PostgreSQL Point-in-time Recovery (Incremental Backup)

PostgreSQL “Point-in-time Recovery” (PITR) also called as incremental database backup, online backup or may be archive backup. The PostgreSQL server records all users’ data modification transaction like insert, update or delete and write it into a file call write-ahead (WAL) log file. This mechanism use the history records stored in WAL file to do roll-forward changes made since last database full backup.

Advantages

1. Zero down time – The incremental database backup is important to critical system that cannot afford even a minute down time. With Point-in-time Recovery, database backup down time can totally eliminated because this mechanism can make database backup and system access happened at the same time.
2. Save storage size – with incremental database backup, we backup the latest archive log file since last backup instead of full database backup every day.

**Summary of PostgreSQL Backup Steps**

1. Modify postgresql.conf to support archive log
2. Make a base backup (full database backup)
3. Backup base backup to remote storage.
4. Backup WAL (archive log files) to remote storage (continuous process)

**Summary of PostgreSQL Point-in-time Recovery Steps**

1. Extract files from base backup
2. Copy files from pg\_wal folder
3. Create recovery.conf file
4. Start Recover

Changing of parameters in Database

**1) Make change in Postgresql configuration file (postgresql.auto.conf)**, we need to make some changes in postgresql.conf file to tell PostgreSQL how to copy or archive WAL files that generated from PostgreSQL server.

Modify postgresql.auto.conf

[postgres@eOfficeMasterDB~]#vim /var/lib/pgsql/10/data/postgresql.auto.conf

Make following changes in postgresql.conf

archive\_mode= on

archive\_command='scp %p remoteserverIP:/var/lib/pgsql/pg\_log\_archive/%f'

wal\_keep\_segments = '10'

Rsync the data from master to slave for PITR purpose/

Create a pg\_log\_archive folder on slave side.

root@eOfficeMasterDB~]# su – postgres

[postgres@eOfficeSRDB~]$mkdir /var/lib/pgsql/pg\_log\_archive

For Keyless authentication between Master and Slave(SR) because it will not ask for password when ever the master tried to connect to slave with postgres user.So that it can keep on archiving wal files from master to slave .

root@eOfficeMasterDB~]# su – postgres

root@eOfficeMasterDB~]$ ssh-keygen

root@eOfficeMasterDB~]$ ssh-copy-id –i postgres@SR\_IP

Restart the database

[root@eOfficeMasterDB~]#/etc/init.d/postgresql-10 restart

**Attention!!! Understand how PostgreSQL handle log files, pg\_wal and archive log**  
pg\_wal is a PostgreSQL log file folder that use to store all data history records. It located at /var/lib/pgsql/10/data/pg\_wal. For example, when user inserted , update or delete a record, all transaction history will automatically create or append to a file log file under pg\_wal folder. Log file format is look like following format 000000010000000000000001 -> 000000010000000000000006

For example,

[postgres@eOfficeMasterDBpg\_wal]#ls -lsh

total 113M

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:25 000000010000000000000006

Every log file can handle around 16M data, when it excess this limit, it will automatically create a new log file, filename is follow 0-9 and A-Z

000000010000000000000001

..

..

000000010000000000000009

..

..

00000001000000000000000A

..

..

00000001000000000000000Z

This is the log files that we going to use as the roll-forward PostgreSQL Point-in-time Recovery :.

Data Simulation & Backup Process

Create dummy tables and records – we will dynamic create 455,252 records in a new table, 400k records will force PostgreSQL to create enough log files in pg\_xlog folder and fire the archive process to archive the log files from **/var/lib/pgsql/10/data/pg\_wal** **(Master Server)**to **/var/lib/pgsql/pg\_log\_archive (Remote Server)**, every logs file contain around 16M size file.

**1) Table testPITR1 created in postgres database at 2018-02-25 17:17**

Connect with Database postgres

postgres=# select (\*) from pg\_class; –- contain 229 records

postgres=# select (\*) from pg\_description; –- contains 1988 records

postgres=# create table testPITR1 as select \* from pg\_class, pg\_description;

#--totally 229 x 1988 = 455,252 records

postgres=# select \* from current\_timestamp; –-2018-02-25 17:17

Log files look like following

**Master Server**

[postgres@eOfficeMasterDBdata]#cd pg\_wal/

[postgres@eOfficeMasterDBpg\_wal]#ls -lsh

total 112M

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:25 000000010000000000000006

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:16 000000010000000000000007

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000008

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000009

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 00000001000000000000000A

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 00000001000000000000000B

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 00000001000000000000000C

**Remote Server**

[postgres@RemoteSRpgsql]#cd pg\_log\_archive

[postgres@RemoteSRpg\_log\_archive]#ls -lsh

total 97M

16M -rw——- 1 postgrespostgres 16M 2018-11-25 17:17 000000010000000000000000

16M -rw——- 1 postgrespostgres 16M 2018-11-25 17:17 000000010000000000000001

16M -rw——- 1 postgrespostgres 16M 2018-11-25 17:17 000000010000000000000002

16M -rw——- 1 postgrespostgres 16M 2018-11-25 17:17 000000010000000000000003

16M -rw——- 1 postgrespostgres 16M 2018-11-25 17:17 000000010000000000000004

16M -rw——- 1 postgrespostgres 16M 2018-11-25 17:17 000000010000000000000005

**2) Create a full databse backup – base backup**

Connect with Database postgres and start base backup process by any one them.

postgres=# SELECT pg\_start\_backup('label', true);

pg\_start\_backup()

——————–

0/6BA9328

(1 row)

pg\_start\_backup and pg\_basebackup is use to create a label, and log it into log file.

**Use a tar command to compress all Data Directory folder to make a database base backup.**

[postgres@eOfficeMaster]$ tar -cvzf /var/lib/pgsql/db\_file\_backup-date.tar -C /var/lib/pgsql/10/data/ .

**Remember !!! db\_file\_backup-20180225.tar**this is the full database backup (base backup) including Postgresqlconfiguration , system and all others files and folder.

pg\_stop\_backup() create a label in log file as well.

postgres=#Select pg\_stop\_backup();

————————

0/6BA9384

(1 row)

**3) Table testPITR2 created at 2018-02-25 18:08:06** –prepare for Point-in-time Recovery

**Master Server**

Connect with Database postgres

postgres=# create table testPITR2 as select \* from pg\_class, pg\_description;

postgres=# select \* from current\_timestamp; --2018-02-25 18:08:06

[postgres@eOfficeMasterDBdata]#cd pg\_wal/

[postgres@eOfficeMasterDBpg\_wal]#ls -lsh

total 113M

8.0K -rw—— 1 postgrespostgres 254 2018-02-25 18:02 000000010000000000000006.00BA9328.backup

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000A

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000B

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000C

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:08 00000001000000000000000D

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:06 00000001000000000000000E

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000F

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000010

**Slave Server**

[postgres@RemoteSRpg\_log\_archive]#ls -lsh

total 209M

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000000

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000001

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000002

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000003

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000004

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000005

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:02 000000010000000000000006

8.0K -rw——- 1 postgrespostgres 254 2018-02-25 18:02 000000010000000000000006.00BA9328.backup

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:06 000000010000000000000007

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000008

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000009

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000A

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000B

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000C

P.S The pg\_start\_backup() and pg\_stop\_backup() backup labels will created in **000000010000000000000006.00BA9328.backup** file.

[postgres@localhostwals]$cat 000000010000000000000006.00BA9328.backup

START WAL LOCATION: 0/6BA9328 (file 000000010000000000000006)

STOP WAL LOCATION: 0/6BA9384 (file 000000010000000000000006)

CHECKPOINT LOCATION: 0/6BA9328

START TIME: 2018-02-25 17:45:24 IST

LABEL: Full\_Backup

STOP TIME: 2018-02-25 18:02:18 IST

**3) Table testPITR3 created at 2018-02-25 18:15:23**–prepare for Point-in-time Recovery

**Master Server**

Connect with Database postgres

postgres=# create table testPITR3 as select \* from pg\_class, pg\_description;

postgres=# select \* from current\_timestamp; --–2008-11-25 18:15:23

[postgres@eOfficeMasterDB pg\_wal]#ls -lsh

total 129M

8.0K -rw——- 1 postgrespostgres 254 2008-11-25 18:02 000000010000000000000006.00BA9328.backup

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000010

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000011

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000012

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 000000010000000000000013

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 000000010000000000000014

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000015

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000016

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000017

**Slave Server**

[postgres@RemoteSRpg\_log\_archive]#ls -lsh

total 321M

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000000

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000001

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000002

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000003

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000004

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000005

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:02 000000010000000000000006

8.0K -rw——- 1 postgrespostgres 254 2018-02-25 18:02 000000010000000000000006.00BA9328.backup

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:06 000000010000000000000007

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000008

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000009

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000A

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000B

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000C

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 00000001000000000000000D

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 00000001000000000000000E

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 00000001000000000000000F

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000010

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000011

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 000000010000000000000012

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 000000010000000000000013

**4) Table testPITR4 created at 2018-02-25 20:00:04** –prepare for Point-in-time Recovery

**Master Server**

Connect with Database postgres

postgres=# create table testPITR4 as select \* from pg\_class, pg\_description;

postgres=# select \* from current\_timestamp; –-2018-02-25 20:00:04

[postgres@eOfficeMasterDBpg\_wal]#ls -lsh

total 129M

8.0K -rw——- 1 postgrespostgres 254 2018-02-25 18:02 000000010000000000000006.00BA9328.backup

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000017

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000018

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000019

16M -rw——- 1 postgrespostgres 16M 2018-02-25 20:01 00000001000000000000001A

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 00000001000000000000001B

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:58 00000001000000000000001C

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:58 00000001000000000000001D

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 00000001000000000000001E

**Slave Server**

[postgres@RemoteSRpg\_log\_archive]#ls -lsh

total 417M

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000000

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000001

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000002

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000003

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000004

16M -rw——- 1 postgrespostgres 16M 2018-02-25 17:17 000000010000000000000005

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:02 000000010000000000000006

8.0K -rw——- 1 postgrespostgres 254 2018-02-25 18:02 000000010000000000000006.00BA9328.backup

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:06 000000010000000000000007

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000008

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 000000010000000000000009

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000A

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000B

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:07 00000001000000000000000C

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 00000001000000000000000D

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 00000001000000000000000E

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 00000001000000000000000F

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000010

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:14 000000010000000000000011

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 000000010000000000000012

16M -rw——- 1 postgrespostgres 16M 2018-02-25 18:15 000000010000000000000013

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:58 000000010000000000000014

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:58 000000010000000000000015

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000016

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000017

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000018

16M -rw——- 1 postgrespostgres 16M 2018-02-25 19:59 000000010000000000000019

**We created 4 tables for PITR recovery**, remember table creation time, we going to do recovery base on the time later.

Table testPITR1 created at 2018-06-21 18:09:08.334795+05:30

Table testPITR2 created at 2018-06-21 18:13:08.849434+05:30

Table testPITR3 created at 2018-06-21 18:16:06.147784+05:30

Table testPITR4 created at 2018-06-21 18:18:54.581011+05:30

postgres=# \d

List of relations

Schema | Name | Type | Owner

——–+———–+——-+———-

public | testpitr1 | table |postgres

public | testpitr2 | table |postgres

public | testpitr3 | table |postgres

public | testpitr4 | table |postgres

(4 rows)

Copy

**Attention!!!**Before move on, please study above transaction log files movement that generated by PostgreSQL. We have to fully understand when PostgreSQL will create a new log file and when it will move to archive folder, do not forget the log file format :) ~ take sometime to review and understand the above log file generation sequence

Disaster come in

Let us suppose Database Master Server has been crashed or manually shut down the master database service

1) Kill the postgresql process of master and slave

root@Remotemaster~]# kill -9 $(head -1 /var/lib/pgsql/10/data/postmaster.pid)

root@RemoteSR~]# kill -9 $(head -1 /var/lib/pgsql/10/data/postmaster.pid)

Recovery Process

Finally we reach recovery process, Please remember 1 file and 2 folders

**a) Base backup file** – db\_file\_backup-date.tar  
**b) Log files hanv’t archive yet** – all files under Pg\_wal folder  
**c) pg\_log\_archive** – all archive files under pg\_log\_archive folder (Remote storage )

**Remember!!!**Assume we already having backup **db\_file\_backup-20180225.tar** and all archived files .

[root@RemoteSR~]# mkdir -p /var/lib/pgsql/10/pitrbackup/basebackup

[root@Remotemaster~]# scp -r /var/lib/pgsql/db\_file\_backup-20180225.tar postgres@SR\_IP:/var/lib/pgsql/10/pitrbackup/basebackup/.

[root@RemoteSR~]# cp -r /var/lib/pgsql/pg\_log\_archive/ postgres@srip:/var/lib/pgsql/10/pitrbackup/.

**1) Rename Data directory toData.bad.data**, assume database file in Data Directory folder was damaged due to disaster we created just now, we need to create a fresh database later.

[root@RemoteSR~]# mv /var/lib/pgsql/10/data /var/lib/pgsql/10/data.bad\_old

**2) Unzip / extract files from db\_file\_backup-20180225.tar**, create a new **Data** folder under **/var/lib/pgsql/10**, it just like what we did before. Move all extracted files from **db\_file\_backup-20180225.tar** to **/var/lib/pgsql/10/data/**.

[root@RemoteSR~]# cd /var/lib/pgsql/10/pitrbackup/basebackup

[root@RemoteSR~]# tar -xvzf pgdatabk20181125.tar

[root@RemoteSR~]# mkdir /var/lib/pgsql/10/data

[root@RemoteSR~]# chmod 700 /var/lib/pgsql/10/data

[root@RemoteSR~]# chown -R postgres:postgres /var/lib/pgsql

[root@RemoteSR~]# mv \* /var/lib/pgsql/10/data/.

**Start database**

[root@RemoteSR~]# /etc/init.d/postgresql-10 start

postgres=# \d

List of relations

Schema | Name | Type | Owner

——–+———–+——-+———-

public | testpitr1 | table |postgres

Copy

Table testPITR1 created at 2018-02-25 17:17 is restored. This testPITR1 table is created before base backup process launched, so this is correct.

**3) Copy log files from pg\_wal folder**. Some log files still located in Data.bad.datapg\_wal folder (those log files hanv’t archive yet) during disaster happening, we need to copy the log file back and recover it as much as possible.

[root@Remotemaster~]# scp –r /var/lib/pgsql/10/Data.bad.data/pg\_wal/0\* postgres@SRIP:/var/lib/pgsql/10/pitrbackup/.

cp -r /var/lib/pgsql/pg\_log\_archive/\* /var/lib/pgsql/10/pitrbackup/.

**4) Create a recovery.conf file and put it under /usr/local/pgsql/pgDataPITR**

Vim recovery.conf

Create following content in **recovery.conf**

restore\_command='cp /var/lib/pgsql/10/pitrbackup/%f %p'

recovery\_target\_time='2018-06-21 18:13:08'

**a) /var/lib/pgsql/pg\_log\_archive/**is the folder that we backup our archive log files  
**b) recovery\_target\_time**is the time we need to recover to. Omit this setting will make PostgreSQL recover as much as it can, it may recover all changes.

Remember four tables creation time

Table testPITR1 created at 2018-06-21 18:09:08.334795+05:30

Table testPITR2 created at 2018-06-21 18:13:08.849434+05:30

Table testPITR3 created at 2018-06-21 18:16:06.147784+05:30

Table testPITR4 created at 2018-06-21 18:18:54.581011+05:30

**Remember !!!** Above recovery.conf file will make PostgreSQL take the archive log files from **/var/lib/pgsql/pg\_log\_archive/** folder and recover the data changes until **2018-02-25 18:08:06** (table testPITR2 created).

**5) Start database**

[postgres@RemoteSR~]#/var/lib/pgsql/10/bin/pg\_ctl start -D /var/lib/pgsql/10/data

$ psql

postgres=# \d

List of relations

Schema | Name |Type | Owner

--------+-----------+-------+----------

public | testpitr1 | table |postgres

public | testpitr2 | table |postgres

(2 rows)

Copy

Table testpitr2 is restored back.

P.S After recovery process finished, recovery.conf will rename to recovery.done by PostgreSQL to avoid start the recovery process again.

We can the view pg.log file to understand how PostgreSQL process the recovery process.

[postgres@SRpg\_log]$cat postgresql-2018-02-25\_174500.log

STATEMENT: selectpg\_start\_backup("Full Backup - Testing");

LOG: database system was interrupted; last known up at 2018-11-25 17:45:23 MYT

LOG: starting archive recovery

LOG: restore\_command='cp /var/lib/pgsql/10/pitrbackup/%f %p'

LOG: recovery\_target\_time='2018-02-25 18:08:06+08'

cp: cannot stat `/var/lib/pgsql/pg\_log\_archive/00000001.history': No such file or directory

LOG: restored log file"000000010000000000000006.00BA9328.backup" from archive

LOG: restored log file"000000010000000000000006" from archive

LOG: automatic recovery in progress

LOG: redo starts at 0/6BA9368

LOG: restored log file"000000010000000000000007" from archive

LOG: restored log file"000000010000000000000008" from archive

LOG: restored log file"000000010000000000000009" from archive

LOG: restored log file"00000001000000000000000A" from archive

LOG: restored log file"00000001000000000000000B" from archive

LOG: restored log file"00000001000000000000000C" from archive

LOG: restored log file"00000001000000000000000D" from archive

LOG: recovery stopping before commit of transaction 395, time2018-02-25 18:08:34.180397+08

LOG: redo done at 0/D85E0FC

LOG: last completed transaction was at log time2018-02-25 18:08:34.180397+08

C

**Attention !!!Attention !!!** THIS IS ONE TIME PROCESS, after recovery process started and finished, we cant make any recovery changes (like roll forward to another time).

If we want to roll forward to another restore time, we need to start whole recovery process again, like extract files from base backup and copy log files. This is because after PostgreSQL recovered the data , all log files format will changed to other format like following

[postgres@eOfficeMasterDBpg\_wal]#cd pg\_wal/

[postgres@localhostpg\_wal]#ls -ls

total 147696

16408 -rw------- 1 postgrespostgres 16777216 2018-02-26 14:28 00000002000000000000000D

8 -rw------- 1 postgrespostgres 83 2018-02-26 14:22 00000002.history

16408 -rw------- 1 postgrespostgres 16777216 2018-02-26 14:30 00000003000000000000000D

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:16 00000003000000000000000E

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:17 00000003000000000000000F

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:17 000000030000000000000010

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:45 000000030000000000000011

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:17 000000030000000000000012

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:17 000000030000000000000013

16408 -rw------- 1 postgrespostgres 16777216 2018-02-25 17:17 000000030000000000000014

8 -rw------- 1 postgrespostgres 158 2018-02-26 14:30 00000003.history

After recovered, log file number will increased  
00000001 –> 00000002 –> 00000003

If we want to **restore table testPITR3 created at 2018-02-25 18:15:23**, we are unable to do it, it will output error in log file unless we start the whole recovery from recover.conf process