迁移步骤

1. **系统信息**
2. 服务器列表

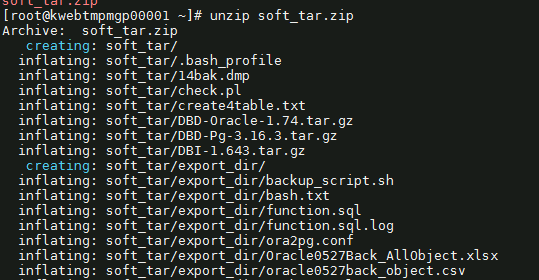
|  |  |  |  |
| --- | --- | --- | --- |
| 机器 | IP | OS系统 | 实例规格 |
| Postgres14.8 | 10.28.160.242 | CentOS release 6.10(Final) | 88U755G |

1. **系统环境安装准备**
2. 安装pg14.8
   1. 上传文件



* 1. 解压文件

unzip soft\_tar.zip

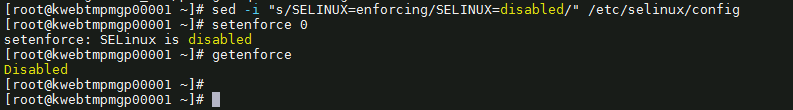


* 1. 禁用SELinux强制模式

sed -i "s/SELINUX=enforcing/SELINUX=disabled/" /etc/selinux/config

setenforce 0

getenforce



* 1. 关闭防火墙

service iptables stop

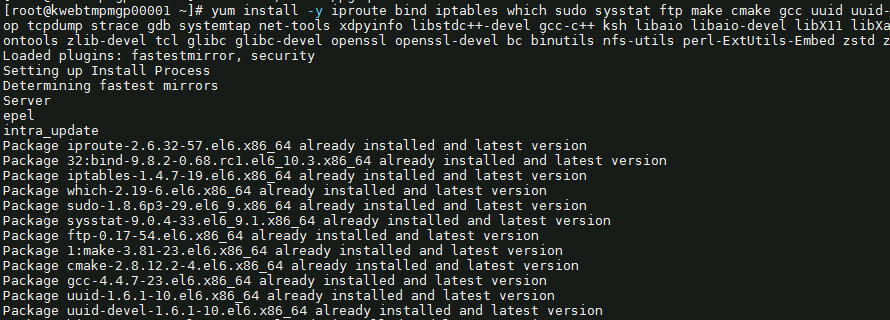
* 1. 设置变量

PGDATA=’/data01/pgsql’



* 1. 安装依赖包

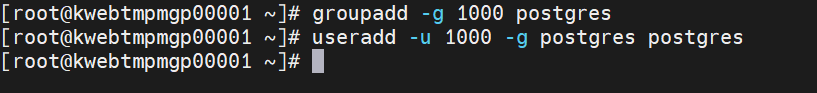
yum install -y iproute bind iptables which sudo sysstat ftp make cmake gcc uuid uuid-devel bison flex perl perl-devel python-devel readline readline-devel libxml2 libxml2-devel iotop tcpdump strace gdb systemtap net-tools xdpyinfo libstdc++-devel gcc-c++ ksh libaio libaio-devel libX11 libXau libXi libXtst libXrender libXrender-devel libgcc libstdc++ libstdc++-devel libxcb make smartmontools zlib-devel tcl glibc glibc-devel openssl openssl-devel bc binutils nfs-utils perl-ExtUtils-Embed zstd zstd-devel libcurl libcurl-devel lz4 lz4-devel libicu libicu-devel autoconf



* 1. 创建用户和组

groupadd -g 1000 postgres

useradd -u 1000 -g postgres postgres



* 1. 创建数据库目录

mkdir -p $PGDATA



* 1. 需要先上传到root

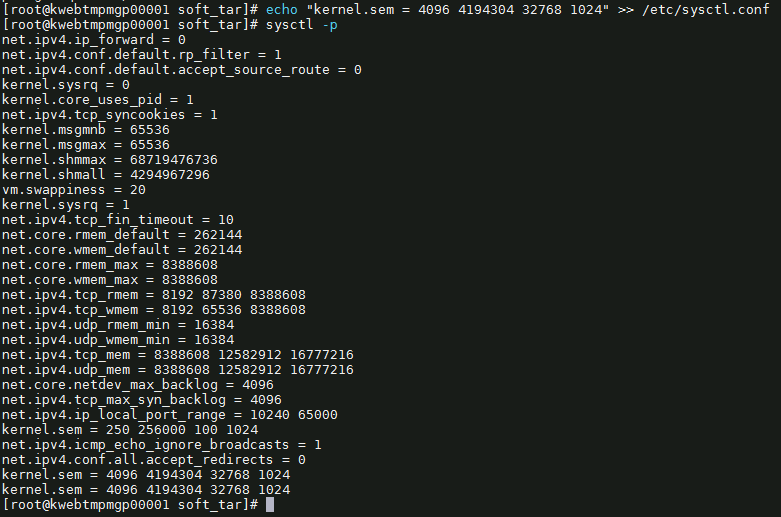
cd soft\_tar

tar -zxvf postgresql-14.8.tar.gz -C /data01/app/install 

* 1. 修改服务器内存和信号量

echo "kernel.sem = 4096 4194304 32768 1024" >> /etc/sysctl.conf

sysctl -p



* 1. 修改服务器资源限制

echo "postgres soft nproc unlimited" >> /etc/security/limits.conf

echo "postgres hard nproc unlimited" >> /etc/security/limits.conf

echo "postgres soft nofile 1024000" >> /etc/security/limits.conf

echo "postgres hard nofile 1024000" >> /etc/security/limits.conf

echo "postgres soft stack unlimited" >> /etc/security/limits.conf

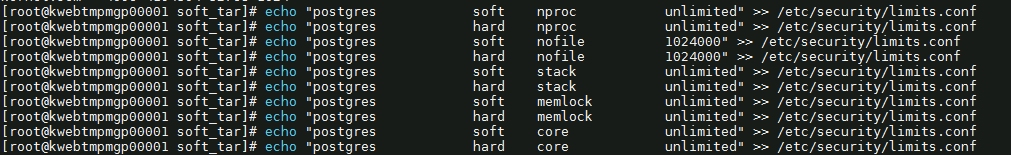
echo "postgres hard stack unlimited" >> /etc/security/limits.conf

echo "postgres soft memlock unlimited" >> /etc/security/limits.conf

echo "postgres hard memlock unlimited" >> /etc/security/limits.conf

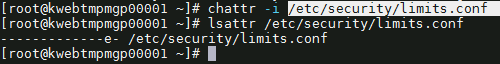
echo "postgres soft core unlimited" >> /etc/security/limits.conf

echo "postgres hard core unlimited" >> /etc/security/limits.conf



报错没有权限写入

chattr -i /etc/security/limits.conf

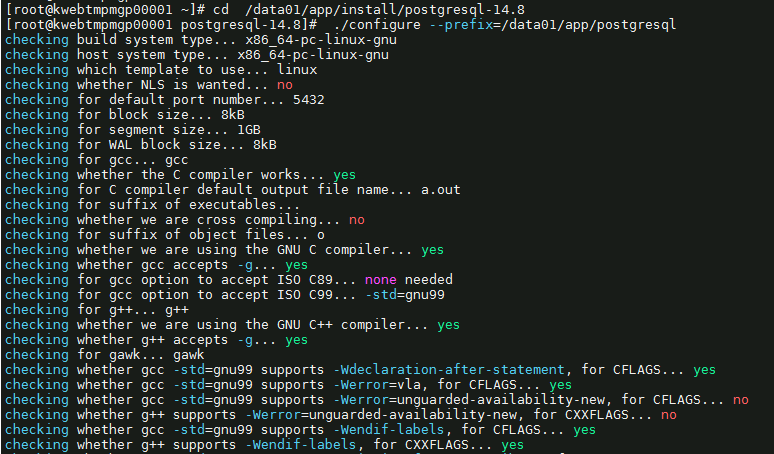


* 1. 编译包

cd /data01/app/install/postgresql-14.8

./configure --prefix=/data01/app/postgresql

make world && make install-world



* 1. 添加配置文件

su - postgres -c "echo 'export PG\_HOME=/data01/app/postgresql' >> /home/postgres/.bash\_profile"

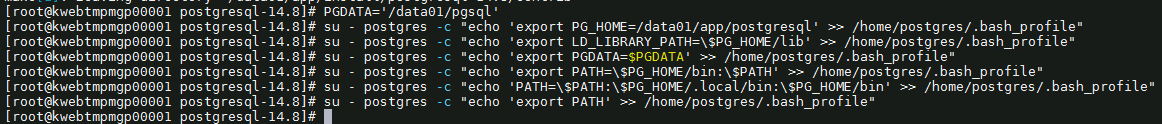
su - postgres -c "echo 'export LD\_LIBRARY\_PATH=\$PG\_HOME/lib' >> /home/postgres/.bash\_profile"

su - postgres -c "echo 'export PGDATA=$PGDATA' >> /home/postgres/.bash\_profile"

su - postgres -c "echo 'export PATH=\$PG\_HOME/bin:\$PATH' >> /home/postgres/.bash\_profile"

su - postgres -c "echo 'PATH=\$PATH:\$PG\_HOME/.local/bin:\$PG\_HOME/bin' >> /home/postgres/.bash\_profile"

su - postgres -c "echo 'export PATH' >> /home/postgres/.bash\_profile"



* 1. 初始化数据库

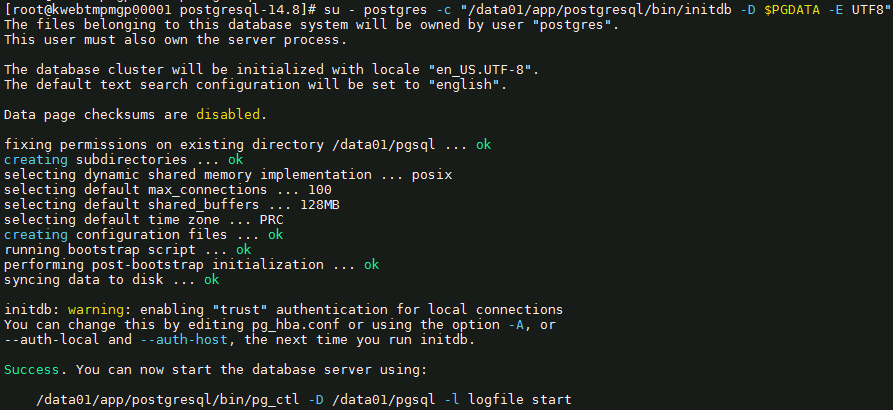
chown -R postgres:postgres /data01/app/install

chown -R postgres:postgres /data01/app/postgresql

chown -R postgres:postgres $PGDATA



su - postgres -c "/data01/app/postgresql/bin/initdb -D $PGDATA -E UTF8"



* 1. 创建归档日志目录

su - postgres -c "mkdir -p $PGDATA/archivedir"



* 1. 修改pg\_hba.conf

su - postgres -c "echo 'host all all 0/0 md5' >> $PGDATA/pg\_hba.conf"

su - postgres -c "echo 'host replication replica 0/0 md5' >> $PGDATA/pg\_hba.conf"



* 1. 配置postgresql.conf

if [ $(lscpu |grep '^CPU(s): ' | awk -F " " '{print $1}') == 'CPU(s):' ]

then

CPU=$(lscpu |grep '^CPU(s): ' | awk -F " " '{print $2}')

elif [ $(lscpu |grep '^CPU: ' | awk -F " " '{print $1}') == 'CPU:' ]

then

CPU=$(lscpu |grep '^CPU: ' | awk -F " " '{print $2}')

else

echo "没有符合的条件"

fi

MEM\_S=$(free -m|grep '^Mem:' | awk -F " " '{print expr $2/1024\*0.4}' | cut -d '.' -f1)GB

MEM\_E=$(free -m|grep '^Mem:' | awk -F " " '{print $2/1024\*0.5}' | cut -d '.' -f1)GB

sed -i "s/#listen\_addresses = 'localhost'/listen\_addresses = '\*'/" $PGDATA/postgresql.conf

sed -i "s/#port = 5432/port = 5432/" $PGDATA/postgresql.conf

sed -i "s/max\_connections = 100/max\_connections = 3000/" $PGDATA/postgresql.conf

# 开启大页，这个项也要开启

# sed -i "s/#huge\_pages = try/huge\_pages = on/" $PGDATA/postgresql.conf

sed -i "s/shared\_buffers = 128MB/shared\_buffers = $MEM\_S/" $PGDATA/postgresql.conf

sed -i "s/#work\_mem = 4MB/work\_mem = 16MB/" $PGDATA/postgresql.conf

sed -i "s/#wal\_buffers = -1/wal\_buffers = 16MB/" $PGDATA/postgresql.conf

sed -i "s/#checkpoint\_completion\_target = 0.9/checkpoint\_completion\_target = 0.9/" $PGDATA/postgresql.conf

sed -i "s/max\_wal\_size = 1GB/max\_wal\_size = 8GB/" $PGDATA/postgresql.conf

sed -i "s/min\_wal\_size = 80MB/min\_wal\_size = 2GB/" $PGDATA/postgresql.conf

sed -i "s/#archive\_mode = off/archive\_mode = on/" $PGDATA/postgresql.conf

sed -i "s/#archive\_command = ''/archive\_command = 'test ! -f \/data01\/pgsql\/archivedir\/%f \&\& cp %p \/data01\/pgsql\/archivedir\/%f'/" $PGDATA/postgresql.conf

sed -i "s/#default\_statistics\_target = 100/default\_statistics\_target = 100/" $PGDATA/postgresql.conf

sed -i "s/#log\_destination = 'stderr'/log\_destination = 'csvlog'/" $PGDATA/postgresql.conf

sed -i "s/#logging\_collector = off/logging\_collector = on/" $PGDATA/postgresql.conf

sed -i "s/#effective\_cache\_size = 4GB/effective\_cache\_size = $MEM\_E/" $PGDATA/postgresql.conf

sed -i "s/#random\_page\_cost = 4.0/random\_page\_cost = 1.1/" $PGDATA/postgresql.conf

sed -i "s/#maintenance\_io\_concurrency = 10/maintenance\_io\_concurrency = 200/" $PGDATA/postgresql.conf

sed -i "s/#max\_worker\_processes = 8/max\_worker\_processes = $CPU/" $PGDATA/postgresql.conf

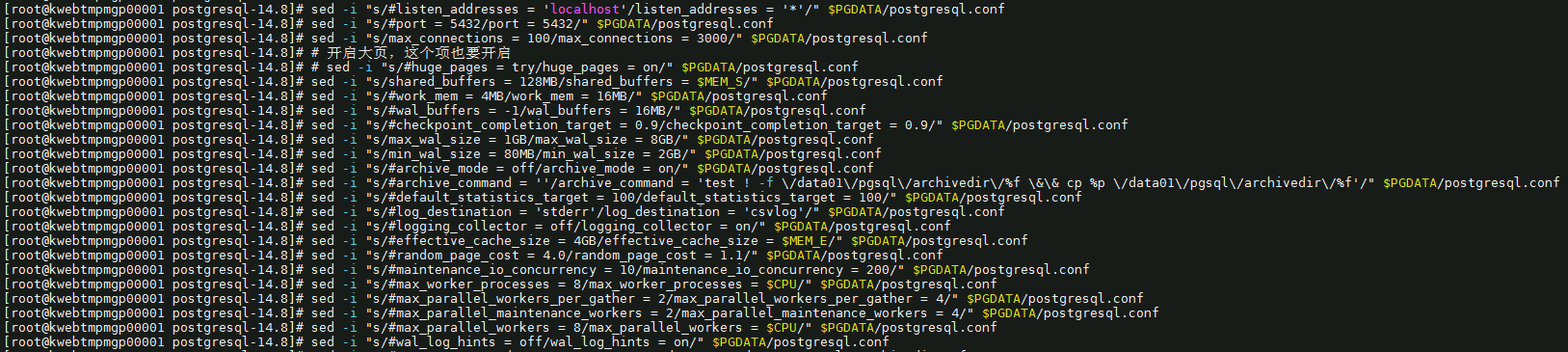
ccccsed -i "s/#max\_parallel\_workers\_per\_gather = 2/max\_parallel\_workers\_per\_gather = 4/" $PGDATA/postgresql.conf

sed -i "s/#max\_parallel\_maintenance\_workers = 2/max\_parallel\_maintenance\_workers = 4/" $PGDATA/postgresql.conf

sed -i "s/#max\_parallel\_workers = 8/max\_parallel\_workers = $CPU/" $PGDATA/postgresql.conf

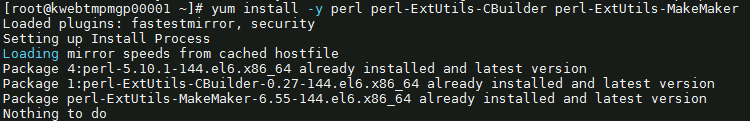
sed -i "s/#wal\_log\_hints = off/wal\_log\_hints = on/" $PGDATA/postgresql.conf

sed -i "s/#restore\_command = ''/restore\_command = 'cp \/data01\/pgsql\/archivedir\/%f %p'/" $PGDATA/ postgresql.conf

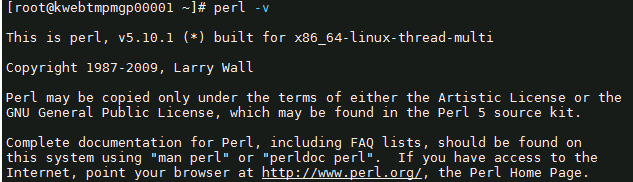


1. 安装ora2pg
   * 1. 安装perl

yum install -y perl perl-ExtUtils-CBuilder perl-ExtUtils-MakeMaker



perl -v



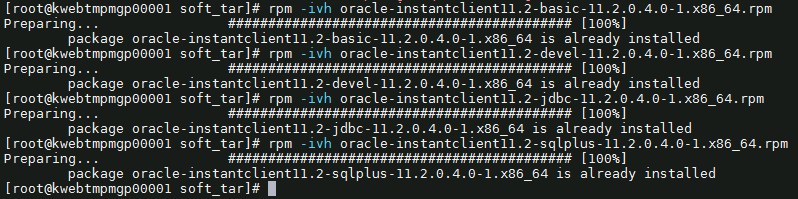
* + 1. 安装oracle客户端

rpm -ivh oracle-instantclient11.2-basic-11.2.0.4.0-1.x86\_64.rpm

rpm -ivh oracle-instantclient11.2-devel-11.2.0.4.0-1.x86\_64.rpm

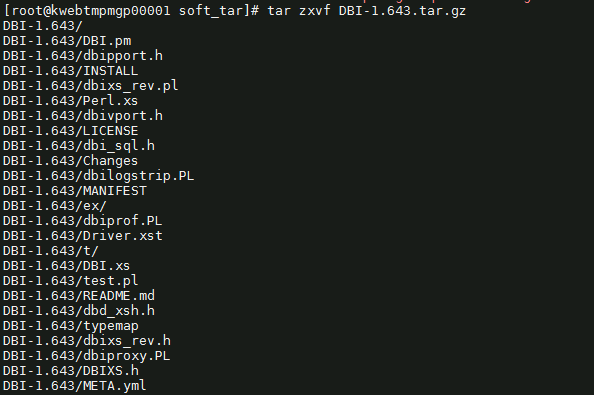
rpm -ivh oracle-instantclient11.2-jdbc-11.2.0.4.0-1.x86\_64.rpm

rpm -ivh oracle-instantclient11.2-sqlplus-11.2.0.4.0-1.x86\_64.rpm



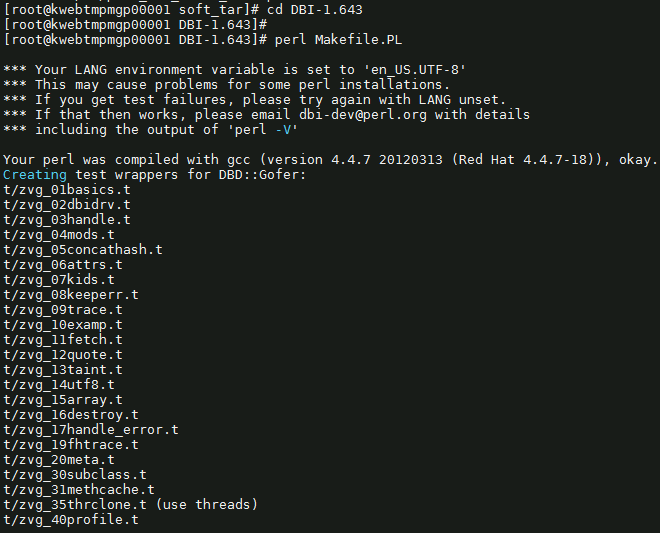
* + 1. 安装DBI模块

tar zxvf DBI-1.643.tar.gz

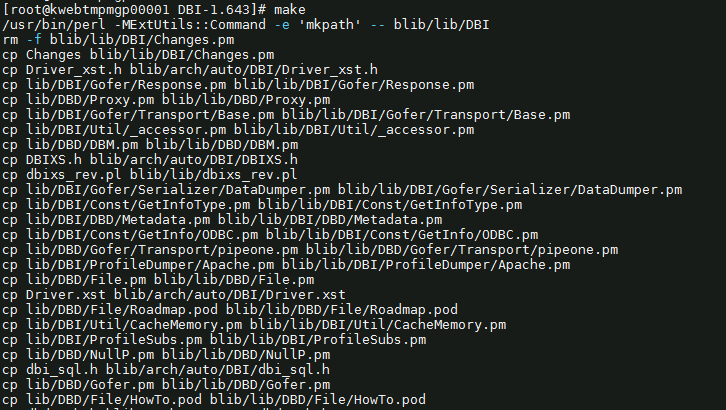


cd DBI-1.643

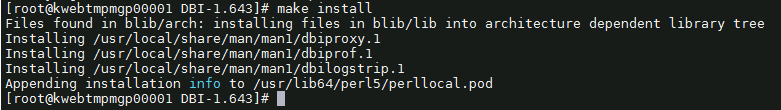
perl Makefile.PL



make



make install

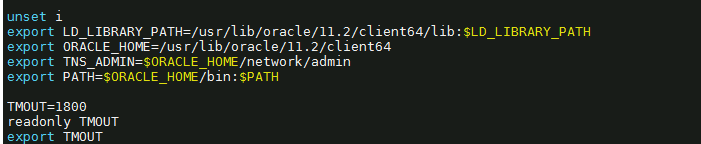


* + 1. 安装DBD::oracle驱动模块

vi /etc/profile

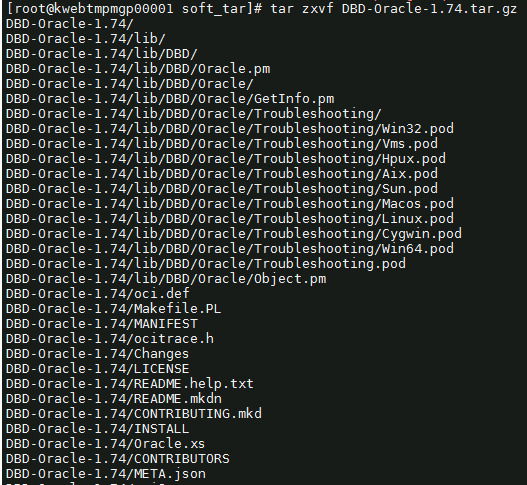
export LD\_LIBRARY\_PATH=/usr/lib/oracle/11.2/client64/lib

export ORACLE\_HOME=/usr/lib/oracle/11.2/client64



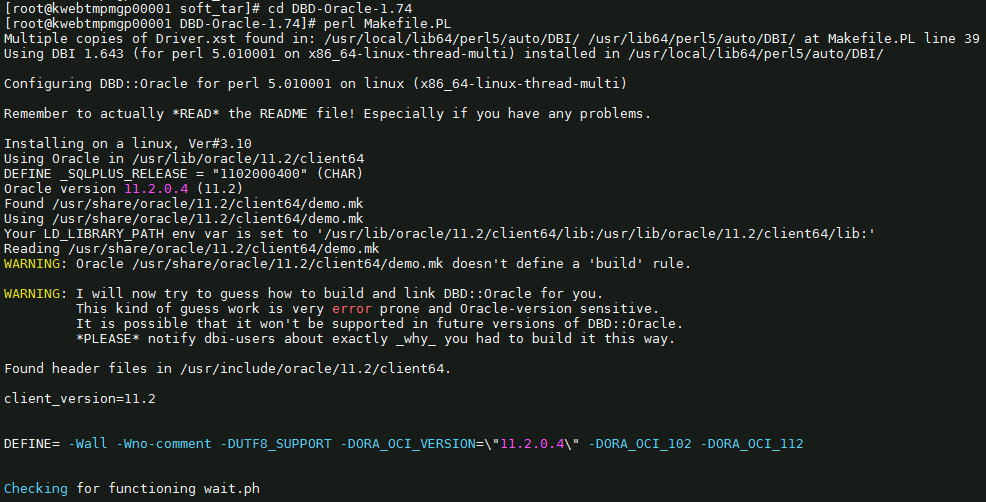
source /etc/profile

tar zxvf DBD-Oracle-1.74.tar.gz

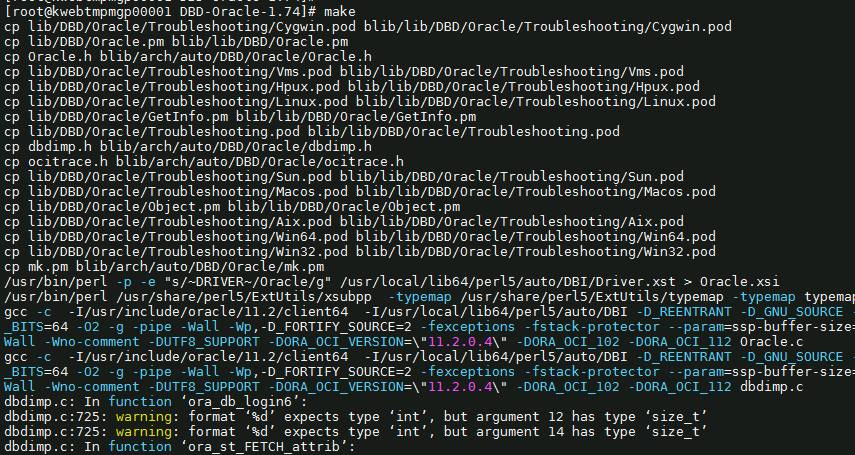


cd DBD-Oracle-1.74

perl Makefile.PL



make



make install

* + 1. 安装DBD:pg驱动模块

Vi /etc/profile

export POSTGRES\_INCLUDE="/data01/app/postgresql/include"

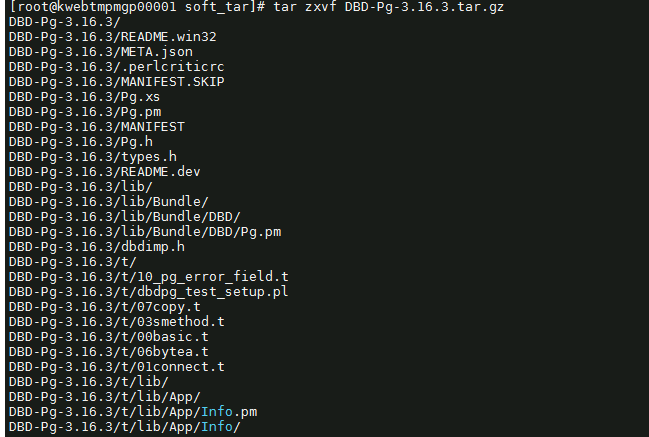
export POSTGRES\_LIB="/data01/app/postgresql/lib"

export POSTGRES\_HOME="/data01/app/postgresql"



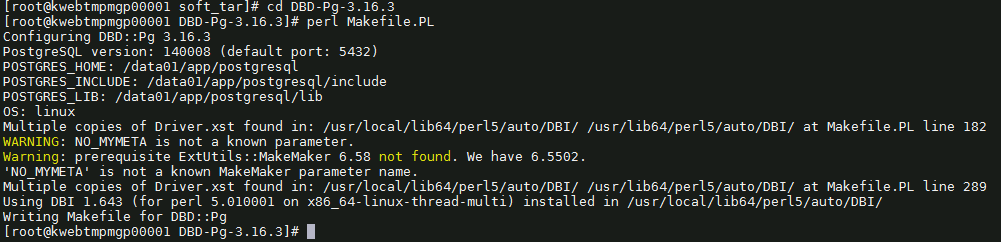
source /etc/profile

tar zxvf DBD-Pg-3.16.3.tar.gz

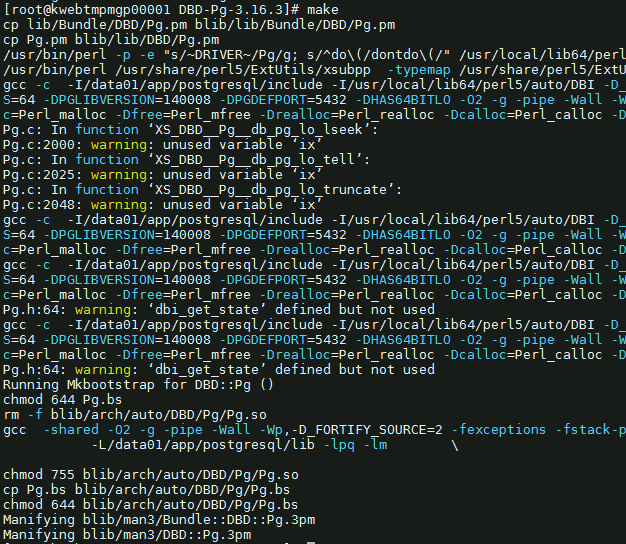


cd DBD-Pg-3.16.3

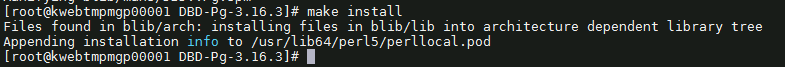
perl Makefile.PL



make

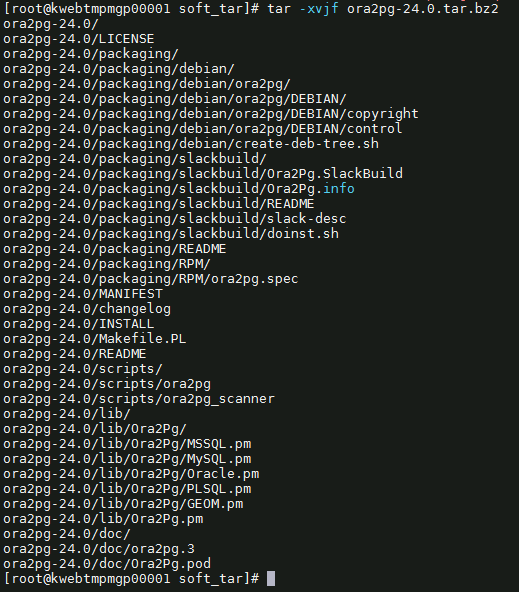


make install



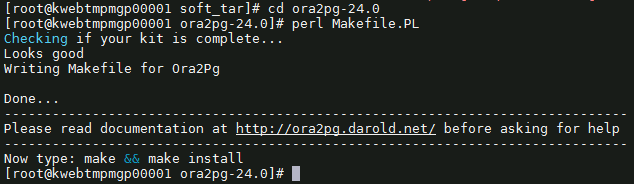
* + 1. 安装ora2pg

tar -xvjf ora2pg-24.0.tar.bz2

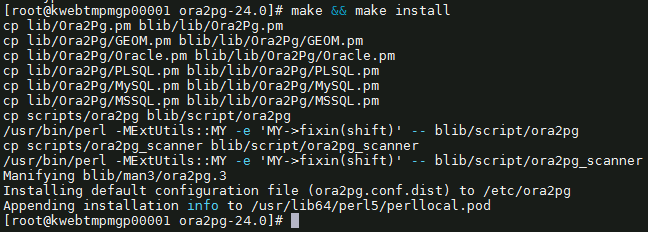


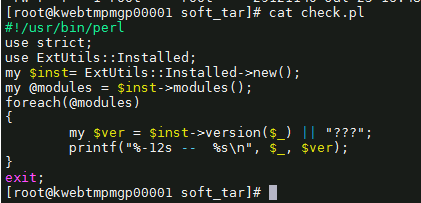
cd ora2pg-24.0

perl Makefile.PL

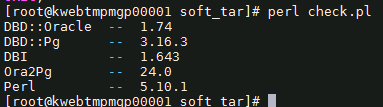


make && make install





perl check.pl



ora2pg



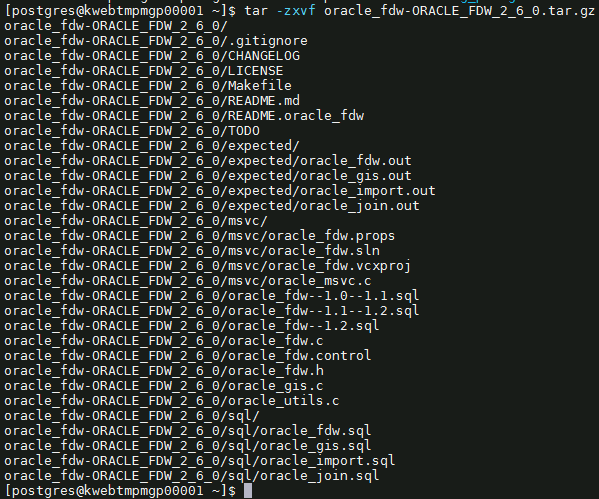
1. 安装oracle\_fdw
   1. 下载地址：

https://github.com/laurenz/oracle\_fdw

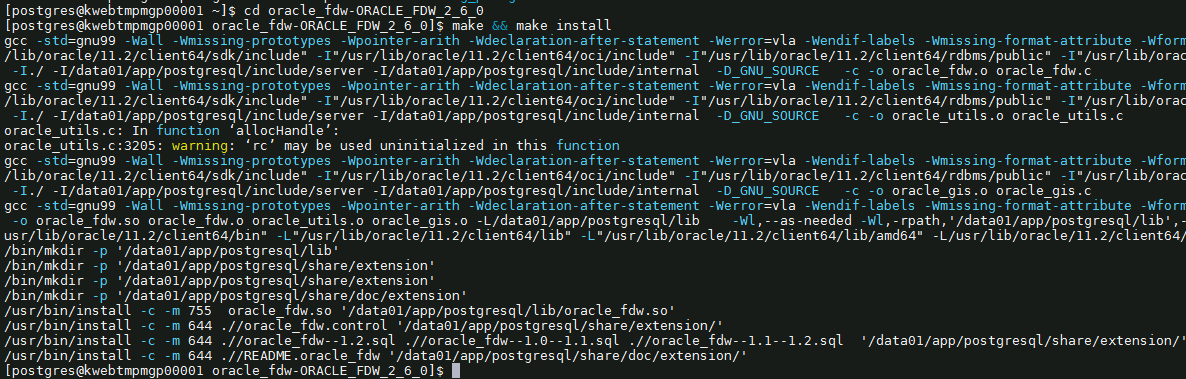
下载到 /home/postgres

* 1. 安装步骤

tar -zxvf oracle\_fdw-ORACLE\_FDW\_2\_6\_0.tar.gz



cd oracle\_fdw-ORACLE\_FDW\_2\_6\_0



* 1. 添加环境变量

vi .bash\_profile

PATH=$PATH:$HOME/bin

export PATH

export PG\_HOME=/data01/app/postgresql

export ORACLE\_HOME=/usr/lib/oracle/11.2/client64

export LD\_LIBRARY\_PATH=$PG\_HOME/lib

export PGDATA=/data01/pgsql

export PATH=$PG\_HOME/bin:$ORACLE\_HOME/bin:$PATH

PATH=$PATH:$PG\_HOME/.local/bin:$PG\_HOME/bin

export LD\_LIBRARY\_PATH=/usr/lib/oracle/11.2/client64/lib:$LD\_LIBRARY\_PATH

export TNS\_ADMIN=$ORACLE\_HOME/network/admin

export PATH

export PG\_HOME=/data01/app/postgresql

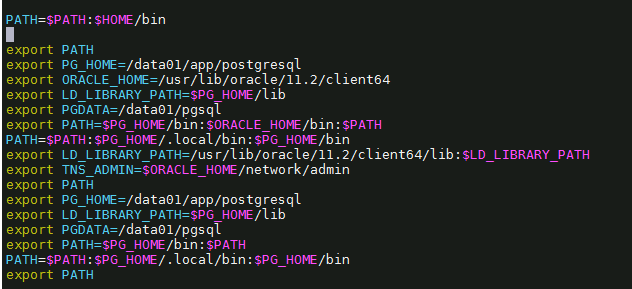
export LD\_LIBRARY\_PATH=$PG\_HOME/lib

export PGDATA=/data01/pgsql

export PATH=$PG\_HOME/bin:$PATH

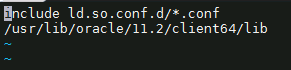
PATH=$PATH:$PG\_HOME/.local/bin:$PG\_HOME/bin

export PATH



* 1. 修改/etc/ld.so.conf添加一行

/usr/lib/oracle/11.2/client64/lib



运行ldconfig



* 1. 数据库加载插件



* 1. 新建tnsnames.ora

cd /usr/lib/oracle/11.2/client64/network/admin

vi tnsnames.ora

test\_oracle =

(DESCRIPTION =

(ADDRESS\_LIST=

(ADDRESS=(PROTOCOL=TCP)(HOST=MPMUAToracle01.beta.hic.cloud)(PORT=1521))

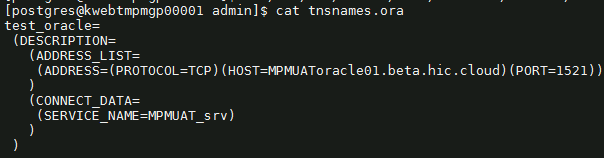
)

(CONNECT\_DATA=

(SERVICE\_NAME=MPMUAT\_srv)

)

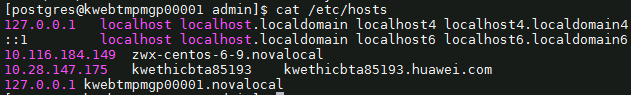
)



* 1. 修改/etc/hosts文件

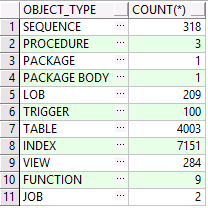
127.0.0.1 kwebtmpmgp0001.novalocal

cat /etc/hosts



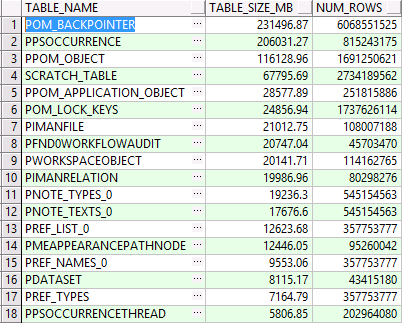
1. **数据库准备**
2. 查看原数据库信息
   1. 查原库oracle对象

select object\_type,count(\*) from dba\_objects where owner='INFODBA' group by object\_type;



* 1. 查看原库每张表的大小和记录

select table\_name,round((num\_rows \* avg\_row\_len) / 1024 /1024,2) AS table\_size\_mb,num\_rows from all\_tables where owner = 'INFODBA' and num\_rows is not null order by 2 desc;



1. 数据库准备

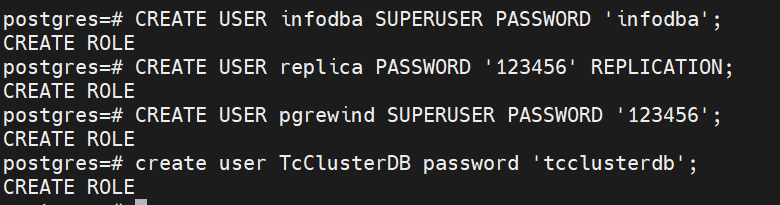
2.1.新建用户

CREATE USER infodba SUPERUSER PASSWORD 'infodba';

CREATE USER replica PASSWORD '123456' REPLICATION;

CREATE USER pgrewind SUPERUSER PASSWORD '123456';

create user TcClusterDB password 'tcclusterdb';



2.2新建表空间

-- infodba\_idata /data01/pgsql/tc/infodba\_idata

-- infodba\_ilog /data01/pgsql/tc/infodba\_ilog

-- infodba\_indx /data01/pgsql/tc/infodba\_index

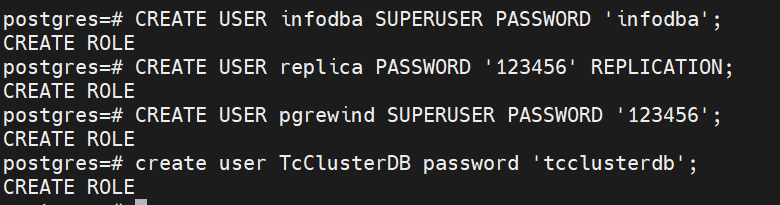
--tcclusterdb\_idata /data01/pgsql/tc/tcclusterdb\_idata

mkdir -p /data01/pgsql/tc/infodba\_idata

mkdir -p /data01/pgsql/tc/infodba\_ilog

mkdir -p /data01/pgsql/tc/infodba\_index

mkdir -p /data01/pgsql/tc/tcclusterdb\_idata

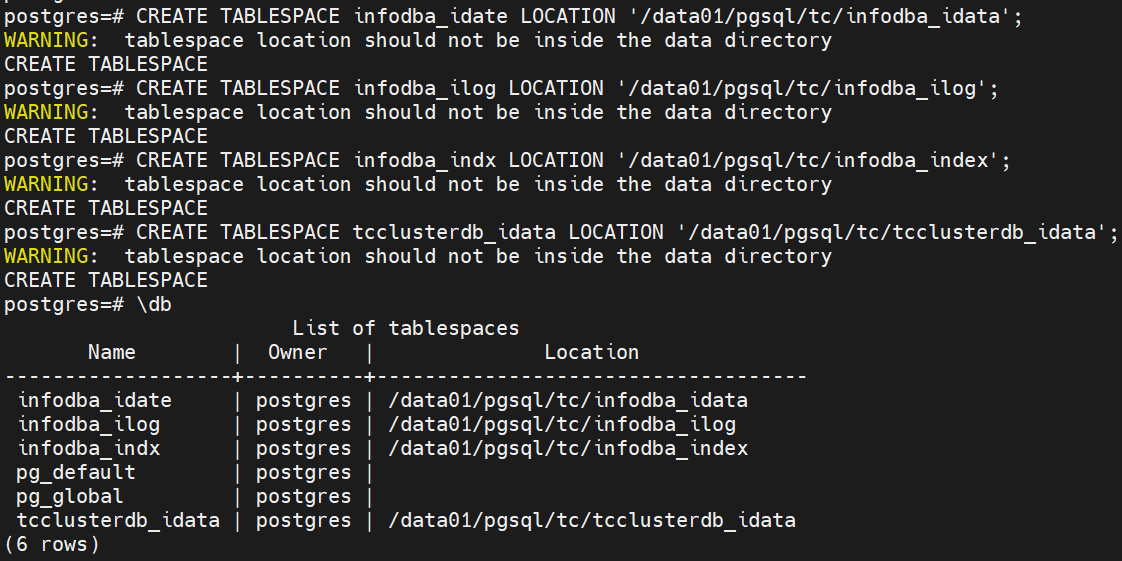


CREATE TABLESPACE infodba\_idata LOCATION '/data01/pgsql/tc/infodba\_idata';

CREATE TABLESPACE infodba\_ilog LOCATION '/data01/pgsql/tc/infodba\_ilog';

CREATE TABLESPACE infodba\_indx LOCATION '/data01/pgsql/tc/infodba\_index';

CREATE TABLESPACE tcclusterdb\_idata LOCATION '/data01/pgsql/tc/tcclusterdb\_idata';



2.3新建库

CREATE DATABASE tc WITH owner infodba encoding 'UTF8' template template0 LC\_COLLATE='C' tablespace infodba\_idata ;

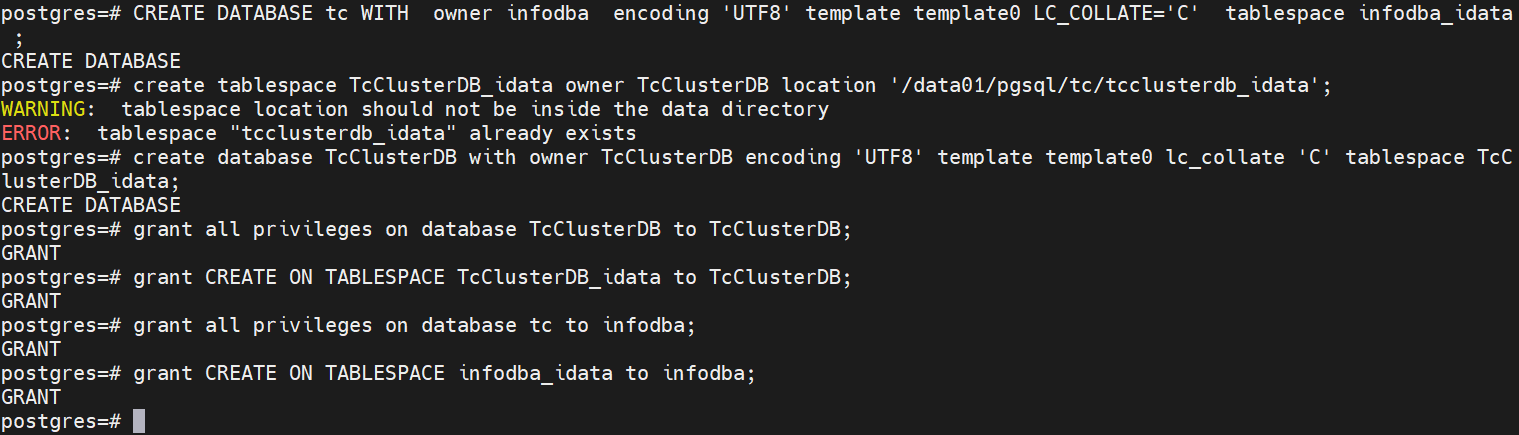
create database TcClusterDB with owner TcClusterDB encoding 'UTF8' template template0 lc\_collate 'C' tablespace TcClusterDB\_idata;

grant all privileges on database TcClusterDB to TcClusterDB;

grant CREATE ON TABLESPACE TcClusterDB\_idata to TcClusterDB;

grant all privileges on database tc to infodba;

grant CREATE ON TABLESPACE infodba\_idata to infodba;



2.4修改认证md5

vi postgres.conf

password\_encryption = md5;

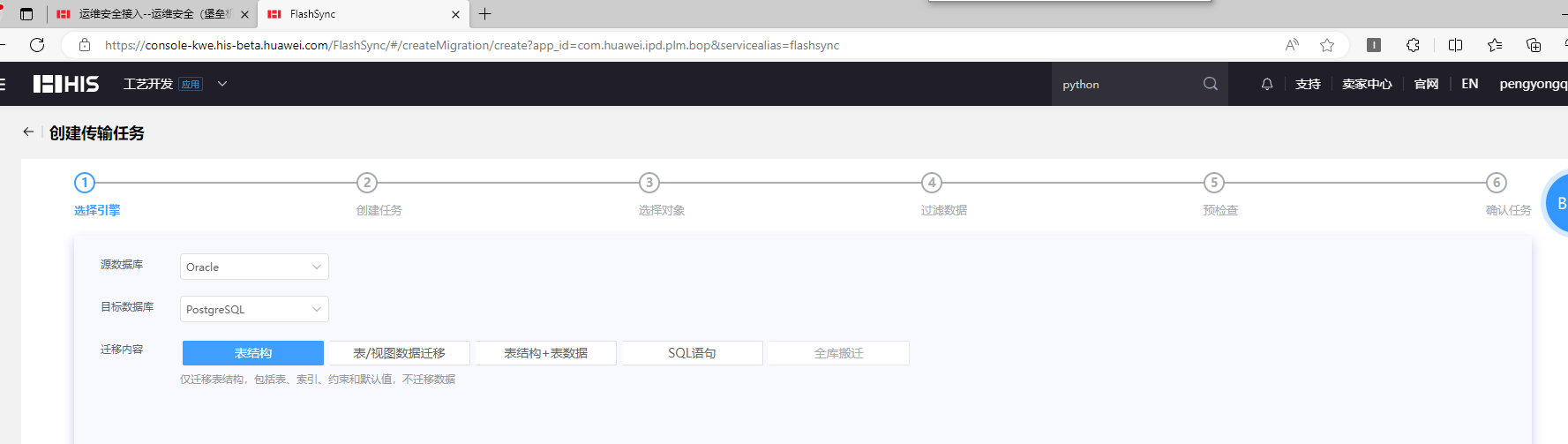


重设密码

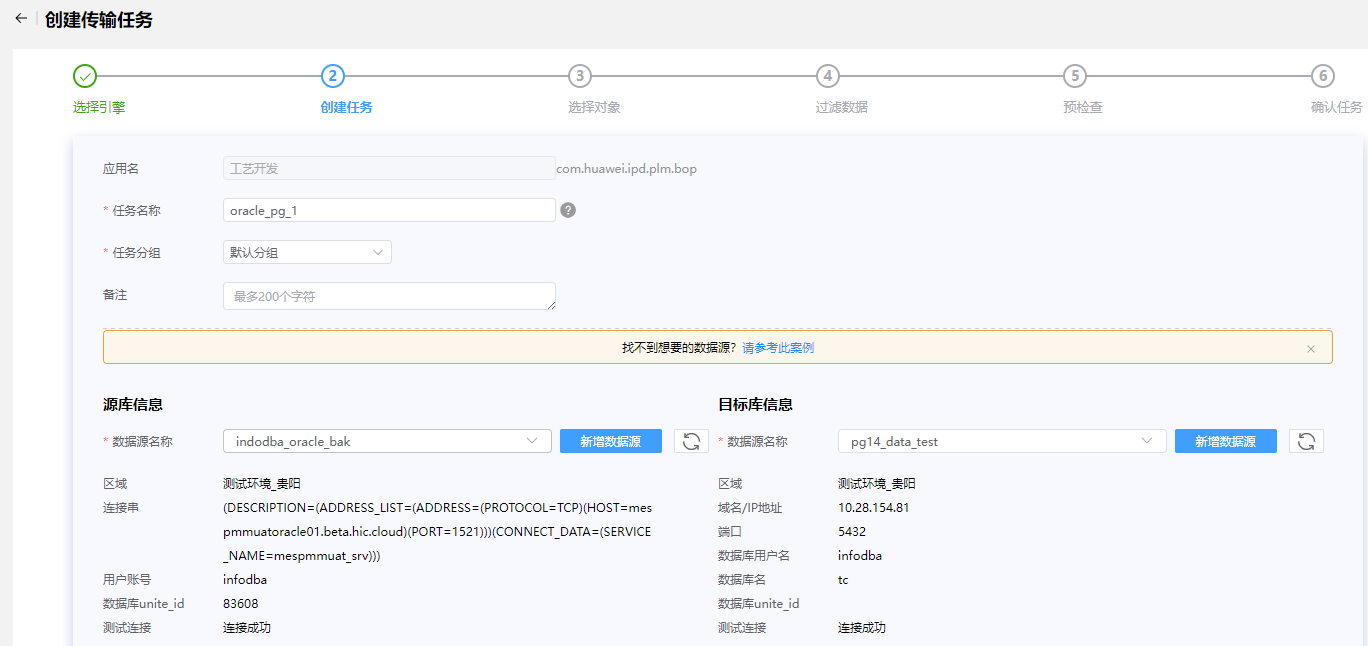
Alter user infodba password ‘infodba’;



1. **迁移为了方便创建的函数**
2. 新建函数及表
   1. 新建传输任务
3. 删除函数及表
4. **Oracle数据库表、索引迁移到PG库**
5. 使用flashsync导表结构
   1. 选择引擎



* 1. 创建任务

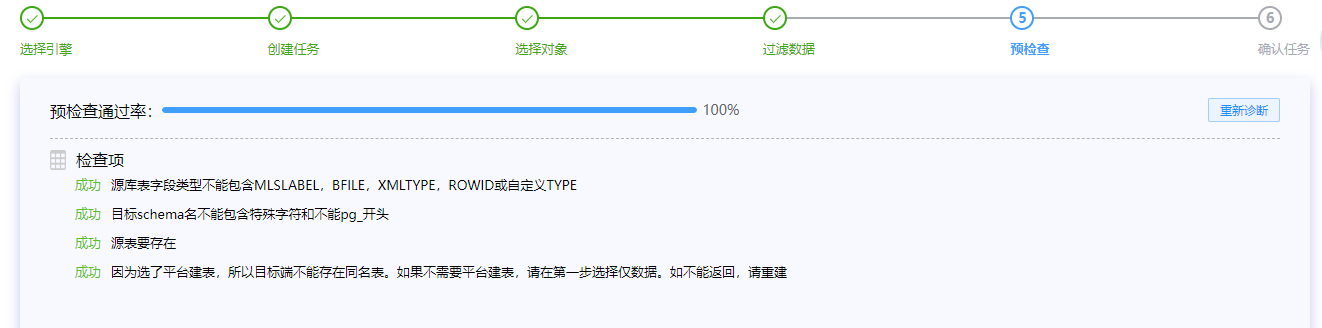


* 1. 选择对象



排除表mmv\_spatial\_cell\_index

* 1. 过滤数据



* 1. 预检查



* 1. 确认任务



1. flashsync导表数据
   1. 选择引擎

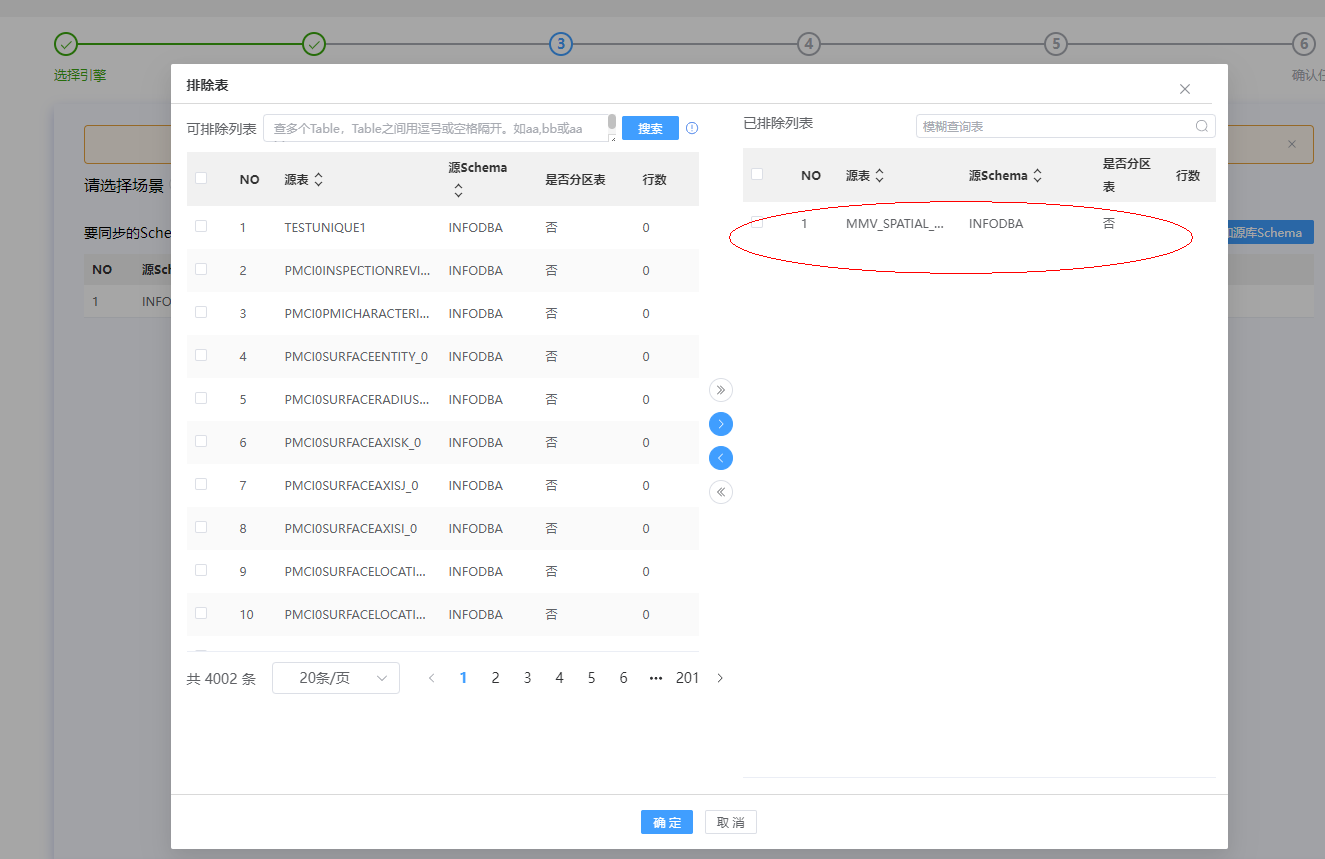


16迁移并发度

* 1. 创建任务

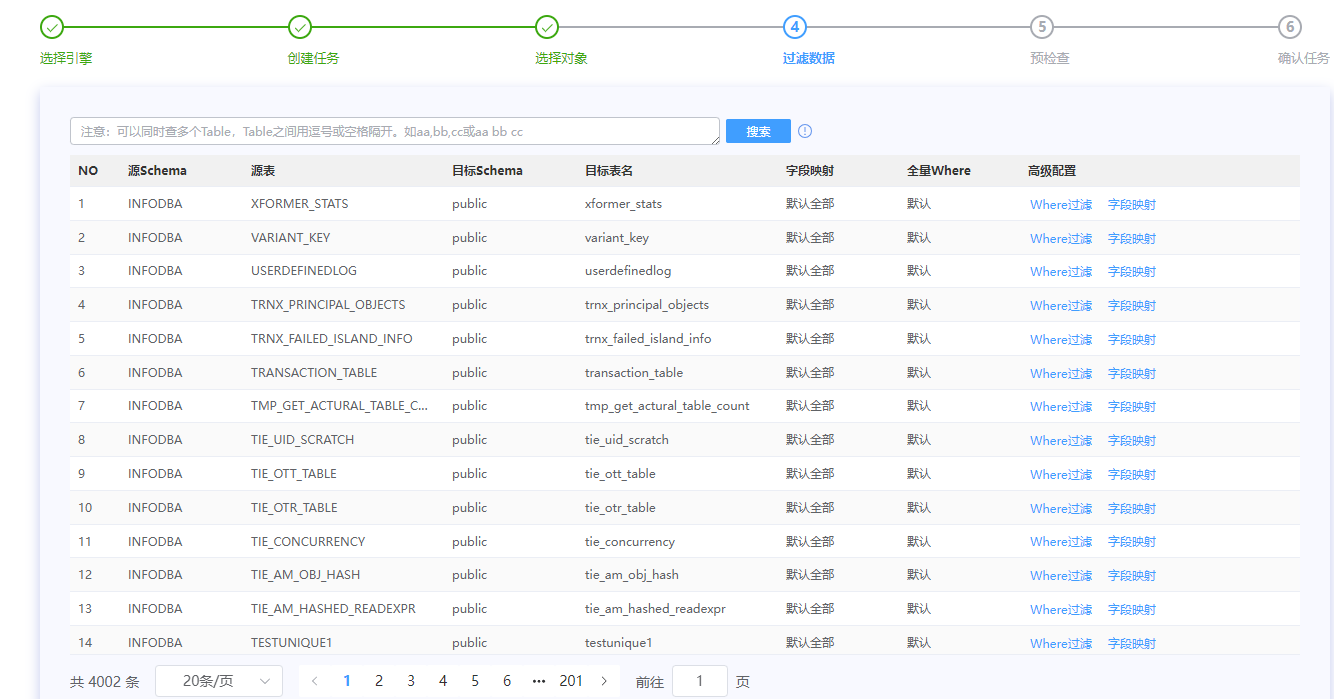


* 1. 选择对象

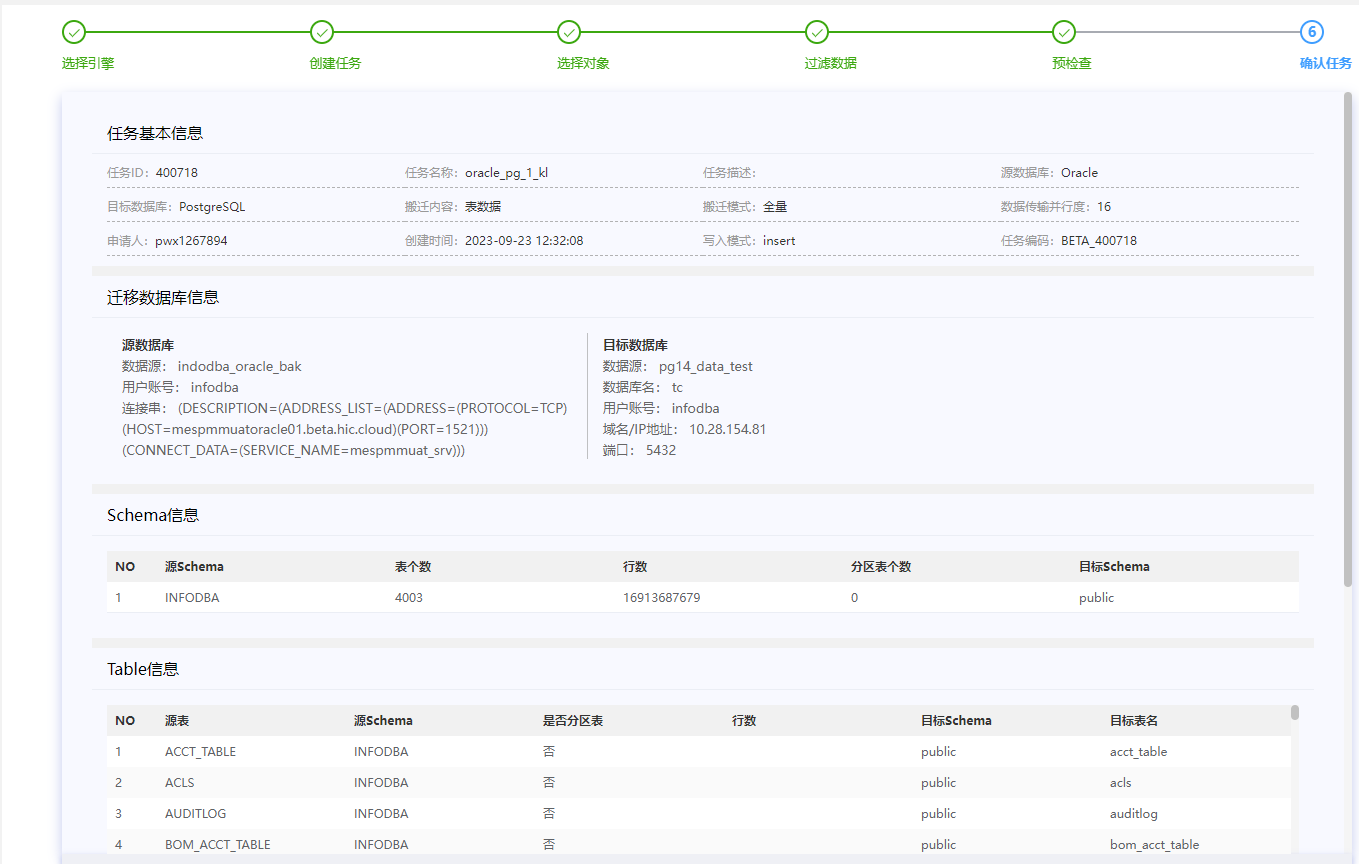


排除表mmv\_spatial\_cell\_index

* 1. 过滤数据



* 1. 预检查



* 1. 确认任务



* 1. 创建mmv\_spatial\_cell\_index （ora2pg -c ora2pg.conf -a mmv\_spatial\_cell\_index -o mmv.sql）

CREATE TABLE mmv\_spatial\_cell\_index (

puid varchar(15) NOT NULL,

spatial\_rep varchar(15) NOT NULL,

cell\_id numeric(38) NOT NULL,

occ\_path\_prefix varchar(900) NOT NULL,

occ\_path\_suffix varchar(4000),

pxmin bigint NOT NULL,

ymin bigint NOT NULL,

zmin bigint NOT NULL,

pxmax bigint NOT NULL,

ymax bigint NOT NULL,

zmax bigint NOT NULL

) ;

CREATE UNIQUE INDEX mmvspatialcellindex\_puid\_pk ON mmv\_spatial\_cell\_index (puid);

CREATE INDEX mmv\_spatial\_cellid1 ON mmv\_spatial\_cell\_index (cell\_id);

CREATE INDEX mmv\_spatial\_occpathprefix1 ON mmv\_spatial\_cell\_index (occ\_path\_prefix);

CREATE INDEX mmv\_spatial\_spatialrep1 ON mmv\_spatial\_cell\_index (spatial\_rep);

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN PUID SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN SPATIAL\_REP SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN CELL\_ID SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN OCC\_PATH\_PREFIX SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN pXMIN SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN YMIN SET NOT NULL;

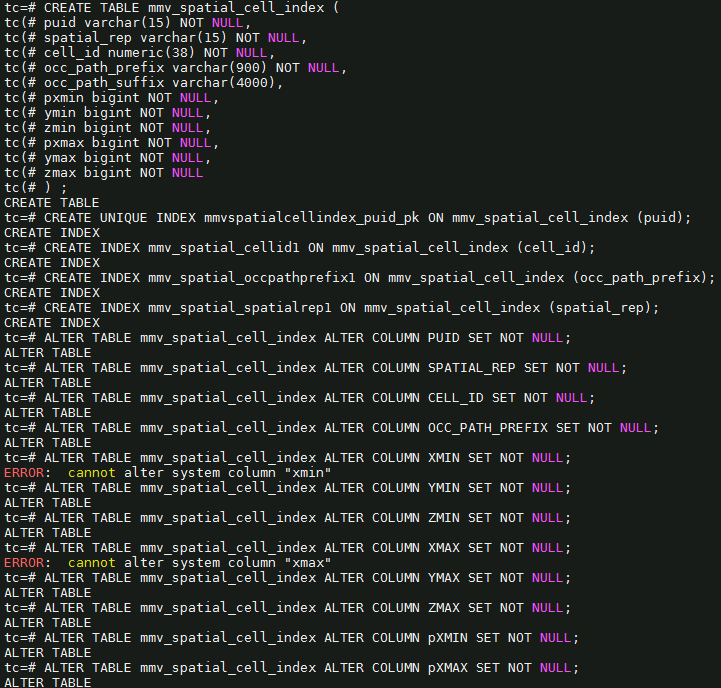
ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN ZMIN SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN pXMAX SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN YMAX SET NOT NULL;

ALTER TABLE mmv\_spatial\_cell\_index ALTER COLUMN ZMAX SET NOT NULL;

再使用工具flashsync把数据导入。



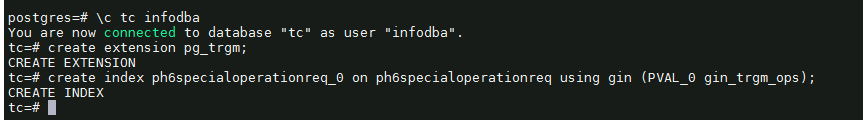
* 1. 索引报错处理

rebuild index piph6specialoperationreq\_0 error,error:index row size 2808 exceeds btree version 4 maximum 2704 for index "piph6specialoperationreq\_0" DETAIL: Index row references tuple (8386,36) in relation "ph6specialoperationreq". HINT: Values larger than 1/3 of a buffer page cannot be indexed. Consider a function index of an MD5 hash of the value, or use full text indexing.

解决方案：

create extension pg\_trgm;

create index ph6specialoperationreq\_0 on ph6specialoperationreq using gin (PVAL\_0 gin\_trgm\_ops);



安装插件create extension pg\_trgm;报错信息是extenction文件目录中缺少control文件。应该是编译信息没有遍全

( tar -zxvf postgresql-14.8.tar.gz -C /app/install

cd /app/install/postgresql-14.8

./configure –prefix=/app/postgresql

make world && make install-world)

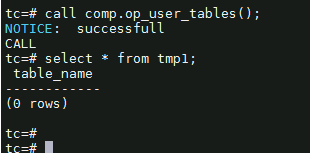
再然后在flashsync,目标表索引重建详情 选择状态 “已失败”，点击“重试”



1. **oracle数据库迁移约束、序列、视图、过程、函数、包到PG**
2. 表

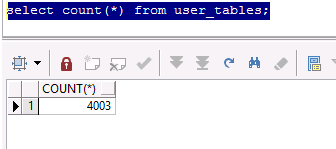
call comp.op\_user\_tables();

select \* from tmp1;



Oracle:

select count(\*) from user\_tables;

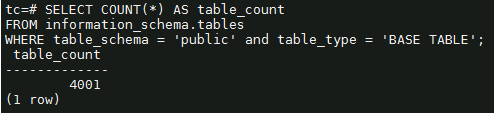


Pg14:

SELECT COUNT(\*) AS table\_count

FROM information\_schema.tables

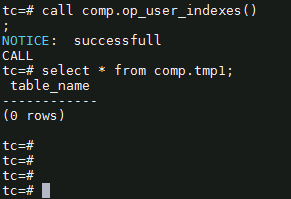
WHERE table\_schema = 'public' and table\_type = 'BASE TABLE ';



1. 索引

call comp.op\_user\_indexes();

select \* from comp.tmp1;



oracle:

select index\_name as indexname from user\_indexes i where i.index\_name not like 'SYS\_%';

pg:

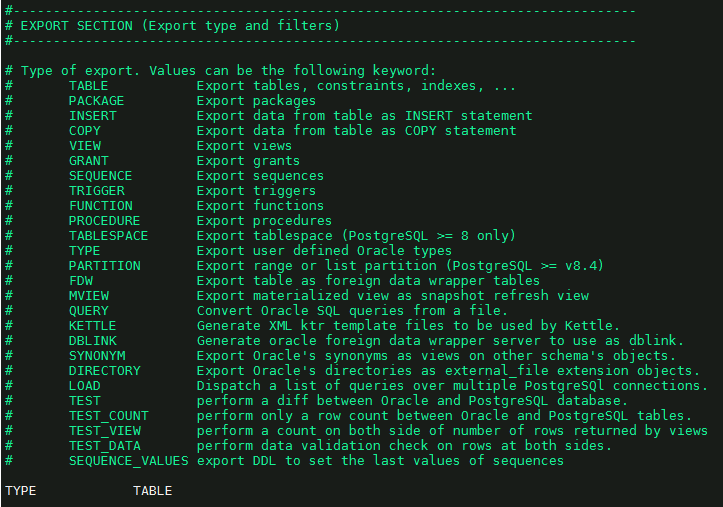
select upper(indexname) from pg\_indexes where schemaname = 'public' and indexname not like 'sys\_%';

1. 约束

vi /data01/soft\_tar/export\_dir/ora2pg.conf

--修改内容如下

TYPE TABLE



ora2pg -c /data01/soft\_tar/export\_dir/ora2pg.conf -o /data01/soft\_tar/export\_dir/test1/output.sql > /data01/soft\_tar/export\_dir/test1/output.sql.log



查看有没有报错：grep 'psql' output.sql.log

产生的 output.sql 文件用 table\_split.pl 处理下，将table的创建语句和constraint，index的创建语句分开。

table\_split.pl 内容如下

---------------------------------------------------------------------------------------

open(INPUT, "output.sql");

open(OUTPUT, ">", "table.sql");

open(OUTPUT2, ">", "constraints.sql");

open(OUTPUT3, ">", "index.sql");

while(<INPUT>){

$str.=$\_

}

@index=$str=~/CREATE\s\*(?:UNIQUE)?\s\*INDEX\s+\w+\s+ON[^\n]\*;\n/igm;

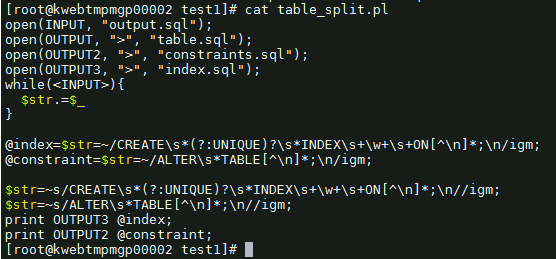
@constraint=$str=~/ALTER\s\*TABLE[^\n]\*;\n/igm;

$str=~s/CREATE\s\*(?:UNIQUE)?\s\*INDEX\s+\w+\s+ON[^\n]\*;\n//igm;

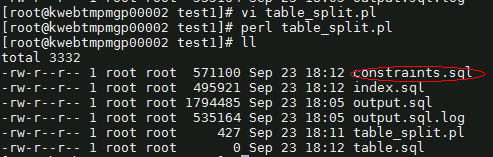
$str=~s/ALTER\s\*TABLE[^\n]\*;\n//igm;

print OUTPUT3 @index;

print OUTPUT2 @constraint;



运行 perl table\_split.pl



导入约束

psql -U infodba -d tc -f constraints.sql > constraints\_output.log 2>&1

grep ‘psql’ constraints\_output.log

psql:constraints.sql:53:ERROR:cannot alter system column “xmin”

psql:constraints.sql:56:ERROR:cannot alter system column “xmin”

psql:constraints.sql:7127:ERROR: multiple primary keys for table “pom\_lock\_keys”

(alter table pom\_lock\_keys add primary key(puid); ) –主键创造多次，可以忽略。

alter table mmv\_spatial\_cell\_index alter column pxmin set not null;

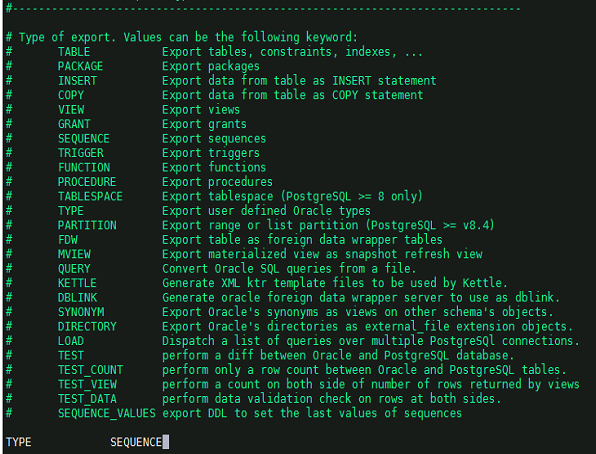
alter table mmv\_spatial\_cell\_index alter column pxmax set not null;

1. 序列

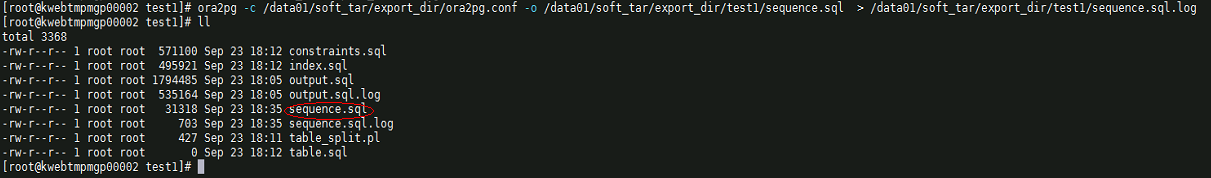
vi /data01/soft\_tar/export\_dir/ora2pg.conf

--修改内容如下

TYPE SEQUENCE

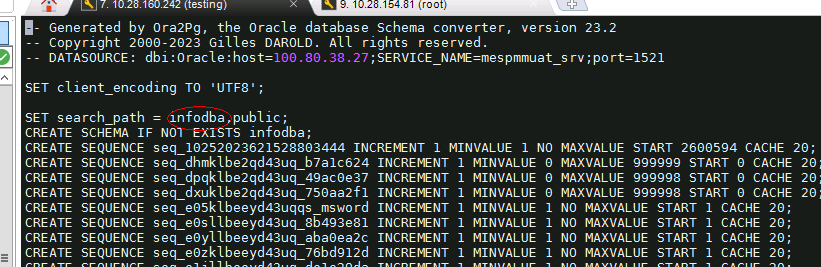


ora2pg -c /data01/soft\_tar/export\_dir/ora2pg.conf -o /data01/soft\_tar/export\_dir/test1/sequence.sql > /data01/soft\_tar/export\_dir/test1/sequence.sql.log



vi sequence.sql

把创建和设置schema删除。因为序列创建到public



psql -U infodba -d tc -f sequence.sql > sequence\_output.log 2>&1

检查：grep 'psql' sequence\_output.log

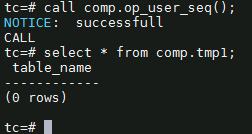
查看oracle序列个数：select count(\*) from user\_sequences; 318

查看PG序列个数：select count(\*) from information\_schema.sequences where sequence\_schema =CURRENT\_SCHEMA; 318

检查哪些没有序列导入

call comp.op\_user\_seq();

select \* from comp.tmp1;



1. 视图

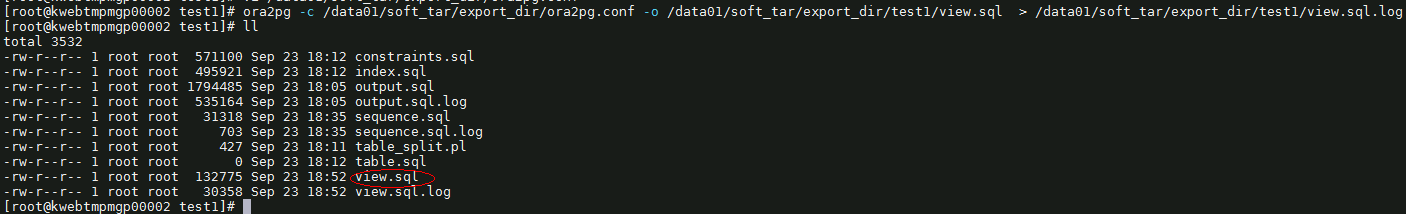
vi /data01/soft\_tar/export\_dir/ora2pg.conf

--修改内容如下

TYPE VIEW



ora2pg -c /data01/soft\_tar/export\_dir/ora2pg.conf -o /data01/soft\_tar/export\_dir/test1/view.sql > /data01/soft\_tar/export\_dir/test1/view.sql.log



导入

psql -U infodba -d tc -f view.sql > view\_output.log 2>&1

grep “psql” view\_output.log

psql:view.sql:744: ERROR: function lpad(bigint,integer,unknown) does not exist

psql:view.sql:892: ERROR: function lpad(bigint,integer,unknown) does not exist

psql:view.sql:1116: ERROR: function lpad(bigint,integer,unknown) does not exist

解决方案：

CREATE OR REPLACE FUNCTION pg\_catalog.lpad(bigint, integer, varchar)

RETURNS text

LANGUAGE sql

IMMUTABLE STRICT

AS $function$

SELECT pg\_catalog.lpad($1::text, $2, $3::text)

$function$

查看oracle视图数量：select count(\*) from user\_views; 284

查看pg视图数量：select count(\*) from information\_schema.views where table\_schema = CURRENT\_SCHEMA; 281

----有相差3个对比一下

Oracle：select view\_name as table\_name from user\_views;

把它导入pg，表名为t1

Pg: select upper(table\_name) from information\_schema.views where table\_schema ='public';

Select \* from t1

Except

select upper(table\_name) from information\_schema.views where table\_schema ='public';

结果为

VL10N\_EINFO

VL10N\_VALUE

VL10N\_AWP0CONTENTNAMS

CREATE

OR REPLACE VIEW VL10N\_EINFO AS SELECT

puid,

SUBSTR( pval\_0, 1, 5 ) AS locale,

SUBSTR( pval\_0, 7, 1 ) AS preference,

SUBSTR( pval\_0, 9, 1 ) AS status,

SUBSTR( pval\_0, 11, 4 ) AS sequence\_no,

SUBSTR( pval\_0, 49, 240 ) AS pval\_0

FROM

PL10N\_EINFO UNION ALL

SELECT

PEINFO.puid,

'NONE' AS locale,

'M' AS preference,

'M' AS status,

lpad( pseq, 4, '0' ) AS sequence\_no,

pval\_0

FROM

PEINFO,

PICML

WHERE

PICML.puid = PEINFO.puid

AND PICML.VLA\_487\_5 = 0;

CREATE

OR REPLACE VIEW VL10N\_VALUE AS SELECT

puid,

SUBSTR( pval\_0, 1, 5 ) AS locale,

SUBSTR( pval\_0, 7, 1 ) AS preference,

SUBSTR( pval\_0, 9, 1 ) AS status,

SUBSTR( pval\_0, 11, 4 ) AS sequence\_no,

SUBSTR( pval\_0, 49, 256 ) AS pval\_0

FROM

PL10N\_VALUE UNION ALL

SELECT

PVALUE.puid,

'NONE' AS locale,

'M' AS preference,

'M' AS status,

lpad( pseq, 4, '0' ) AS sequence\_no,

pval\_0

FROM

PVALUE,

PSTXT

WHERE

PSTXT.puid = PVALUE.puid

AND PSTXT.VLA\_491\_19 = 0;

CREATE

OR REPLACE VIEW VL10N\_AWP0CONTENTNAMES AS SELECT

puid,

SUBSTR( pval\_0, 1, 5 ) AS locale,

SUBSTR( pval\_0, 7, 1 ) AS preference,

SUBSTR( pval\_0, 9, 1 ) AS status,

SUBSTR( pval\_0, 11, 4 ) AS sequence\_no,

SUBSTR( pval\_0, 49, 128 ) AS pval\_0

FROM

PL10N\_AWP0CONTENTNAMES UNION ALL

SELECT

PAWP0CONTENTNAMES.puid,

'NONE' AS locale,

'M' AS preference,

'M' AS status,

lpad( pseq, 4, '0' ) AS sequence\_no,

pval\_0

FROM

PAWP0CONTENTNAMES,

PAWP0TILETEMPLATE

WHERE

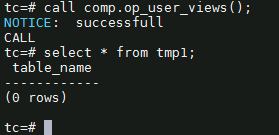
PAWP0TILETEMPLATE.puid = PAWP0CONTENTNAMES.puid

AND PAWP0TILETEMPLATE.VLA\_1310\_9 = 0;

检查是否全部导入

call comp.op\_user\_views();

select \* from tmp1;



Oracle:

Select count(\*) from user\_views;

Pg14:

SELECT table\_name

FROM information\_schema.views

WHERE table\_schema = current\_schema;

1. 过程

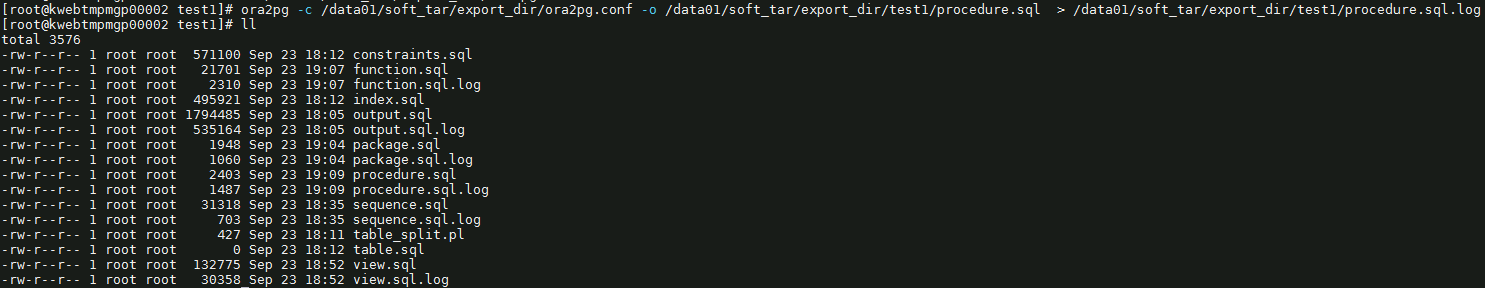
vi ora2pg.conf

--修改内容如下

TYPE PROCEDURE



ora2pg -c /data01/soft\_tar/export\_dir/ora2pg.conf -o /data01/soft\_tar/export\_dir/test1/procedure.sql > /data01/soft\_tar/export\_dir/test1/procedure.sql.log



导入

psql -U infodba -d tc -f procedure.sql > procedure\_output.log 2>&1

grep ‘psql’ procedure\_output.log

psql:procedure.sql:59:ERROR: syntax error at or near “dbms\_stats”

1. 函数

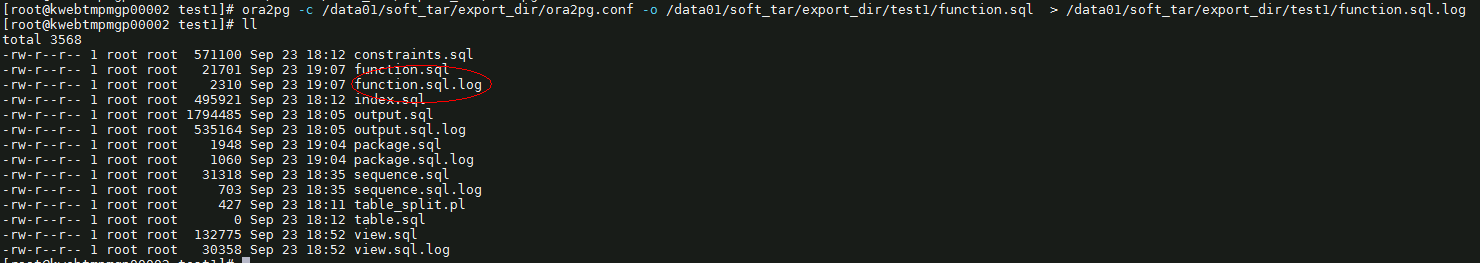
Vi ora2pg.conf

--修改内容如下

TYPE FUNCTION



ora2pg -c /data01/soft\_tar/export\_dir/ora2pg.conf -o /data01/soft\_tar/export\_dir/test1/function.sql > /data01/soft\_tar/export\_dir/test1/function.sql.log



导入（注意把schema删除）

psql -U infodba -d tc -f function.sql > function\_output.log 2>&1

grep ‘psql’ function\_output.log

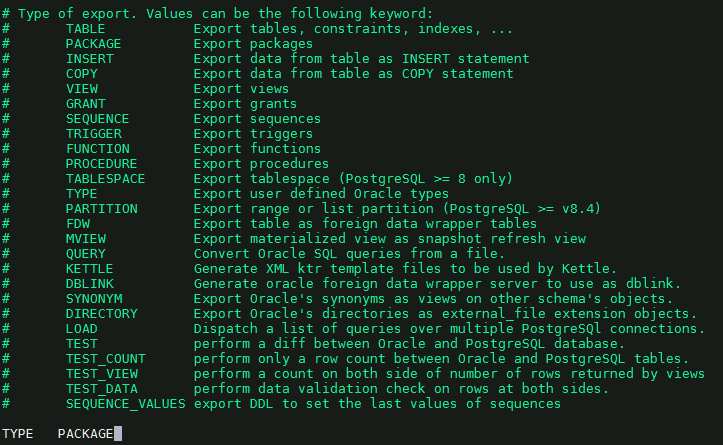
psql:function.sql:81:ERROR:syntax error at or near “is” ---此函数初始化里有，直接复制初始化库的。

1. 包

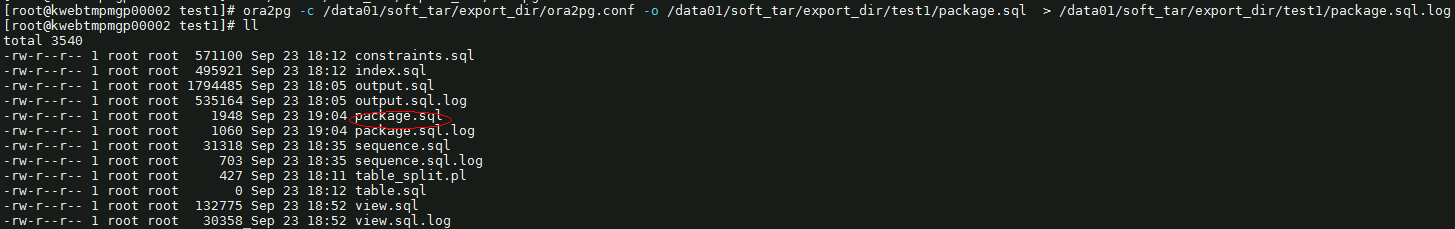
Vi ora2pg.conf

--修改内容如下

TYPE PACKAGE



ora2pg -c /data01/soft\_tar/export\_dir/ora2pg.conf -o /data01/soft\_tar/export\_dir/test1/package.sql > /data01/soft\_tar/export\_dir/test1/package.sql.log



导入

psql -U infodba -d tc -f package.sql > package\_output.log 2>&1

1. 数据量的检查

1）在oracle数据库中执行：

-------------------------------------------------------------------------------

create table TMP\_GET\_ACTURAL\_TABLE\_COUNT(table\_name varchar(50),table\_cnt int);

select \* from TMP\_GET\_ACTURAL\_TABLE\_COUNT;

CREATE OR REPLACE PROCEDURE GET\_ACTURAL\_TABLE\_COUNT(isrun integer) AUTHID CURRENT\_USER IS

cursor\_sql SYS\_REFCURSOR;

record\_count number;

BEGIN

FOR cursor\_sql IN (SELECT table\_name FROM user\_tables ) LOOP

EXECUTE IMMEDIATE 'SELECT COUNT(\*) FROM ' || cursor\_sql.table\_name INTO record\_count;

INSERT INTO TMP\_GET\_ACTURAL\_TABLE\_COUNT (table\_name, table\_cnt) VALUES (cursor\_sql.table\_name, record\_count);

COMMIT;

END LOOP;

COMMIT;

END;

/

truncate table TMP\_GET\_ACTURAL\_TABLE\_COUNT;

call GET\_ACTURAL\_TABLE\_COUNT(1);

commit;

-------------------------------------------------------------------------------

2）在postgresql数据库中执行。

-------------------------------------------------------------------------------

create table TMP\_GET\_ACTURAL\_TABLE\_COUNT(table\_name varchar(50),table\_cnt bigint);

CREATE OR REPLACE PROCEDURE GET\_ACTURAL\_TABLE\_COUNT(isrun integer)

language plpgsql

AS $$

declare

table\_count bigint;

cursor\_sql record;

begin

for cursor\_sql in(

select table\_name from information\_schema.tables where table\_schema='public' and table\_type ='BASE TABLE'

) loop

execute 'SELECT COUNT(\*) FROM ' || cursor\_sql.table\_name INTO table\_count;

insert into TMP\_GET\_ACTURAL\_TABLE\_COUNT(table\_name,table\_cnt) values(cursor\_sql.table\_name,table\_count);

commit;

end loop;

end;

$$;

truncate table TMP\_GET\_ACTURAL\_TABLE\_COUNT;

/

call GET\_ACTURAL\_TABLE\_COUNT(1);

/

select \* from TMP\_GET\_ACTURAL\_TABLE\_COUNT;

1. **oracle迁移的数据库与初始化数据库比较**
2. 表

call pp\_user\_tables();

select \* from tmp1;

pg\_dump -s -t pfndoimpactingobjects -f output\_file.sql tc\_test

psql -U infodba -d tc -f output\_file.sql > output.log 2>&1

grep 'psql' output.log

1. 索引

call pp\_user\_index() ;

select \* from tmp1;

1. 序列

call PROCEDURE pp\_user\_seq();

select \* from tmp1;

eim\_uid\_seed 和 qs\_prod\_path\_seq

1. 约束

Select \* from user\_constraints where constraint\_type =’P’; 数量1

查看oracle外键约束

Select \* from user\_constraints where constraint\_type =’P’; 数量5

Pg都导入。

1. 视图

call pp\_user\_views();

select \* from tmp1;

pg\_dump -U infodba -d tc\_test -t vl10n\_fnd0instructions -s -f output\_file.sql

psql -U infodba -d tc -f output\_file.sql > output.log 2>&1

1. 存储过程、函数
2. 触发器

pg\_dump -U infodba -d tc\_test -s -f triggers.sql

cat triggers.sql|grep 'CREATE TRIGGER' >output\_file.sql

psql -U infodba -d tc -f output\_file.sql > output.log 2>&1

1. **验证数据**
2. 表结构的一致性全量验证
3. 数据量的一致性-全量验证
4. 数据本身一致性-抽查30%验证
5. 数据编码的一致性-抽查30%验证