<http://www.cnblogs.com/huajiezh/p/6131244.html>

**一、pg\_dump的用法：**  
数据库的导入导出是最常用的功能之一，每种数据库都提供有这方面的工具，例如Oracle的exp/imp，Informix的dbexp/dbimp，MySQL的mysqldump，而PostgreSQL提供的对应工具为pg\_dump和pg\_restore。  
pg\_dump是用于备份PostgreSQL数据库的工具。它可以在数据库正在使用的时候进行完整一致的备份，并不阻塞其它用户对数据库的访问。  
转储格式可以是一个脚本或者归档文件。转储脚本的格式是纯文本，包含许多SQL命令，这些SQL命令可以用于重建该数据库并将之恢复到保存脚本时的状态。可以使用 psql从这样的脚本中恢复。它们甚至可以用于在其它机器甚至是其它硬件体系的机器上重建数据库，通过对脚本进行一些修改，甚至可以在其它SQL数据库产品上重建数据库。  
归档文件格式必须和pg\_restore一起使用重建数据库。它们允许pg\_restore对恢复什么东西进行选择，甚至是在恢复之前对需要恢复的条目进行重新排序。归档文件也是可以跨平台移植的。  
D:\Program Files\PowerCmd>pg\_dump --help  
pg\_dump 把一个数据库转储为纯文本文件或者是其它格式.  
用法:  pg\_dump [选项]... [数据库名字]  
一般选项:  
  -f, --file=FILENAME         output file or directory name  
  -F, --format=c|d|t|p        output file format (custom, directory, tar, plain text)  
  -v, --verbose            详细模式  
  -Z, --compress=0-9       被压缩格式的压缩级别  
--lock-wait-timeout=TIMEOUT 在等待表锁超时后操作失败  
  --help                       显示此帮助信息, 然后退出  
  --versoin                    输出版本信息, 然后退出  
控制输出内容选项:  
  -a, --data-only          只转储数据,不包括模式  
  -b, --blobs              在转储中包括大对象  
  -c, --clean              在重新创建之前，先清除（删除）数据库对象  
  -C, --create             在转储中包括命令,以便创建数据库  
  -E, --encoding=ENCODING     转储以ENCODING形式编码的数据  
  -n, --schema=SCHEMA      只转储指定名称的模式  
 -N, --exclude-schema=SCHEMA     不转储已命名的模式  
  -o, --oids               在转储中包括 OID  
  -O, --no-owner           在明文格式中, 忽略恢复对象所属者  
  -s, --schema-only        只转储模式, 不包括数据  
  -S, --superuser=NAME     在转储中, 指定的超级用户名  
  -t, --table=TABLE        只转储指定名称的表  
  -T, --exclude-table=TABLE       只转储指定名称的表  
  -x, --no-privileges      不要转储权限 (grant/revoke)  
  --binary-upgrade         只能由升级工具使用  
  --column-inserts          以带有列名的INSERT命令形式转储数据  
  --disable-dollar-quoting     取消美元 (符号) 引号, 使用 SQL 标准引号  
  --disable-triggers         在只恢复数据的过程中禁用触发器  
  --inserts                 以INSERT命令，而不是COPY命令的形式转储数据  
  --no-security-labels        do not dump security label assignments  
  --no-tablespaces           不转储表空间分配信息  
  --no-unlogged-table-data    do not dump unlogged table data  
  --quote-all-identifiers     quote all identifiers, even if not key words  
  --serializable-deferrable   wait until the dump can run without anomalies  
 --use-set-session-authorization  
   使用 SESSION AUTHORIZATION 命令代替ALTER OWNER 命令来设置所有权  
联接选项:  
  -h, --host=主机名        数据库服务器的主机名或套接字目录  
  -p, --port=端口号        数据库服务器的端口号  
  -U, --username=名字      以指定的数据库用户联接  
  -w, --no-password        永远不提示输入口令  
  -W, --password           强制口令提示 (自动)  
  --role=ROLENAME          do SET ROLE before dump  
如果没有提供数据库名字, 那么使用 PGDATABASE 环境变量的数值.

**二、pg\_dump的使用实例**  
1、创建两个数据库  
CREATE DATABASE "TestDb1"  
  WITH OWNER = "TestRole1"  
       ENCODING = 'UTF8'  
       TABLESPACE = "TestTbs1";  
CREATE DATABASE "TestDb2"  
  WITH OWNER = "TestRole1"  
       ENCODING = 'UTF8'  
       TABLESPACE = "TestTbs1";  
在TestDb1中创建表csm\_bill、cfg\_public\_int\_transport插入几条记录，并创建索引，索引使用索引表空间TestTbsIndex。

2、仅迁移数据库结构：  
E:\>pg\_dump -U TestRole1 -s -f TestDb1.sql TestDb1  
口令:  
-U TestRole1和超级用户-U postgres结果完全相同：  
E:\>pg\_dump -U postgres -s -f TestDb11.sql TestDb1  
E:\>psql -U TestRole2 -f TestDb1.sql TestDb2 >a.txt 2>&1  
用户 TestRole2 的口令：  
导入时，使用-U TestRole2往往有很多权限不够，要想成功导入需要修改相关数据库对象的属主，所以最好使用超级用户-U postgres：  
E:\>psql -U postgres -f TestDb1.sql TestDb2 >a.txt 2>&1  
不转储权限选项：-x  
E:\>pg\_dump -U postgres -x -s -f TestDb12.sql TestDb1  
TestDb12.sql比TestDb1.sql少了一下几行：

为了可以多次运行TestDb1.sql，可以在文件开始加以下两行：  
drop schema public cascade;  
create schema public;  
或者使用-c选项：  
E:\>pg\_dump -U postgres -c -x -s -f TestDb13.sql TestDb1  
TestDb13.sql比TestDb1.sql多以下几行：

此时，可以多次运行：  
E:\>psql -U postgres -f TestDb13.sql TestDb2 >a.txt 2>&1  
但是，如果两个库有不同的表或索引，应该使用第一种方法，因为第二种方法在找不到某些数据库对象时会报错。

3、迁移数据库结构和数据（可以实现数据库的备份与恢复）  
数据的复制使用copy命令：  
E:\>pg\_dump -U postgres TestDb1>TestDb14.sql

数据的复制使用insert语句：  
E:\>pg\_dump -U postgres --column-inserts TestDb1>TestDb15.sql

4、把远程linux上PostgreSQL上的cpost数据库结构迁移至本地PostgreSQL  
（1）在本地建一个完全相同的环境  
create user "cpost" inherit createdb;  
create tablespace "pis\_data" owner cpost location 'E:\PostgreSQL/data/pis\_data';  
create tablespace "pis\_index" owner cpost location 'E:\PostgreSQL/data/pis\_index';  
远程数据库cpost仍使用了默认表空间：  
CREATE DATABASE cpost  
  WITH OWNER = cpost  
       --ENCODING = 'LATIN9'  
       TABLESPACE = pg\_default  
       --LC\_COLLATE = 'C'  
       --LC\_CTYPE = 'C'  
       CONNECTION LIMIT = -1;  
使用以上三个参数报错，建成后的数据库如下：  
CREATE DATABASE cpost  
  WITH OWNER = cpost  
       ENCODING = 'UTF8'  
       TABLESPACE = pg\_default  
       LC\_COLLATE = 'Chinese (Simplified)\_People''s Republic of China.936'  
       LC\_CTYPE = 'Chinese (Simplified)\_People''s Republic of China.936'  
       CONNECTION LIMIT = -1;  
（2）使用pg\_dump迁移表结构  
使用-h选项，使导出的sql文件直接存放在本地：  
E:\>pg\_dump -h 132.10.10.11 -p 1234 -U cpost -x -s -f cpost.sql cpost  
E:\>psql -U postgres -f cpost.sql  
导入成功，但报了一个错误：  
psql:cpost.sql:22: ERROR:  character 0xe99499 of encoding "UTF8" has no equivalent in "LATIN9"  
字符集错误，字符集问题详见我的另一篇blog：由PostgreSQL的区域与字符集说起

**三、使用pg\_dump和pg\_restore实现数据库的备份与恢复**  
E:\>pg\_restore --help  
pg\_restore 从一个归档中恢复一个由 pg\_dump 创建的 PostgreSQL 数据库.  
用法:  
  pg\_restore [选项]... [文件名]  
一般选项:  
  -d, --dbname=名字        连接数据库名字  
  -f, --file=文件名        输出文件名  
  -F, --format=c|d|t       backup file format (should be automatic)  
  -l, --list               打印归档文件的 TOC 概述  
  -v, --verbose            详细模式  
  --help                   显示此帮助信息, 然后退出  
  --version                输出版本信息, 然后退出恢复控制选项:  
  -a, --data-only          只恢复数据, 不包括模式  
  -c, --clean              在重新创建数据库对象之前需要清除（删除）数据库对象  
  -C, --create             创建目标数据库  
  -e, --exit-on-error      发生错误退出, 默认为继续  
  -I, --index=名称         恢复指定名称的索引  
  -j, --jobs=NUM           可以执行多个任务并行进行恢复工作  
  -L, --use-list=文件名    从这个文件中使用指定的内容表排序输出  
  -n, --schema=NAME      在这个模式中只恢复对象  
  -O, --no-owner           忽略恢复对象所属者  
  -P, --function=名字(参数)  恢复指定名字的函数  
  -s, --schema-only        只恢复模式, 不包括数据  
  -S, --superuser=NAME     使用指定的超级用户来禁用触发器  
  -t, --table=NAME         恢复指定命字的表  
  -T, --trigger=NAME       恢复指定命字的触发器  
  -x, --no-privileges      跳过处理权限的恢复 (grant/revoke)  
  -1, --single-transaction   作为单个事务恢复  
 --disable-triggers        在只恢复数据的过程中禁用触发器  
  --no-data-for-failed-tables 没有恢复无法创建表的数据  
  --no-security-labels     do not restore security labels  
--no-tablespaces          不恢复表空间的分配信息  
  --use-set-session-authorization 使用 SESSION AUTHORIZATION 命令代替ALTER OWNER命令来设置对象所有权  
联接选项:  
  -h, --host=主机名        数据库服务器的主机名或套接字目录  
  -p, --port=端口号        数据库服务器的端口号  
  -U, --username=名字      以指定的数据库用户联接  
  -w, --no-password        永远不提示输入口令  
  -W, --password           强制口令提示 (自动)  
  --role=ROLENAME          在恢复前执行SET ROLE操作  
如果没有提供输入文件名, 则使用标准输入.  
1、使用dump格式备份和恢复：  
E:\>pg\_dump -U postgres -Fc TestDb1 >TestDb1.dump  
postgres=# drop database "TestDb2";  
DROP DATABASE  
postgres=# create database "TestDb2"  
postgres-# with owner="TestRole2"  
postgres-# tablespace="TestTbs2";  
CREATE DATABASE  
E:\>pg\_restore -U postgres -d TestDb2 TestDb1.dump >a.txt 2>&1  
2、使用tar格式备份和恢复：  
E:\>pg\_dump -U postgres -Ft TestDb1>TestDb1.tar  
postgres=# drop database "TestDb2";  
DROP DATABASE  
postgres=# create database "TestDb2"  
postgres-# with owner="TestRole2"  
postgres-# tablespace="TestTbs2";  
CREATE DATABASE  
E:\>pg\_restore -U postgres -d TestDb2 TestDb1.tar >a.txt 2>&1