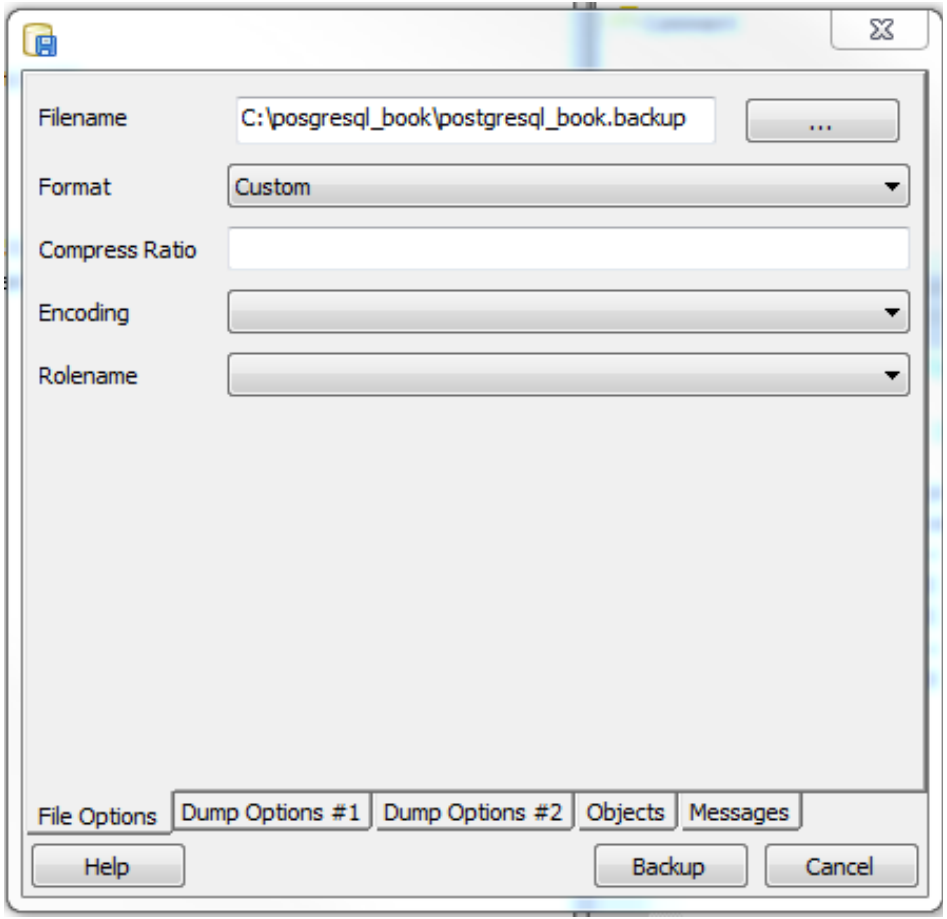


Configuración correcta de binarios



Backup de una base de datos

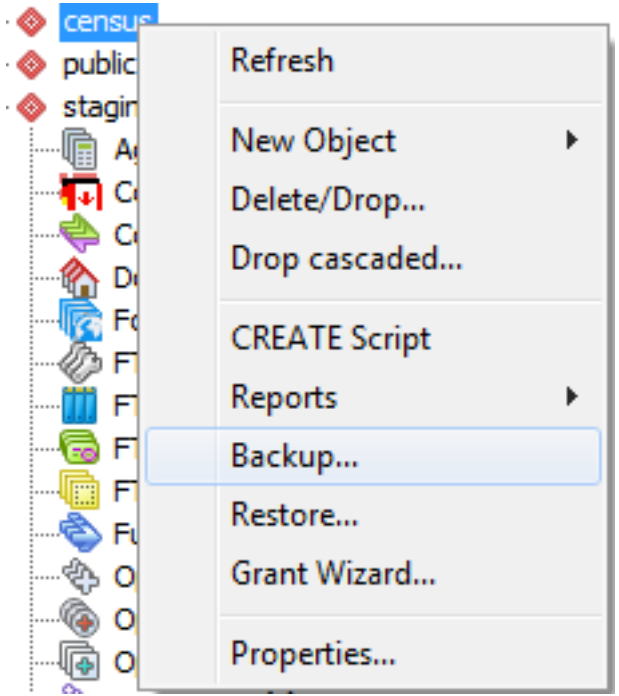
- Botón derecho sobre BD

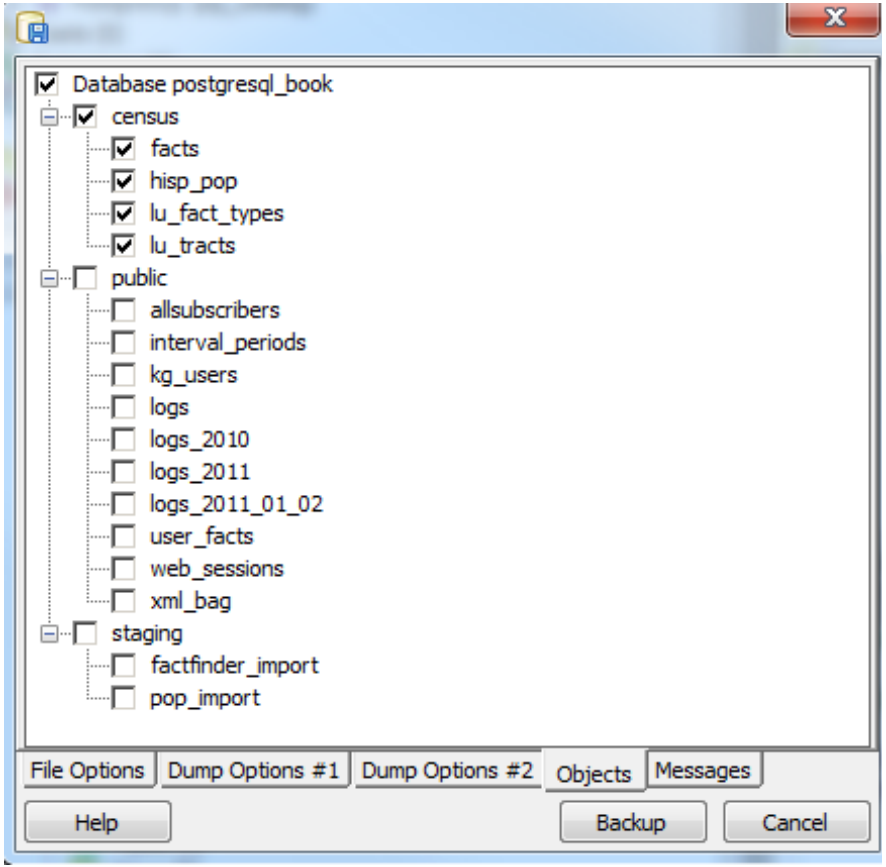


Backup de objetos

- Tools→Backup Globals.
- pgAdmin hace backup de todos los tablespaces y roles.
- Para respaladar todo el servidor (pg_dumpall) Tools→Backup Server.

Backup selectivo





pgScript

```
DECLARE @I, @labels, @tdef;
SET @I = 0;

SET @labels =
SELECT
quote_ident(
replace(
replace(lower(COALESCE(fact_subcats[4], fact_subcats[3])), ' ',
'_'),':',''
)
) As col_name,
fact_type_id
FROM census.lu_fact_types
WHERE category = 'Population' AND fact_subcats[3] ILIKE 'Hispanic or
Latino%'
ORDER BY short_name;
SET @tdef = 'census.hisp_pop(tract_id varchar(11) PRIMARY KEY ';
```

pgScript

```
WHILE @I < LINES(@labels)
BEGIN
SET @tdef = @tdef + ', ' + @labels[@I][0]  + ' numeric(12,3)  ';
SET @I = @I + 1;
END
SET @tdef = @tdef + ')';
Print out table def.
PRINT @tdef;
create the table.
CREATE TABLE @tdef;
```

pgScript

```
DECLARE @I, @labels, @tload, @tcols, @fact_types;
SET @I = 0;
SET @labels =
SELECT
quote_ident(
replace(
replace(
lower(COALESCE(fact_subcats[4], fact_subcats[3])), ' ',
'_'),':',''
)
) As col_name,
fact_type_id
FROM census.lu_fact_types
WHERE category = 'Population' AND fact_subcats[3] ILIKE 'Hispanic or
Latino%'
ORDER BY short_name;
SET @tload = 'tract_id';
SET @tcols = 'tract_id';
SET @fact_types = '-1';
```

pgScript

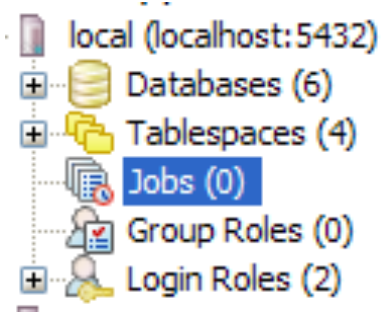
```
WHILE @I < LINES(@labels)
BEGIN
SET @tcols = @tcols + ', ' + @labels[@I][0] ;
SET @tload = @tload +
', MAX(CASE WHEN fact_type_id = ' +
CAST(@labels[@I][1] AS STRING) +
' THEN val ELSE NULL END)';
SET @fact_types = @fact_types + ', ' + CAST(@labels[@I][1] As STRING);
SET @I = @I + 1;
END
INSERT INTO census.hisp_pop(@tcols)
SELECT @tload FROM census.facts
WHERE fact_type_id IN(@fact_types) AND yr=2010
GROUP BY tract_id;
```

Explicación Gráfica

•Se deshabilita si Query->Explain->Buffers está habilitado

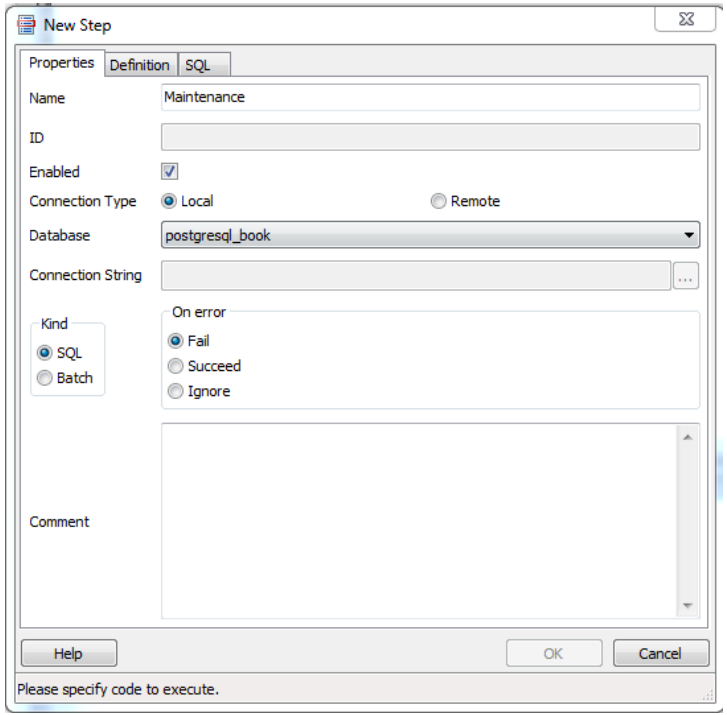
pgAgent

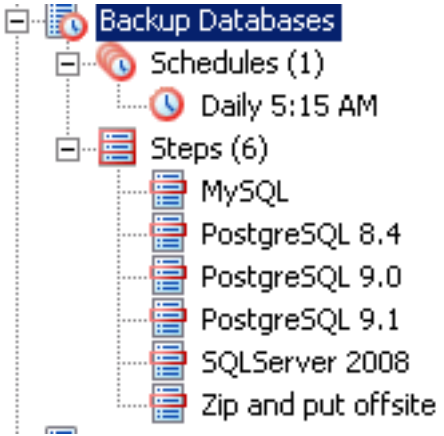
- Descargar pgAgent https://www.pgadmin.org/download/pgagent.php



Creación de un Job

- Pasos
- Agenda de ejecución





Tablas de pgAgent

```
SELECT c.relname As table_name, d.description
FROM
pg_class As c INNER JOIN
pg_namespace n ON n.oid = c.relnamespace INNER JOIN
pg_description As d ON d.objoid = c.oid AND d.objsubid = 0
WHERE n.nspname = 'pgagent'
ORDER BY c.relname;
table_name | description
---------------+-------------------------
pga_job | Job main entry
pga_jobagent | Active job agents
pga_jobclass | Job classification
pga_joblog | Job run logs.
pga_jobstep | Job step to be executed
pga_jobsteplog | Job step run logs.
pga_schedule | Job schedule exceptions
```

Tablas de pgAgent

```
SELECT c.relname As table_name, d.description
FROM
pg_class As c INNER JOIN
pg_namespace n ON n.oid = c.relnamespace INNER JOIN
pg_description As d ON d.objoid = c.oid AND d.objsubid = 0
WHERE n.nspname = 'pgagent'
ORDER BY c.relname;
table_name | description
---------------+-------------------------
pga_job | Job main entry
pga_jobagent | Active job agents
pga_jobclass | Job classification
pga_joblog | Job run logs.
pga_jobstep | Job step to be executed
pga_jobsteplog | Job step run logs.
pga_schedule | Job schedule exceptions
```

Log de los pasos de jobs ejecutados en el día

```
SELECT j.jobname, s.jstname, l.jslstart,l.jslduration, l.jsloutput
FROM
pgagent.pga_jobsteplog As l INNER JOIN
pgagent.pga_jobstep As s ON s.jstid = l.jsljstid INNER JOIN
pgagent.pga_job As j ON j.jobid = s.jstjobid
WHERE jslstart > CURRENT_DATE
ORDER BY j.jobname, s.jstname, l.jslstart DESC;
```