

优炫数据库uxpool工具使用手册 2.1



UXSINO
优炫软件

优炫数据库uxpool工具使用手册 2.1

版权 © 2016-2023 北京优炫软件股份有限公司

法律声明

优炫数据库管理系统(简称: UXDB) 是由北京优炫软件股份有限公司开发并发布的一款商业性数据库管理系统。

优炫数据库管理系统 (UXDB) 的一切知识产权以及与该软件产品相关的所有信息内容, 包括但不限于: 文字表述及其组合、图标、图饰、图表、色彩、界面设计、版面框架、有关数据、及电子文档等均属北京优炫软件股份有限公司所有。本软件及其文档的任何使用、复制、修改、出租、传播、销售及分发等行为均须经北京优炫软件股份有限公司书面许可。

凡侵犯北京优炫软件股份有限公司知识产权的行为, 北京优炫软件股份有限公司将依法追究其法律责任。

本声明的最终解释权归属于北京优炫软件股份有限公司。



和其他优炫公司商标均为北京优炫软件股份有限公司的商标。

本文档提及的其他所有商标或注册商标, 由各自的所有人拥有。

注意

由于产品版本安装或其他原因, 本文档内容会不定期进行更新。除非另有约定, 本文档仅作为使用指导, 本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

北京优炫软件股份有限公司 (总部)

- 地址: 北京市海淀区学院南路62号中关村资本大厦11层 (邮编: 100081)
 - 网址: <http://www.uxsino.com>
 - 邮箱: <uxdb_support@uxsino.com>
 - 电话: 010-82886998
 - 传真: 010-82886338
 - 服务热线: 400-650-7837
-

目录

前言	vi
1. 文档目的	vi
2. 文档对象	vi
3. 修改记录	vi
1. 概述	1
1.1. 工作模式	1
1.2. 功能特性	1
1.3. 程序模块	2
2. 安装	4
2.1. 安装UXDBServer	4
2.2. 安装uxpool	4
3. 配置前的准备工作	5
3.1. 网络配置	5
3.2. 配置ssh秘钥	5
3.2.1. 生成秘钥	5
3.2.2. 修改配置	6
3.2.3. 复制公钥	7
3.2.4. 验证配置	7
3.3. 流复制	7
3.3.1. 初始化集群	8
3.3.2. 流复制配置	8
3.3.3. 创建备库	9
3.3.4. 配置recovery.conf	10
3.3.5. 配置.uxpass	11
3.3.6. 配置备节点	12
3.3.7. 流复制数据同步示例	12
4. 配置uxpool	15
4.1. 配置uxpool.conf	15
4.2. 配置pool_hba.conf	15
4.3. 配置pcp.conf	16
5. 管理uxpool	18
5.1. 启动uxpool	18
5.2. 停止uxpool	18
5.3. 限制最大连接数	18
5.4. 复制和负载均衡管理	18
5.5. 查看uxpool版本号	19
6. uxpool场景示例	20
6.1. 复制和负载均衡示例	20
6.1.1. 复制模式配置	20
6.1.2. 复制模式下的负载均衡	25
6.1.3. 复制模式下的高可用	26
6.1.4. 复制模式下的宕机恢复	28
6.2. 使用流复制的主备切换场景示例	31
6.2.1. 主备模式配置	31
6.2.2. 主备切换	37
6.2.3. 主备切换模式下的高可用	41
7. 常见问题	44
7.1. uxpool服务停止	44
7.2. 虚拟IP绑定情况	44
8. 参考	47
8.1. 服务器命令	47

8.1.1.	uxpool	47
8.2.	PCP命令	48
8.2.1.	pcp_common_options	49
8.2.2.	pcp_node_count	50
8.2.3.	pcp_node_info	50
8.2.4.	pcp_watchdog_info	52
8.2.5.	pcp_proc_count	54
8.2.6.	pcp_proc_info	54
8.2.7.	pcp_pool_status	55
8.2.8.	pcp_detach_node	56
8.2.9.	pcp_attach_node	57
8.2.10.	pcp_promote_node	57
8.2.11.	pcp_stop_uxpool	58
8.2.12.	pcp_recovery_node	59
8.3.	其他命令	59
8.3.1.	ux_md5	59
8.3.2.	ux_enc	60
8.3.3.	uxproto	62
8.3.4.	uxpool_setup	65
8.3.5.	watchdog_setup	69
8.4.	SQL类型命令	74
8.4.1.	UXPOOL SHOW	74
8.4.2.	UXPOOL SET	76
8.4.3.	UXPOOL RESET	76
8.4.4.	SHOW POOL_STATUS	77
8.4.5.	SHOW POOL_NODES	77
8.4.6.	SHOW POOL_PROCESSES	78
8.4.7.	SHOW POOL_POOLS	79
8.4.8.	SHOW POOL_VERSION	89
8.4.9.	SHOW POOL_CACHE	90
8.5.	uxpool_adm扩展	90
8.5.1.	uxpool_adm_pcp_node_info	91
8.5.2.	uxpool_adm_pcp_pool_status	92
8.5.3.	uxpool_adm_pcp_node_count	93
8.5.4.	uxpool_adm_pcp_attach_node	93
8.5.5.	uxpool_adm_pcp_detach_node	94

表格清单

1. 文档更新记录	vi
1.1. 工作模式与功能对应表	1
1.2. 模式优缺点	2
2.1. 安装配置说明	4
3.1. uxpool复制和负载均衡示例环境	8
3.2. ux_basebackup 参数说明	9
6.1. uxpool复制和负载均衡示例环境	20
6.2. uxpool复制和负载均衡示例环境	31
8.1. UXPOOL SHOW 参数说明	74
8.2. UXPOOL RESET 参数说明	77
8.3. pcp_node_info 参数说明	92
8.4. pcp_pool_status 参数说明	92
8.5. pcp_node_count 参数说明	93
8.6. pcp_attach_node 参数说明	94
8.7. pcp_detach_node 参数说明	94

前言

1. 文档目的

本文档介绍了uxpool的安装、配置和主备切换等功能，为软件的使用和维护提供必要的信息。

2. 文档对象

- 技术支持工程师
- 维护工程师

3. 修改记录

修改记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

表 1. 文档更新记录

工具版本	发布日期	修改说明
2.1.1.5C	2022-12-21	第一次正式发布。

第 1 章 概述

uxpool是UXDBServer服务器之间一种有效的中间件和UXDBServer数据库客户端。

uxpool提供的功能，如下所示。

- 连接池

uxpool保持已经连接到UXDBServer服务器的连接，当一个相同新连接(如用户名、数据库、协议版本)进来时，重用它们。它减少了连接开销，提高了系统的整体吞吐量。

- 复制

uxpool可以管理多个UXDBServer服务器。使用复制功能可以使2个或更多的物理磁盘上创建一个实时备份，这样服务不会因服务器的磁盘故障而中断。

- 负载均衡

如果数据库是复制的，在任何服务器上执行一个SELECT查询会返回相同的结果。uxpool复制特性的优势在于减少每个UXDBServer服务器上的负载，因为它可以使用分布在多个服务器之间进行SELECT查询，从而提高系统的整体吞吐量。最好是查询和UXDBServer服务器数量成一定比例，多用户同时执行多查询达到负载均衡最好的效果。

- 限制连接数

UXDBServer会限制当前的最大并发连接数，当连接数达到这个数量时，新的连接将被拒绝。增加这个最大连接数会增加资源消耗，并且对系统的全局性能有一定的负面影响。uxpool也支持限制最大连接数，但它的做法是将超过限制的连接放入到队列，而不是立即返回一个错误。

1.1. 工作模式

uxpool存在三种工作模式，如下所示。

- 原始模式

只实现一个故障切换的功能，可以配置多个后端数据库，当第一个后端数据库不能工作时，uxpool会切换到第二个后端数据库，如果第二个后端数据库也不能工作，再切换到第三个后端数据库，以此类推。

- 复制模式

此模式下，实现了同步复制的功能，实际上就是把修改数据库的操作分发到后端所有数据库上处理，而只读查询则发送给任意一台数据库，此模式下可以实现负载均衡的功能。

- 主备模式

此模式下，使用其它软件（UXDB）完成实际的数据复制功能，如使用流复制的复制模式，此时uxpool主要提供高可用的功能。

1.2. 功能特性

表 1.1. 工作模式与功能对应表

功能/模式	原始模式	复制模式	主/备模式
连接池	X	0	0

功能/模式	原始模式	复制模式	主/备模式
复制	X	0	X
负载均衡	X	0	0
故障恢复	0	0	0
需要的服务器数	1或更多	2或更多	2或更多
是否需要系统数据库	否	否	否

针对表 1.1 “工作模式与功能对应表”的说明如下所示。

- 0意味着可用，X意味着不可用。
- (*1) 表示并行查询模式需要同时打开复制和负载均衡，但是复制和负载均衡无法用于并行查询模式中的分布式表。
- 模式优缺点

表 1.2. 模式优缺点

模式	优点	缺点
复制模式	<ol style="list-style-type: none"> 1. 复制是同步的，不存在最终一致性的问题 2. 自动Failover 3. 可以做读的负载均衡 4. 容易配置 	<ol style="list-style-type: none"> 1. 写性能不是很好，有30%的写性能下降 2. 不支持部分查询，如一些随机函数、序列号、直接在不同的后端数据库上执行SQL，将产生不同的结果，所以在复制模式下，不能使用这些函数及序列号
主备模式	<ol style="list-style-type: none"> 1. 写性能较好，只有10%~20%的写性能下降 2. 自动Failover 3. 可以做读的负载均衡 	<ol style="list-style-type: none"> 1. 复制是异步的，对应用的使用有限制 2. 不能实现DDL的复制，不支持大对象的复制 3. 配置复杂

1.3. 程序模块

uxpool的主程序模块只有一个，名称为uxpool，uxpool还提供了一些命令行管理工具，这些工具都是由“pcp”开头的，存放于/home/uxdb/uxdbinstall/uxpool/bin目录下，如下所示。

```
pcp_attach_node
pcp_detach_node
pcp_node_count
pcp_node_info
pcp_proc_count
pcp_proc_info
pcp_promote_node
pcp_recovery_node
pcp_stop_uxpool
```


pcp_watchdog_info
ux_md5
uxpool

这些命令行工具有有的用于在线恢复，有的用于查询信息。

uxpool还提供了一个生成md5的小工具，ux_md5, 这个工具主要用于配置uxpool的密码文件，uxpool的密码文件中存放的密码都是md5值，所以配置时需要使用这个工具。

第 2 章 安装

使用uxpool之前需要先安装UXDBServer和uxpool。

2.1. 安装UXDBServer

1. UXDBServer的安装包选择uxdb-std-linux7-x86_64-v2.1.1.4.run。
2. 使用root用户安装，进入安装包所在路径后，直接运行./uxdb-std-linux7-x86_64-v2.1.1.4.run可以按照提示自动安装UXDBServer，关于UXDBServer的安装配置请参见《优炫数据库 2.1 安装手册》，安装路径为/home/uxdb/uxdbinstall。安装完成后，请配置正确的license工具。

2.2. 安装uxpool

1. uxpoo1的安装包文件：uxpool-linux-v4.1.1.tar.bz2。
2. 解压安装包，执行install.sh脚本进行安装，安装在/home/uxdb/uxdbinstall/uxpool路径下。
3. 如果要配置多个uxpool，则需要每台机器上分别安装uxpool，uxpool和UXDBServer可以安装在同一台主机上，也可以分别安装在不同的主机上，下面均以两台机器192.71.0.197和192.71.0.199上分别安装uxpool和UXDBServer进行配置说明。

表 2.1. 安装配置说明

IP地址	uxpool	UXDBServer	主备关系
192.71.0.197	已安装	已安装	主库
192.71.0.199	已安装	已安装	备库

第 3 章 配置前的准备工作

3.1. 网络配置

不同的主机之间，如果需要互相访问，就需要配置网络环境，如果需要一个外部访问接口，则需要配置虚拟IP，以两台机器为例，具体配置方法如下所示。

1. 分别在192.71.0.197和192.71.0.199两台机器，互相配置对应IP，执行命令`vi /etc/hosts`进行设置，主机名分别设置为`ux57.com`、`ux58.com`、`uc60.com`。设置如下所示。

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.0.197 ux57.com
192.168.0.199 ux58.com
192.168.0.200 uc60.com
```

2. 其中192.71.0.200作为虚拟IP,用于外部程序接口，可以通过该地址访问数据库。
3. 保证机器之间可以互相ping通。
4. 添加防火墙规则，允许访问。

3.2. 配置ssh密钥

为了实现uxpool的高可用，机器之间需要实现不需要密码就可以访问，因此需要配置ssh。

本次配置以两台机器为例，192.71.0.197作为主库，192.71.0.199作为备库，具体配置如下所示。

3.2.1. 生成密钥

在两台机器上分别生成ssh，默认保存在`~/.ssh`目录中，使用`uxdb`用户登录，进入`/home/uxdb`目录，执行如下命令。

ssh-keygen -t rsa

主库如下所示。

```

[uxdb@localhost ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/uxdb/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/uxdb/.ssh/id_rsa.
Your public key has been saved in /home/uxdb/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:63sDpzGuQBvF+1+ZQsM/DCN4A4q9IbFXGIriKPNIDNU uxdb@localhost.localdomain
The key's randomart image is:
+---[RSA 2048]---+
|
| . O +
|o + E =
|+O = + + .
|=+ B o S *
|=O + = O+=.* O
|.O. + .O*. B
| . .O.OO .
| . .OO..
+----[SHA256]-----+

```

备库如下所示。

```

[uxdb@localhost ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/uxdb/.ssh/id_rsa):
Created directory '/home/uxdb/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/uxdb/.ssh/id_rsa.
Your public key has been saved in /home/uxdb/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:8GfwuKWFCq05eSqQWpyrRRwlh+D/Spt5Ch+m75ekRbo uxdb@localhost.localdomain
The key's randomart image is:
+---[RSA 2048]---+
|....O
| . .+
| . . . .
| . . . O =
| o.= S *
| o =.= . B
| .B+=.O O
| .O*EB+
|O+B%+
+----[SHA256]-----+
[uxdb@localhost ~]$

```

3.2.2. 修改配置

分别在主备库上修改/etc/ssh/ssh_config文件的配置，在文件末添加如下内容。

```

PubkeyAuthentication yes
StrictHostKeyChecking no
UserKnownHostsFile /dev/null

```

配置完成之后分别在主备库重启ssh服务，执行如下命令。

```
systemctl restart sshd
```

3.2.3. 复制公钥

将主备库的公钥拷互相复制到~/.ssh目录下，并追加到authorized_keys文件后面。

在主库下，将备库id_rsa.pub拷贝到主库，并将id_rsa.pub内容追加到authorized_keys文件后面。

```
scp uxdb@192.71.0.199:~/.ssh/id_rsa.pub ~/.ssh/id_rsa.pub_temp
cat ~/.ssh/id_rsa.pub_temp >> ~/.ssh/authorized_keys
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
rm ~/.ssh/id_rsa.pub_temp
```

在备库下，将主库id_rsa.pub拷贝到备库。

```
scp uxdb@192.71.0.197:~/.ssh/authorized_keys ~/.ssh/id_rsa.pub_temp
cat ~/.ssh/id_rsa.pub_temp >> ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
rm ~/.ssh/id_rsa.pub_temp
```

3.2.4. 验证配置

uxdb用户登录，进入/home/uxdb目录，执行如下命令。

在主备库都执行以下命令，验证主备库互信和主备库自身ssh信任（主备切换时需要用）。

```
ssh 192.71.0.199
exit
ssh 192.71.0.197
exit
```

```
[uxdb@localhost ~]$ ssh 192.71.0.199
Warning: Permanently added '192.71.0.199' (ECDSA) to the list of known hosts.
Last login: Wed Nov 17 19:44:58 2021 from 192.71.0.196
[uxdb@localhost ~]$ exit
logout
Connection to 192.71.0.199 closed.
```

```
[uxdb@localhost ~]$ ssh 192.71.0.197
Warning: Permanently added '192.71.0.197' (ECDSA) to the list of known hosts.
Last login: Wed Nov 17 19:54:45 2021 from 192.71.0.199
[uxdb@localhost ~]$ exit
logout
Connection to 192.71.0.197 closed.
```

验证完成后，需要exit退出。

3.3. 流复制

主备模式不能使用uxpool自身的复制功能进行配置，因此配置主备模式时，可以采用UXDB的流复制功能。

创建一个简单的示例环境，如[表 3.1 “uxpool复制和负载均衡示例环境”](#)所示。

表 3.1. uxpath复制和负载均衡示例环境

主机名称	IP地址	uxpool	UXDBServer	实例名称	数据库名称
ux57.com	192.71.0.197	已安装	主库	master	uxdb
ux58.com	192.71.0.199	已安装	备库	slave_1	uxdb

这个数据库实例是在操作系统用户“uxdb”下，uxpool安装在/home/uxdb/uxdbinstall/uxpool目录下，UXDBServer是安装在/home/uxdb/uxdbinstall/dbsql/bin目录下，具体操作步骤如下所示。

3.3.1. 初始化集群

在192.71.0.197主机上进入数据库bin目录下，初始化并启动master。

```
./initdb -W -D master
./ux_ctl -D master start
```

3.3.2. 流复制配置

1. 创建流复制用户

进入/home/uxdb/uxdbinstall/dbsql/bin目录下，执行./uxsql，进入控制台执行如下命令。

```
uxdb=# create user repuser replication LOGIN CONNECTION LIMIT 3 ENCRYPTED
PASSWORD '123456';
```

```
uxdb=#
uxdb=# create user repuser replication LOGIN CONNECTION LIMIT 3 ENCRYPTED PASSWORD '123456';
CREATE ROLE
uxdb=#
```

2. 配置ux_hba.conf

在master的ux_hba.conf末尾添加如下内容。

```
# "local" is for Unix domain socket connections only
local all all md5
# IPv4 local connections:
host all all 0.0.0.0/0 md5
# IPv6 local connections:
host all all ::1/128 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
local replication all md5
host replication all 127.0.0.1/32 md5
host replication all ::1/128 md5
host replication repuser ux58.com md5
```

3. 配置uxsinodb.conf

在master端的uxsinodb.conf配置如下所示。

```
listen_addresses = '*'
```

```

port = 5432
wal_level = replica
archive_mode = on
archive_command = 'cd ./'
hot_standby = on
max_wal_senders = 3
wal_keep_segments = 64
full_page_writes = on
wal_log_hints = on

```

如果需要配置同步流复制模式，还需配置如下的参数，不配置时默认为异步流复制模式。

```

synchronous_commit = on
synchronous_standby_names='ux58' #此处为application_name（此处的应用名称不是机器名，可以为任意名称，synchronous_standby_names不能包含@等特殊字符）

```

4. 重启master数据库

ux_ctl -D master restart

```

[uxdb@ux57 master]$ vi ux_hba.conf
[uxdb@ux57 master]$ vi uxsinodb.conf
[uxdb@ux57 master]$ cd ../
[uxdb@ux57 bin]$ ./ux_ctl -D master restart
waiting for server to shut down.... done
server stopped
waiting for server to start.....2018-09-20 15:28:45.080 CST [2977] LOG:  listening on IPv4
address "0.0.0.0", port 5432
2018-09-20 15:28:45.080 CST [2977] LOG:  listening on IPv6 address ":::", port 5432
2018-09-20 15:28:45.098 CST [2977] LOG:  listening on Unix socket "/tmp/.s.UXSQL.5432"
2018-09-20 15:28:45.112 CST [2977] LOG:  redirecting log output to logging collector process
2018-09-20 15:28:45.112 CST [2977] HINT:  Future log output will appear in directory "log".
uxmaster status starting
uxmaster status ready
done
server started
[uxdb@ux57 bin]$

```

3.3.3. 创建备库

- master主机上创建主库后，登录备机slave_1，进入/home/uxdb/uxdbinstall/dbsql/bin目录下，执行如下命令。

```
./ux_basebackup -D slave_1/ -Fp -Xs -v -P -h ux57.com -p 5432 -U repuser -W -R
```

参数含义，如下所示。

表 3.2. ux_basebackup 参数说明

名称	描述
-h	主机地址
-W	需要密码
-Fp	普通文件格式输出
-Xs	通过流复制抓取备份日志
-R	在输出目录默认创建一个standby.signal文件（uxdb-2.1.1.0之后将recovery.conf文件合

名称	描述
	并到了uxsino.conf中，这里将不再生成recovery.conf文件)
-U	用户
-D	指定数据库存放的目录

```
[uxdb@ux58 bin]$ ./ux_basebackup -D slave_1/ -Fp -Xs -v -P -h ux57.com -p 5432 -U repuser
Password:
ux_basebackup: initiating base backup, waiting for checkpoint to complete
ux_basebackup: checkpoint completed
ux_basebackup: write-ahead log start point: 0/2000028 on timeline 1
ux_basebackup: starting background WAL receiver
23566/23566 kB (100%), 1/1 tablespace
ux_basebackup: write-ahead log end point: 0/20000F8
ux_basebackup: waiting for background process to finish streaming ...
ux_basebackup: base backup completed
[uxdb@ux58 bin]$
```

- 各库创建后，同样需要修改slave_1中ux_hba.conf，进入/uxdb/uxdbinstall/dbsql/bin目录下，在slave_1的ux_hba.conf末尾添加如下内容。

```
# "local" is for Unix domain socket connections only
local    all                all                                md5
# IPv4 local connections:
host     all                all                                0.0.0.0/0          md5
# IPv6 local connections:
host     all                all                                ::1/128            md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
local    replication        all                                md5
host     replication        all                                127.0.0.1/32       md5
host     replication        all                                ::1/128            md5
host     replication        repuser                        ux57.com            md5
```

注意

如果是uxdb-2.1.1.0之前的版本请按照[配置recovery.conf](#)和[配置.uxpass](#)继续做后续的配置。

如果是uxdb-2.1.1.0及其之后的版本，请按照[配置各节点](#)继续做后续的配置。

3.3.4. 配置recovery.conf

- master端需要配置

1. 进入/home/uxdb/uxdbinstall/dbsql/share目录下，复制recovery.conf.sample文件并存放于master数据目录下，重命名为recovery.done，如图所示。

```
[uxdb@ux57 bin]$ pwd
/home/uxdb/uxdbinstall/dbsql/bin
[uxdb@ux57 bin]$ cp ../share/recovery.conf.sample master/recovery.done
[uxdb@ux57 bin]$ vim master/recovery.done
[uxdb@ux57 bin]$
```

2. 配置recovery.done文件


```

recovery_target_timeline = 'latest'
standby_mode = on
primary_conninfo = 'application_name=ux57 host=ux58.com port=5432 user=repuser
password=123456'
trigger_file = '/home/uxdb/uxdbinstall/dbsql/bin/master/trigger_file'

```

- Slave_1端配置

1. 进入/home/uxdb/uxdbinstall/dbsql/share目录下，复制recovery.conf.sample文件并存放于slave_1数据目录下，重命名为recovery.conf，如图所示。

```

[uxdb@ux58 bin]$ pwd
/home/uxdb/uxdbinstall/dbsql/bin
[uxdb@ux58 bin]$ pwd
/home/uxdb/uxdbinstall/dbsql/bin
[uxdb@ux58 bin]$ cp ../share/recovery.conf.sample slave_1/recovery.conf
[uxdb@ux58 bin]$ vim slave_1/recovery.conf
[uxdb@ux58 bin]$ █

```

2. 配置recovery.conf文件。

```

recovery_target_timeline = 'latest'
standby_mode = on
primary_conninfo = 'application_name=ux58.com host=ux57.com port=5432 user=repuser
password=123456'
trigger_file = '/home/uxdb/uxdbinstall/dbsql/bin/slave_1/trigger_file'

```

3.3.5. 配置.uxpass

1. 在master上配置访问slave_1参数，在 ~目录下创建并配置 .uxpass文件。

```
ux58.com:5432:uxdb:repuser:123456
```

```

[uxdb@ux57 bin]$ cd ~
[uxdb@ux57 ~]$ vim .uxpass
[uxdb@ux57 ~]$ chmod -R 600 .uxpass
[uxdb@ux57 ~]$ █
[uxdb@ux57 ~]$ █

```

```
ux58.com:5432:uxdb:repuser:123456
```

2. 在slave_1上配置访问master参数。

```
ux57.com:5432:uxdb:repuser:123456
```

```

[uxdb@ux58 slave_1]$ cd ~
[uxdb@ux58 ~]$ vim .uxpass
[uxdb@ux58 ~]$ chmod -R 600 .uxpass
[uxdb@ux58 ~]$ █
[uxdb@ux58 ~]$ █

```

```
ux57.com:5432:uxdb:repuser:123456
```

3.3.6. 配置备节点

1. 进入/home/uxdb/uxdbinstall/dbsql/bin/slave_1目录下，修改standby.signal文件，添加备节点标识。

```
standby_mode = 'on'
```

2. 配置uxsinodb.conf文件。

```
recovery_target_timeline = 'latest'  
primary_conninfo = 'application_name=ux58 host=ux57.com port=5432 user=repuser  
password=123456 '  
promote_trigger_file = '/home/uxdb/uxdbinstall/dbsql/bin/slave_1/promote_trigger_file'
```

3. 配置uxsinodb.auto.conf文件。

在uxsinodb.auto.conf中有自动生成的primary_conninfo配置，里面没有application_name配置，在里面添加节点名称。

```
recovery_target_timeline = 'latest'  
primary_conninfo = 'application_name=ux58 host=ux57.com port=5432 user=repuser  
password=123456 '  
promote_trigger_file = '/home/uxdb/uxdbinstall/dbsql/bin/slave_1/promote_trigger_file'
```

4. 启动备库。

```
./ux_ctl -D slave_1/ start
```

3.3.7. 流复制数据同步示例

1. 分别启动master，slave_1数据库，在master上创建数据库和表。

```
[uxdb@ux57 bin]$ ./uxsql
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# drop database testdb;
DROP DATABASE
uxdb=# \q
[uxdb@ux57 bin]$ ./uxsql
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# create database testdb;
CREATE DATABASE
uxdb=# \c testdb;
You are now connected to database "testdb" as user "uxdb"
testdb=# create table student (sno int,sname char(10));
CREATE TABLE
testdb=# insert into student values (1001,'JM');
INSERT 0 1
testdb=#
```

2. 在Salve_1上查询刚才创建的表和数据，判定是否有数据同步。

```
[uxdb@ux58 bin]$ ./uxsql
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# \c testdb;
You are now connected to database "testdb" as user "uxdb".
testdb=# select * from student;
   sno   |  sname
-----+-----
  1001   |   JM
(1 row)

testdb=#
```

3. 另外可以在主机上通过如下命令查询主备复制的状态。

select * from ux_stat_replication;

```
uxdb=# select * from ux_stat_replication ;
 pid | usesysid | username | application_name | client_addr | client_hostname | client_port |
----+-----+-----+-----+-----+-----+-----+
n  | replay_lsn | write_lag | flush_lag | replay_lag | sync_priority | sync_state
-----+-----+-----+-----+-----+-----+-----+
 6325 |      16384 | repuser | ux58           | 192.168.0.158 | ux58.com       |      55228 |
B0 | 0/120001B0 |         |               |              | 1 | sync
(1 row)

uxdb=#
```

4. 到此可见，从库已经同步了主库的数据，到此可以说UXDB流复制热备已经创建完成。

第 4 章 配置uxpool

uxpool的配置项很多，安装完成后有三个模板文件，都存放在安装目录的etc目录下，一般在配置之前，先拷贝这个模板文件，然后在此基础上进行配置。进入/home/uxdb/uxdbinstall/uxpool/etc目录，执行如下命令。

```
cp pool_hba.conf.sample pool_hba.conf
cp uxpath.conf.sample uxpath.conf
cp pcg.conf.sample pcg.conf
```

4.1. 配置uxpool.conf

1. uxpath默认只接受9999的本地连接，如果想从其他主机接受连接，请设置listen_addresses为“*”，在uxpool中相应配置修改为如下内容。

```
listen_addresses = '*'
port = 9999
```

2. 指定配置目录。

```
pid_file_name = '/home/uxdb/uxdbinstall/uxpool/uxpool.pid'
```

3. 配置后端数据库。

uxpool.conf中一项重要的配置就是配置后端数据库，后端数据库的配置示例如下所示。

```
backend_hostname0 = 'ux57.com'
backend_port0 = 5432
backend_weight0 = 1
```

```
backend_hostname1 = 'ux58.com'
backend_port1 = 5432
backend_weight1 = 1
```

4. 其他与uxpool工作模式相关的配置项，将在后面示例场景中介绍。

4.2. 配置pool_hba.conf

pool_hba.conf是对登录用户进行验证的，要和uxdb的ux_hba.conf保持一致，要么都是trust，要么都是md5验证方式，这里采用了md5验证方式，配置如下所示。

进入/home/uxdb/uxdbinstall/uxpool/etc/ 目录下，执行命令：vim pool_hba.conf ，修改trust为md5验证方式，如下图所示。

```
[uxdb@ux57 etc]$ cp pool_hba.conf.sample pool_hba.conf
[uxdb@ux57 etc]$ vim pool_hba.conf
[uxdb@ux57 etc]$
```

```
# TYPE DATABASE USER CIDR-ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all md5
# IPv4 local connections:
host all all 127.0.0.1/32 md5
host all all ::1/128 md5
host all all 0/0 md5
```

4.3. 配置pcp.conf

uxpool有一个用于管理的接口，名为PCP，可通过网络获取数据库的节点信息，要使用pcp命令必须进行用户认证。这种认证需要pcp.conf文件中定义一个用户名和密码。在这个文件中，用户名和密码成对地出现在每一行中，它们用冒号（:）隔开。密码的格式是用md5的哈希加密。如设置一个用户“uxdb”及密码“123456”时，会先计算密码的md5值，如下所示。

1. 使用uxdb用户进入/home/uxdb/uxdbinstall/uxpool/bin目录下，执行如下命令。

```
./ux_md5 123456
```

```
[uxdb@ux57 bin]$ ./ux_md5 123456
e10adc3949ba59abbe56e057f20f883e
```

2. 计算完md5后，进入/home/uxdb/uxdbinstall/uxpool/etc目录下pcp.conf文件中配置如下内容。

```
uxdb:e10adc3949ba59abbe56e057f20f883e
```

其中“e10adc3949ba59abbe56e057f20f883e”就是密码的md5值，如下图所示。

```
[uxdb@ux57 bin]$ vim pcp.conf
uxdb:e10adc3949ba59abbe56e057f20f883e
~
```

3. 执行命令。

```
../bin/ux_md5 -p -m -u uxdb pool_passwd
```

```
[uxdb@ux57 etc]$ ../bin/ux_md5 -p -m -u uxdb pool_passwd
password:
[uxdb@ux57 etc]$ ll
total 236
-rwxrwxrwx. 1 uxdb uxdb 894 Dec 17 11:35 pcp.conf
-rwxrwxrwx. 1 uxdb uxdb 856 Dec 17 11:20 pcp.conf.sample
-rwxrwxrwx. 1 uxdb uxdb 3244 Dec 17 11:20 pool_hba.conf
-rwxrwxrwx. 1 uxdb uxdb 3250 Dec 17 11:20 pool_hba.conf.sample
-rw-rw-r--. 1 uxdb uxdb 41 Dec 17 11:38 pool_passwd
-rwxrwxrwx. 1 uxdb uxdb 35856 Dec 17 11:20 uxpools.conf
-rwxrwxrwx. 1 uxdb uxdb 35831 Dec 17 11:20 uxpools.conf.sample
-rwxrwxrwx. 1 uxdb uxdb 35304 Dec 17 11:20 uxpools.conf.sample-logical
-rwxrwxrwx. 1 uxdb uxdb 35459 Dec 17 11:20 uxpools.conf.sample-master-slave
-rwxrwxrwx. 1 uxdb uxdb 35419 Dec 17 11:20 uxpools.conf.sample-replication
-rwxrwxrwx. 1 uxdb uxdb 35481 Dec 17 11:20 uxpools.conf.sample-stream
[uxdb@ux57 etc]$
```

第 5 章 管理uxpool

5.1. 启动uxpool

进入/home/uxdb/uxdbinstall/uxpool/bin目录下，执行如下命令。

```
cd /home/uxdb/uxdbinstall/uxpool/bin
mkdir -p /home/uxdb/uxdbinstall/uxpool/log
./uxpool -n -D > /home/uxdb/uxdbinstall/uxpool/log/uxpool.log 2>&1 &
```

参数含义，如下所示。

- -n: 表示日志打印到终端
- -d: 表示打印调试信息

```
[uxdb@ux57 uxpoo1]$ pwd
/home/uxdb/uxdbinstall/uxpool
[uxdb@ux57 uxpoo1]$ cd bin
[uxdb@ux57 bin]$ ./uxpool -n -d -D > /home/uxdb/uxdbinstall/uxpool/log/uxpool.log 2>&1 &
[1] 5824
[uxdb@ux57 bin]$ ps -ef|grep uxpoo1
uxdb      5824    2324    0 18:11 pts/1      00:00:00 ./uxpool -n -d -D
uxdb      5825    5824    0 18:11 pts/1      00:00:00 uxpoo1: watchdog
uxdb      5826    5824    0 18:11 pts/1      00:00:00 uxpoo1: lifecheck
uxdb      5827    5824    0 18:11 pts/1      00:00:00 uxpoo1: wait for connection request
```

5.2. 停止uxpool

- 停止uxpool，执行如下命令。

```
uxpool stop
```

- 快速终止uxpool，执行如下命令。

```
./uxpool -m fast stop
```

5.3. 限制最大连接数

- 通过uxpool连接进数据库的个数不能超过 `num_init_children * max_pool` 个，这两个参数存放于/home/uxdb/uxdbinstall/uxpool/etc目录下uxpool.conf文件中，默认配置如下所示。

```
num_init_children = 32
max_pool = 4
```

- 根据默认配置，一个uxpool可以同时并发连接128个会话。配置时可以根据实际情况进行修改。

5.4. 复制和负载均衡管理

uxpool如果要使用复制和负载均衡管理模式，则需要进入uxpool的安装目录/home/uxdb/uxdbinstall/uxpool/etc目录下，修改uxpool.conf配置，如下所示。


```
replication_mode = on    #复制开关  
load_balance_mode = on  #负载均衡开关
```

如果是uxdb-2.1.1.0之后的版本还需要进行如下配置。

```
failover_when_quorum_exists = off
```

5.5. 查看uxpool版本号

查看uxpool版本号，如下命令所示。

```
show pool_version;
```

```
uxdb=# show pool_version;  
      pool_version  
-----  
3.7.1 (amefuriboshi)  
(1 row)  
  
uxdb=#
```

第 6 章 uxpool场景示例

6.1. 复制和负载均衡示例

先建一个简单的示例环境，如[表 6.1 “uxpool复制和负载均衡示例环境”](#)所示。

表 6.1. uxpool复制和负载均衡示例环境

主机名称	IP地址	uxpool	UXDBServer	实例名称	数据库名称
ux57.com	192.71.0.197	已安装	已安装	test01	uxdb
ux58.com	192.71.0.199	已安装	已安装	test02	uxdb

这个数据库实例是在操作系统用户“uxdb”下，uxpool安装在/home/uxdb/uxdbinstall/uxpool目录下，UXDBServer是安装在/home/uxdb/uxdbinstall/dbsql/bin目录下。

在ux57.com主机 /home/uxdb/uxdbinstall/dbsql/bin目录下执行如下命令。

```
./initdb -D test01 -W
```

在ux58.com主机 /home/uxdb/uxdbinstall/dbsql/bin目录下执行如下命令。

```
./initdb -D test02 -W
```

无需配置文件，进行下一步即可。

6.1.1. 复制模式配置

uxpool的配置项很多，安装完成后有三个模板文件，都存放在安装目录的etc目录下，一般在配置之前，先拷贝这个模板文件，然后在此基础上进行配置。配置方法为进入/home/uxdb/uxdbinstall/uxpool/etc目录，执行如下命令（ux57.com主机和ux58.com主机分别配置）。

```
cp pool_hba.conf.sample pool_hba.conf  
cp uxpool.conf.sample uxpool.conf  
cp pcp.conf.sample pcp.conf
```

1. pcp.conf配置

需要对ux57.com和ux58.com分别配置。

- uxpool有一个用于管理的接口，名为PCP,可通过网络获取数据库的节点信息。要使用pcp命令必须进行用户认证。这种认证需要pcp.conf文件中定义一个用户名和密码。在这个文件中，用户名和密码成对地出现在每一行中，它们用冒号（:）隔开。密码的格式是用md5的哈希加密。如设置一个用户“uxdb”及密码“123456”时，会先计算密码的md5值，如使用uxdb用户，进入/home/uxdb/uxdbinstall/uxpool/bin目录下，执行如下命令。

```
./ux_md5 123456
```

```
[uxdb@ux57 bin]$ ./ux_md5 123456  
e10adc3949ba59abbe56e057f20f883e
```

- 计算完md5后，编辑/home/uxdb/uxdbinstall/uxpool/etc目录下的pcp.conf文件，配置如下内容。

uxdb:e10adc3949ba59abbe56e057f20f883e

其中“e10adc3949ba59abbe56e057f20f883e”就是密码的md5值，如下图所示。

```
[uxdb@ux57 bin]$ vim pcp.conf
```

```
uxdb:e10adc3949ba59abbe56e057f20f883e
```

- 执行如下命令，结果如图所示。

../bin/ux_md5 -p -m -u uxdb pool_passwd

```
[uxdb@ux57 etc]$ ../bin/ux_md5 -p -m -u uxdb pool_passwd
password:
[uxdb@ux57 etc]$ ll
total 236
-rwxrwxrwx. 1 uxdb uxdb 894 Dec 17 11:35 pcp.conf
-rwxrwxrwx. 1 uxdb uxdb 856 Dec 17 11:20 pcp.conf.sample
-rwxrwxrwx. 1 uxdb uxdb 3244 Dec 17 11:20 pool_hba.conf
-rwxrwxrwx. 1 uxdb uxdb 3250 Dec 17 11:20 pool_hba.conf.sample
-rw-rw-r--. 1 uxdb uxdb 41 Dec 17 11:38 pool_passwd
-rwxrwxrwx. 1 uxdb uxdb 35856 Dec 17 11:20 uxpools.conf
-rwxrwxrwx. 1 uxdb uxdb 35831 Dec 17 11:20 uxpools.conf.sample
-rwxrwxrwx. 1 uxdb uxdb 35304 Dec 17 11:20 uxpools.conf.sample-logical
-rwxrwxrwx. 1 uxdb uxdb 35459 Dec 17 11:20 uxpools.conf.sample-master-slave
-rwxrwxrwx. 1 uxdb uxdb 35419 Dec 17 11:20 uxpools.conf.sample-replication
-rwxrwxrwx. 1 uxdb uxdb 35481 Dec 17 11:20 uxpools.conf.sample-stream
[uxdb@ux57 etc]$
```

2. pool_hba.conf配置

需要对ux57.com和ux58.com分别配置。

pool_hba.conf是对登录用户进行验证的，要和uxdb的ux_hba.conf保持一致，要么都是trust，要么都是md5验证方式，这里采用了md5验证方式如下设置（test01和test02分别配置），进入/home/uxdb/uxdbinstall/uxpool/etc目录，配置如下所示。

```
[uxdb@ux57 etc]$ cp pool_hba.conf.sample pool_hba.conf
[uxdb@ux57 etc]$ vim pool_hba.conf
[uxdb@ux57 etc]$
```

```
# TYPE DATABASE USER CIDR-ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all md5
# IPv4 local connections:
host all all 127.0.0.1/32 md5
host all all ::1/128 md5
host all all 0/0 md5
```

3. uxpath.conf配置

需要对ux57.com和ux58.com分别配置。

- 编辑/home/uxdb/uxdbinstall/uxpool/etc目录下的uxpool.conf文件。

```
[uxdb@ux57 etc]$ cp uxpath.conf.sample uxpath.conf
[uxdb@ux57 etc]$ vim uxpath.conf
```

- uxpath.conf配置

需要对ux57.com和ux58.com分别配置。

- uxpath Connection Settings -

listen_addresses = '*'

port = 9999

pcp_listen_addresses = '*'

pcp_port = 9898

- Backend Connection Settings -

#每一个uxpool节点配置一组backend_* 指定集群host和端口号

backend_hostname0 = 'ux57.com'

backend_port0 = 5432

backend_weight0 = 1

backend_data_directory0 = '/home/uxdb/uxdbinstall/dbsql/bin/test01' #集群数据目录

backend_flag0 = 'ALLOW_TO_FAILOVER'

backend_hostname1 = 'ux58.com'

backend_port1 = 5432

backend_weight1 = 1

backend_data_directory1 = '/home/uxdb/uxdbinstall/dbsql/bin/test02' #集群数据目录

backend_flag1 = 'ALLOW_TO_FAILOVER'

- Authentication -

enable_pool_hba = on

pool_passwd = 'pool_passwd' #pool_passwd文件名

pid_file_name = '/home/uxdb/uxdbinstall/uxpool/uxpool.pid'

logdir = '/home/uxdb/uxdbinstall/uxpool/log'

replication_mode = on #复制开关

load_balance_mode = on #负载均衡开关

- Streaming -

sr_check_period = 5

sr_check_user = 'repuser'

sr_check_password = '123456'

sr_check_database = 'uxdb'

#-----

HEALTH CHECK GLOBAL PARAMETERS

#-----

health_check_period = 10 #检查的周期

```

health_check_timeout = 20 #指定做健康检查的超时时间,
health_check_user = 'uxdb' #指定健康检查连接后端数据库的用户。

health_check_password = '123456' #指定做健康检查时, 连接后端数据库的密码。
health_check_database = 'uxdb'

health_check_period0 = 10
health_check_timeout0 = 20
health_check_user0 = 'uxdb'
health_check_password0 = '123456'
health_check_database0 = 'uxdb'

# - Enabling -
use_watchdog = on

# - Watchdog communication Settings -
# 指定当前配置uxpool节点的hostname
wd_hostname = 'ux57.com'

wd_port = 9000

delegate_IP = 'uc60.com' #虚拟IP配置

if_cmd_path = '/sbin'

if_up_cmd = 'ifconfig ens33:0 inet $_IP_$ netmask 255.255.254.0'

if_down_cmd = 'ifconfig ens33:0 down'
arping_cmd = 'arping -fqbDUV -w 1 $_IP_$'

wd_heartbeat_port = 9694
wd_heartbeat_keepalive = 2
wd_heartbeat_deadtime = 30
# 心跳检测目标uxpool节点, 除过自身外有多少个uxpool节点, 就写几组配置
heartbeat_destination0 = 'ux58.com'
heartbeat_destination_port0 = 9694
heartbeat_device0 = 'ens33'

#其他的uxpool节点host位置, uxpoo1节点端口号和看门狗的端口号, 配备有多少个
#uxpool节点就写多少组
other_uxpool_hostname0 = 'ux58.com' #对端

other_uxpool_port0 = 9999
other_wd_port0 = 9000

如果是uxdb-2.1.1.0之后的版本还需要进行如下配置。

failover_when_quorum_exists = off

```

4. 权限配置

如果两台机器之间为主备关系, 用于主机宕机后的主备切换时。

```

[root@ux57 ~]# chmod u+s /sbin/ifconfig
[root@ux57 ~]# chmod u+s /usr/sbin/*

```

5. 创建uxpool日志文件

需要对ux57.com和ux58.com分别创建。

进入/home/uxdb/uxdbinstall/uxpool目录下，创建目录。

mkdir log

```
[uxdb@ux57 uxpoo1]$ pwd
/home/uxdb/uxdbinstall/uxpool
[uxdb@ux57 uxpoo1]$ ll
total 12
drwxr--r--. 2 uxdb uxdb 4096 Sep 18 19:29 bin
drwxr-xr-x. 2 uxdb uxdb 4096 Sep 20 18:00 etc
-rwxrwxrwx. 1 uxdb uxdb 371 Sep 19 18:34 failover_stream.sh
drwxr-xr-x. 2 uxdb uxdb 90 Sep 18 11:17 include
drwxr-xr-x. 2 uxdb uxdb 58 Sep 19 19:00 lib
drwxr-xr-x. 3 uxdb uxdb 23 Sep 18 11:17 share
[uxdb@ux57 uxpoo1]$ mkdir log
[uxdb@ux57 uxpoo1]$ ls
bin etc failover_stream.sh include lib log share
[uxdb@ux57 uxpoo1]$
```

6. 启动uxpool

需要对ux57.com和ux58.com分别配置。

进入/home/uxdb/uxdbinstall/uxpool/bin目录下，执行命令。

lsof -i:9000

检测目标端口无占用，若有则kill掉或者选择其他端口（从配置文件修改）。

启动成功，结果如下所示。

./uxpool -n -D > /home/uxdb/uxdbinstall/uxpool/log/uxpool.log 2>&1 &

```
[uxdb@localhost bin]$ lsof -i:9000
[uxdb@localhost bin]$ ./uxpool -n -D > /home/uxdb/uxdbinstall/uxpool/log/uxpool.log 2>
&1 &
[1] 126908
[uxdb@localhost bin]$ ps -ef|grep uxpoo1
uxdb 126908 12853 0 17:43 pts/8 00:00:00 ./uxpool -n -D
uxdb 126910 126908 0 17:43 pts/8 00:00:00 uxpoo1: watchdog
uxdb 126920 126908 0 17:43 pts/8 00:00:00 uxpoo1: lifeclock
uxdb 126921 126920 0 17:43 pts/8 00:00:00 uxpoo1: heartbeat receiver
uxdb 126922 126920 0 17:43 pts/8 00:00:00 uxpoo1: heartbeat sender
uxdb 126923 126908 0 17:43 pts/8 00:00:00 uxpoo1: wait for connection request
uxdb 126924 126908 0 17:43 pts/8 00:00:00 uxpoo1: wait for connection request
uxdb 126925 126908 0 17:43 pts/8 00:00:00 uxpoo1: wait for connection request
```

显示uxpool的运行状态成功之后在所有uxpool安装环境处执行ifconfig，查看选择建立虚拟ip地址网卡是否建立成功，所有节点处存在一处虚拟网卡即表示uxpool建立成功，否则重新检查配置。

```

ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.71.0.197 netmask 255.255.254.0 broadcast 192.71.1.255
    inet6 fe80::bb4c:3f42:ada5:ac4e prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:28:0e:b4 txqueuelen 1000 (Ethernet)
    RX packets 8186247 bytes 2228065068 (2.0 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3351297 bytes 2953547950 (2.7 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ens33:0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.71.0.200 netmask 255.255.254.0 broadcast 192.71.1.255
    ether 00:0c:29:28:0e:b4 txqueuelen 1000 (Ethernet)

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1 (Local Loopback)
    RX packets 4010962 bytes 660936651 (630.3 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4010962 bytes 660936651 (630.3 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

6.1.2. 复制模式下的负载均衡

以上步骤执行完成并且uxpool启动成功之后执行继续进行以下步骤。

使用虚拟IP连接uxpool的端口9999，插入数据，发现两边数据库的数据相同，实现了负载均衡，情况模拟如下所示。

1. 进入ux57.com主机 /home/uxdb/uxdbinstall/dbsql/bin目录下，执行如下命令。

```

//运行基准测试
./uxbench -i -h uc60.com -p 9999
[uxdb@ux57 bin]$ ./uxbench -i -h uc60.com -p 9999

```

```

[uxdb@ux57 bin]$ ./uxbench -i -h uc60.com -p 9999
Password:
DEBUG: pool_write: to frontend: kind:N po:0
DEBUG: selecting load balance node
DETAIL: selected backend id is 1
dropping old tables...
creating tables...
generating data...
100000 of 100000 tuples (100%) done (elapsed 0.03 s, remaining 0.00 s)
vacuuming...
creating primary keys...
done.
[uxdb@ux57 bin]$

```

uxsql通过虚拟ip和端口连接数据库。

```

[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
uxdb=# \d

```

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password for user uxdb:
DEBUG: pool_write: frontend: N po:0
DEBUG: selecting load balance node
DETAIL: selected backend id is 0
uxsql (2.1.1.4)
Type "help" for help.

uxdb=# \d
               List of relations
 Schema |      Name      | Type | Owner
-----+-----+-----+-----
 public | uxbench_accounts | table | uxdb
 public | uxbench_branches | table | uxdb
 public | uxbench_history  | table | uxdb
 public | uxbench_tellers  | table | uxdb
(4 rows)

uxdb=#
```

2. 进入ux58.com主机 /home/uxdb/uxdbinstall/dbsql/bin目录下，查看数据，发现数据相同，实现了负载均衡。

```
[uxdb@ux58 bin]$ ./uxsql -h uc60.com -p 9999
Password for user uxdb:
DEBUG: pool_write: to frontend: kind:N po:0
DEBUG: selecting load balance node
DETAIL: selected backend id is 0
uxsql (2.1.1.4)
Type "help" for help.

uxdb=# \d
               List of relations
 Schema |      Name      | Type | Owner
-----+-----+-----+-----
 public | uxbench_accounts | table | uxdb
 public | uxbench_branches | table | uxdb
 public | uxbench_history  | table | uxdb
 public | uxbench_tellers  | table | uxdb
(4 rows)
```

6.1.3. 复制模式下的高可用

模拟复制模式下，一台机器宕机后，两台机器分别通过虚拟IP连接uxpool时，可以正常访问，且插入数据后，两边的数据库数据相同，实现了复制功能，具体操作步骤如下所示。

1. ux57.com机器，进入/home/uxdb/uxdbinstall/dbsql/bin目录下，执行如下命令。

`./ux_ctl -D test01 stop`

```
[uxdb@ux57 bin]$ ./ux_ctl -D uxpath_test01/ stop
waiting for server to shut down.... done
server stopped
```

2. 使用外网虚拟IP继续访问。

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=#
```

3. 插入数据。

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# create table test01 (a int,b char(10));
CREATE TABLE
uxdb=# insert into test01 values (12,'zhangsan');
INSERT 0 1
uxdb=# insert into test01 values (13,'lisi');
INSERT 0 1
uxdb=#
```

4. 在ux58上查看，数据相同。

```
[uxdb@ux58 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# \d

               List of relations
 Schema |          Name          | Type  | Owner
-----+-----+-----+-----
 public | test                   | table | uxdb
 public | test01                  | table | uxdb
 public | test03                  | table | uxdb
 public | test04                  | table | uxdb
 public | uxbench_accounts       | table | uxdb
 public | uxbench_branches       | table | uxdb
 public | uxbench_history        | table | uxdb
 public | uxbench_tellers        | table | uxdb
(8 rows)

uxdb=# select * from test01;
 a |      b
---+-----
 12 | zhangsan
 13 | lisi
(2 rows)

uxdb=#
```

6.1.4. 复制模式下的宕机恢复

1. 在任意节点下，使用uxsql连接虚拟ip集群。查看正在运行的节点状态和集群地址。

```
./uxsql -h uc60.com -p 9999
show data_directory;
show pool_nodes;
```

```
uxdb=# show pool_nodes;
 node_id | hostname | port | status | lb_weight | role  | select_cnt | load_balance_node | replication_delay | replication_state | replication_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
 0       | ux57.com | 5432 | up      | 0.5000000 | primary | 44         | true              | 0                | 0                | 0                | 2021-11-26 18:36:27
 1       | ux58.com | 5432 | down    | 0.5000000 | standby | 19         | false             | 0                | 0                | 0                | 2021-11-25 18:02:52
(2 rows)
```

发现ux58.com ip地址下的集群显示down，即宕机状态。

```
[uxdb@ux58 bin]$ ./uxsql -h uc60.com -p 9999
Password for user uxdb:
DEBUG: pool_write: to frontend: kind:N po:0
DEBUG: selecting load balance node
DETAIL: selected backend id is 0
uxsql (2.1.1.4)
Type "help" for help.

uxdb=# show data_directory;
          data_directory
-----
/home/uxdb/uxdbinstall/dbsql/bin/test01
(1 row)
```

可以看到正在运行节点ip地址为ux57.com。

位置为 /home/uxdb/uxdbinstall/dbsql/bin/test01,

2. 进入test01集群所在bin目录下，使用./uxsql连接到test01集群。

创建复制用户，如下所示。

```
create user repuser replication LOGIN CONNECTION LIMIT 3 ENCRYPTED PASSWORD
'123456';
```

```
[uxdb@ux57 bin]$ ./uxsql
Password for user uxdb:
uxsql (2.1.1.4)
Type "help" for help.

uxdb=# create user repuser replication LOGIN CONNECTION LIMIT 3 ENCRYPTED PASSWORD '123456';
CREATE ROLE
uxdb=#
```

3. 修改正常运行状态下的集群目录下（test01）ux_hba.conf文件。

```
# replication privilege.
local  replication  all                                     md5
host   replication  all          127.0.0.1/32          md5
host   replication  all          ::1/128             md5
host   replication  repuser      ux58.com            md5
```

在最后一行加入repuser的复制权限和宕机集群的ip，然后重启当前运行中的集群（test01）。

```
./ux_ctl -D test01 restart
```

然后进入/home/uxdb/uxdbinstall/uxpool/bin执行。

```
./pcp_attach_node -d -U uxdb -h uc60.com -p 9898 -n 0
```

其中0表示pool_node的0节点为test01，将test01节点绑定到uxpool中。

```
[uxdb@localhost bin]$ pwd
/home/uxdb/uxdbinstall/uxpool/bin
[uxdb@localhost bin]$ ./pcp_attach_node -d -U uxdb -h uc60.com -p 9898 -n 0
Password:
DEBUG: recv: tos="N", len=91
BACKEND DEBUG: received PCP packet
DETAIL: EBUG
DEBUG: recv: tos="N", len=72
BACKEND DEBUG: PCP: sending md5 salt to client
DETAIL: EBUG
```

4. 进入宕机集群bin目录下（ip为ux58.com，/home/uxdb/uxdbinstall/dbsql/bin），删除原集群数据文件rm -rf test02。

执行ux_basebackup备份命令，如下所示。

./ux_basebackup -D test02 -Fp -Xs -v -P -h ux57.com -p 5432 -U repuser -W

```
[uxdb@ux58 bin]$ rm -rf test02
[uxdb@ux58 bin]$ ./ux_basebackup -D test02 -Fp -Xs -v -P -h ux57.com -p 5432 -U repuser -W
Password:
ux_basebackup: initiating base backup, waiting for checkpoint to complete
ux_basebackup: checkpoint completed
ux_basebackup: write-ahead log start point: 0/10000028 on timeline 1
ux_basebackup: starting background WAL receiver
ux_basebackup: created temporary replication slot "ux_basebackup_85409"
53374/53374 kB (100%), 1/1 tablespace
ux_basebackup: write-ahead log end point: 0/10000100
ux_basebackup: waiting for background process to finish streaming ...
ux_basebackup: syncing data to disk ...
ux_basebackup: base backup completed
[uxdb@ux58 bin]$
```

5. 备份成功之后，启动集群。

./ux_ctl -D test02 start

然后进入目录(ux58.com)/home/uxdb/uxdbinstall/uxpool/bin，执行如下命令。

./pcp_attach_node -d -U uxdb -h uc60.com -p 9898 -n 1

将集群节点绑定即可，节点1为test02集群。

```
[uxdb@ux58 bin]$ ./pcp_attach_node -d -U uxdb -h uc60.com -p 9898 -n 1
Password:
DEBUG: recv: tos="N", len=91
BACKEND DEBUG: received PCP packet
DETAIL: EBUG
DEBUG: recv: tos="N", len=72
BACKEND DEBUG: PCP: sending md5 salt to client
DETAIL: EBUG
DEBUG: recv: tos="m", len=8
DEBUG: recv: tos="N", len=91
BACKEND DEBUG: received PCP packet
DETAIL: EBUG
DEBUG: recv: tos="r", len=21
```

6. 使用如下命令查看两个节点的运行状态，宕机恢复成功。

```
./uxsql -h uc60.com -p 9999
```

```

uxdb=# show pool_nodes;
 node_id | hostname | port | status | lb_weight | role | select_cnt | load_balance_node | replication_delay | replication_state | replication_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
 0       | ux57.com | 5432 | up     | 0.500000 | primary | 34         | false             | 0                |                    |                    |                    | 2021-11-25 16:53:09
 1       | ux58.com | 5432 | up     | 0.500000 | standby | 12         | true              | 504              |                    |                    |                    | 2021-11-25 16:53:06
(2 rows)

```

6.2. 使用流复制的主备切换场景示例

先建一个简单的示例环境，此环境如[表 6.2 “uxpool复制和负载均衡示例环境”](#)所示。

表 6.2. uxpath复制和负载均衡示例环境

主机名称	IP地址	uxpool	UXDBServer	实例名称	数据库名称
ux57.com	192.168.0.157	已安装	已安装	master	uxdb
ux58.com	192.168.0.158	已安装	已安装	slave_1	uxdb

6.2.1. 主备模式配置

主数据与从数据之间同步的方式是使用同步的流复制关系，搭建流复制的具体方法请参见[第 3.3 节 “流复制”](#)。

uxpool的配置项很多，安装完成后有三个模板文件，都存放在安装目录的etc目录下，一般在配置之前，先拷贝这个模板文件，然后在此基础上进行配置。配置方法为进入/home/uxdb/uxpool/etc目录，执行如下命令（master和slave_1分别配置）。

```

cp pool_hba.conf.sample pool_hba.conf
cp uxpath.conf.sample uxpath.conf
cp pcp.conf.sample pcp.conf

```

在主备模式下，配置uxpool.conf中的参数“replication_mode”必须设置为“off”，而参数“master_slave_mode”必须设置为“on”，因为使用的是流复制的主备模式，所以“master_slave_sub_mode”要设置为“stream”；在主备模式下可以使用负载均衡，所以把“load_balance_mode”参数设置为“on”，配置如下所示。

1. pcp.conf配置

需要对master和slave_1分别配置。

- uxpool有一个用于管理的接口，名为PCP，可通过网络获取数据库的节点信息。要使用pcp命令必须进行用户认证。这种认证需要pcp.conf文件中定义一个用户名和密码。在这个文件中，用户名和密码成对地出现在每一行中，它们用冒号（:）隔开。密码的格式是用md5的哈希加密。如设置一个用户“uxdb”及密码“123456”时，会先计算密码的md5值，如使用uxdb用户，分别进入ux57.com和ux58.com的/home/uxdb/uxdbinstall/uxpool/bin目录下，执行如下命令。

```
./ux_md5 123456
```

```

[uxdb@ux57 bin]$ ./ux_md5 123456
e10adc3949ba59abbe56e057f20f883e

```

- 计算完md5后，在/home/uxdb/uxdbinstall/uxpool/etc目录下的pcp.conf文件中配置如下内容。

uxdb:e10adc3949ba59abbe56e057f20f883e

其中“e10adc3949ba59abbe56e057f20f883e”就是密码的md5值，如下图所示。

```
[uxdb@ux57 bin]$ vim pcp.conf
```

```
uxdb:e10adc3949ba59abbe56e057f20f883e
```

- 执行命令 `../bin/ux_md5 -p -m -u uxdb pool_passwd` 如下图所示。

```
[uxdb@ux57 etc]$ ../bin/ux_md5 -p -m -u uxdb pool_passwd
password:
[uxdb@ux57 etc]$ ll
total 236
-rwxrwxrwx. 1 uxdb uxdb 894 Dec 17 11:35 pcp.conf
-rwxrwxrwx. 1 uxdb uxdb 856 Dec 17 11:20 pcp.conf.sample
-rwxrwxrwx. 1 uxdb uxdb 3244 Dec 17 11:20 pool_hba.conf
-rwxrwxrwx. 1 uxdb uxdb 3250 Dec 17 11:20 pool_hba.conf.sample
-rw-rw-r--. 1 uxdb uxdb 41 Dec 17 11:38 pool_passwd
-rwxrwxrwx. 1 uxdb uxdb 35856 Dec 17 11:20 uxpools.conf
-rwxrwxrwx. 1 uxdb uxdb 35831 Dec 17 11:20 uxpools.conf.sample
-rwxrwxrwx. 1 uxdb uxdb 35304 Dec 17 11:20 uxpools.conf.sample-logical
-rwxrwxrwx. 1 uxdb uxdb 35459 Dec 17 11:20 uxpools.conf.sample-master-slave
-rwxrwxrwx. 1 uxdb uxdb 35419 Dec 17 11:20 uxpools.conf.sample-replication
-rwxrwxrwx. 1 uxdb uxdb 35481 Dec 17 11:20 uxpools.conf.sample-stream
[uxdb@ux57 etc]$
```

2. pool_hba.conf配置

需要对master和slave_1分别配置。

pool_hba.conf是对登录用户进行验证的，要和uxdb的ux_hba.conf保持一致，要么都是trust，要么都是md5验证方式，这里采用了md5验证方式，进入/home/uxdb/uxdbinstall/uxpool/etc目录，配置如下所示。

```
[uxdb@ux57 etc]$ cp pool_hba.conf.sample pool_hba.conf
[uxdb@ux57 etc]$ vim pool_hba.conf
[uxdb@ux57 etc]$
```

```
# TYPE DATABASE USER CIDR-ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all md5
# IPv4 local connections:
host all all 127.0.0.1/32 md5
host all all ::1/128 md5
host all all 0/0 md5
```

3. uxpools.conf配置

需要对master和slave_1分别配置。

编辑/home/uxdb/uxdbinstall/uxpool/etc目录下的uxpools.conf文件。

```
[uxdb@ux57 etc]$ cp uxpath.conf.sample uxpath.conf
[uxdb@ux57 etc]$ vim uxpath.conf
```

如果只部署主备两个uxpool节点，请在配置文件中添加**enable_consensus_with_half_votes = on**。

- master端配置，如下所示。

```
# - uxpath Connection Settings -
listen_addresses = '*'
port = 9999
pcp_listen_addresses = '*'
pcp_port = 9898

# - Backend Connection Settings -

backend_hostname0 = 'ux57.com' #主机master的hostname或者IP
backend_port0 = 5432
backend_weight0 = 1
backend_data_directory0 = '/home/uxdb/uxdbinstall/dbsql/bin/master' #集群数据目录
backend_flag0 = 'ALLOW_TO_FAILOVER'

backend_hostname1 = 'ux58.com'
backend_port1 = 5432
backend_weight1 = 1
backend_data_directory1 = '/home/uxdb/uxdbinstall/dbsql/bin/slave_1' #集群数据目录
backend_flag1 = 'ALLOW_TO_FAILOVER'

# - Authentication -

enable_pool_hba = on
pool_passwd = 'pool_passwd' #pool_passwd文件名

pid_file_name = '/home/uxdb/uxdbinstall/uxpool/uxpool.pid'
logdir = '/home/uxdb/uxdbinstall/uxpool/log'

replication_mode = off #复制开关
load_balance_mode = on #负载均衡开关
master_slave_mode = on
master_slave_sub_mode = 'stream'

# - Streaming -

sr_check_period = 5
sr_check_user = 'repuser'
sr_check_password = '123456'
sr_check_database = 'uxdb'

#-----
# HEALTH CHECK GLOBAL PARAMETERS
#-----

health_check_period = 10 #检查的周期
health_check_timeout = 20 #指定做健康检查的超时时间，
```

```
health_check_user = 'uxdb' #指定健康检查连接后端数据库的用户。
health_check_password = '123456' #指定做健康检查时，连接后端数据库的密码。
health_check_database = 'uxdb'
```

```
health_check_period0 = 10
health_check_timeout0 = 20
health_check_user0 = 'uxdb'
health_check_password0 = '123456'
health_check_database0 = 'uxdb'
```

#必须设置，否则primary数据库down了，uxpool不知道，不能及时切换。从库流复制还在连接数据，报连接失败。#只有下次使用uxpool登录时，发现连接不上，然后报错，这时候，才知道挂了，uxpool进行切换。

```
failover_command = '/home/uxdb/uxdbinstall/uxpool/failover_stream.sh %d %h %p %D %m
%H %M %P %r %R %N %S'
```

```
# - Enabling -
use_watchdog = on
```

```
# - Watchdog communication Settings -
wd_hostname = 'ux57.com'
```

```
wd_port = 9000
```

```
delegate_IP = 'uc60.com' #虚拟IP配置
```

```
if_cmd_path = '/sbin'
```

```
if_up_cmd = 'ifconfig ens33:0 inet $_IP_$ netmask 255.255.254.0'
```

```
if_down_cmd = 'ifconfig ens33:0 down'
```

```
arping_cmd = 'arping -fqbdUV -w 1 $_IP_$'
```

```
wd_heartbeat_port = 9694
wd_heartbeat_keepalive = 2
wd_heartbeat_deadtime = 30
heartbeat_destination0 = 'ux58.com'
heartbeat_destination_port0 = 9694
heartbeat_device0 = 'ens33'
```

```
other_uxpool_hostname0 = 'ux58.com' #对端
other_uxpool_port0 = 9999
other_wd_port0 = 9000
```

如果是uxdb-2.1.1.0之后的版本还需要进行如下配置。

```
failover_when_quorum_exists = off
```

- Slave_1节点配置（该节点的配置与master节点基本相同，此处仅将配置不同的参数列出来）。

```
wd_hostname = 'ux58.com' #本端
```



```
heartbeat_destination0 = 'ux57.com'  #对端
other_uxpool_hostname0 = 'ux57.com'  #对端
```

4. 创建主备切换脚本

master和slave_1分别创建。

uxpool.conf配置文件里，故障处理配置的是failover_command = '/home/uxdb/uxdbinstall/uxpool/failover_stream.sh %d %h %p %D %m %H %M %P %r %R %N %S'，因此，需要在/home/uxdb/uxdbinstall/uxpool/目录中创建failover_stream.sh脚本。

使用uxdb用户，进入/home/uxdb/uxdbinstall/uxpool目录下，创建文件：touch failover_stream.sh，编辑文件 vi failover_stream.sh。脚本内容如下所示。

```
#!/bin/bash
# This script is run by failover_command.

set -o xtrace
#exec >>(logger -i -p local1.info) 2>&1
logger -i -p local1.info failover_command begin
# Special values:
# %d = failed node id
# %h = failed node hostname
# %p = failed node port number
# %D = failed node database cluster path
# %m = new master node id
# %H = new master node hostname
# %M = old master node id
# %P = old primary node id
# %r = new master port number
# %R = new master database cluster path
# %N = old primary node hostname
# %S = old primary node port number
# %% = '%' character

FAILED_NODE_ID="$1"
FAILED_NODE_HOST="$2"
FAILED_NODE_PORT="$3"
FAILED_NODE_UXDATA="$4"
NEW_MASTER_NODE_ID="$5"
NEW_MASTER_NODE_HOST="$6"
OLD_MASTER_NODE_ID="$7"
OLD_PRIMARY_NODE_ID="$8"
NEW_MASTER_NODE_PORT="$9"
NEW_MASTER_NODE_UXDATA="${10}"
OLD_PRIMARY_NODE_HOST="${11}"
OLD_PRIMARY_NODE_PORT="${12}"

UXDBHOME=/home/uxdb/uxdbinstall/dbsql  #数据库目录

logger -i -p local1.info failover_stream.sh: start: failed_node_id=$FAILED_NODE_ID
old_primary_node_id=$OLD_PRIMARY_NODE_ID failed_host=$FAILED_NODE_HOST
new_master_host=$NEW_MASTER_NODE_HOST

## If there's no master node anymore, skip failover.
```

```

if [ $NEW_MASTER_NODE_ID -lt 0 ]; then
    logger -i -p local1.info failover_stream.sh: All nodes are down. Skipping failover.
    exit 0
fi

## Test passwordless SSH
ssh -T -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null uxdb@
${NEW_MASTER_NODE_HOST} -i ~/.ssh/id_rsa ls /tmp > /dev/null

if [ $? -ne 0 ]; then
    logger -i -p local1.info failover_stream.sh: passwordless SSH to uxdb@
    ${NEW_MASTER_NODE_HOST} failed. Please setup passwordless SSH.
    exit 1
fi

## If Standby node is down, skip failover.
if [ $FAILED_NODE_ID -ne $OLD_PRIMARY_NODE_ID ]; then
    logger -i -p local1.info failover_stream.sh: Standby node is down. Skipping failover.

# 默认是uxdb数据库uxdb用户，非默认时需要加上-U和-d参数指明
ssh -T -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null uxdb@
$OLD_PRIMARY_NODE_HOST -i ~/.ssh/id_rsa "
    ${UXDBHOME}/bin/uxsql -p $OLD_PRIMARY_NODE_PORT -c \"SELECT
ux_drop_replication_slot('${FAILED_NODE_HOST}')\"
"

    if [ $? -ne 0 ]; then
        logger -i -p local1.error failover_stream.sh: drop replication slot "${FAILED_NODE_HOST}"
        failed
        exit 1
    fi

    exit 0
fi

## Promote Standby node.
logger -i -p local1.info failover_stream.sh: Primary node is down, promote standby node
${NEW_MASTER_NODE_HOST}.

ssh -T -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null \
    uxdb@${NEW_MASTER_NODE_HOST} -i ~/.ssh/id_rsa ${UXDBHOME}/bin/ux_ctl -D
    ${NEW_MASTER_NODE_UXDATA} -w promote

if [ $? -ne 0 ]; then
    logger -i -p local1.error failover_stream.sh: new_master_host=$NEW_MASTER_NODE_HOST
    promote failed
    exit 1
fi

logger -i -p local1.info failover_stream.sh: end: new_master_node_id=
$NEW_MASTER_NODE_ID started as the primary node
exit 0

```

分别赋予master和slave_1主机 failover_stream.sh脚本可执行权限。

chmod 777 /home/uxdb/uxdbinstall/uxpool/failover_stream.sh

5. 创建uxpool日志文件

master和slave_1分别配置。

进入/home/uxdb/uxdbinstall/uxpool目录下，执行如下命令。

mkdir log

```
[uxdb@ux57 uxpath]$ pwd
/home/uxdb/uxdbinstall/uxpool
[uxdb@ux57 uxpath]$ ll
total 12
drwxr--r--. 2 uxdb uxdb 4096 Sep 18 19:29 bin
drwxr-xr-x. 2 uxdb uxdb 4096 Sep 20 18:00 etc
-rwxrwxrwx. 1 uxdb uxdb 371 Sep 19 18:34 failover_stream.sh
drwxr-xr-x. 2 uxdb uxdb 90 Sep 18 11:17 include
drwxr-xr-x. 2 uxdb uxdb 58 Sep 19 19:00 lib
drwxr-xr-x. 3 uxdb uxdb 23 Sep 18 11:17 share
[uxdb@ux57 uxpath]$ mkdir log
[uxdb@ux57 uxpath]$ ls
bin etc failover_stream.sh include lib log share
[uxdb@ux57 uxpath]$
```

6. 权限配置

如果两台机器之间为主备关系，用于主机宕机后的主备切换时，使用root用户登录，执行如下命令。

[root@ux57 ~]# chmod u+s /sbin/ifconfig

[root@ux57 ~]# chmod u+s /usr/sbin/

7. 启动uxpool

master和slave_1分别启动。

进入/home/uxdb/uxdbinstall/uxpool/bin目录下，执行如下命令。

./uxpool -n -D > /home/uxdb/uxdbinstall/uxpool/log/uxpool.log 2>&1 &

6.2.2. 主备切换

配置了主备模式之后，如果master机器宕机后，即停止当前主节点服务，然后稍等几秒钟，重新连接虚拟节点uc60，发现连接成功，查看主机状态，就会发现，主备发生切换，原来的主机成为备机，原来的备机成为了主机。宕机后的情况模拟如下所示。

1. 进入/home/uxdb/uxdbinstall/dbsql/bin目录下，执行如下命令。

[uxdb@ux57 bin]\$./ux_ctl -D master/ stop

```
[uxdb@ux57 bin]$ ./ux_ctl -D master/ stop
waiting for server to shut down.... done
server stopped
```

2. 稍等几分钟后，重新使用uxpool连接数据库查看主备关系，主备机切换成功。如下图所示。

进入/home/uxdb/uxdbinstall/dbsql/bin目录下，执行如下命令。

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
uxdb=# show pool_nodes;
```

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# show pool_nodes;
node_id | hostname | port | status | lb_weight | role | select_cnt |
+-----+-----+-----+-----+-----+-----+-----+
0       | ux57.com | 5432 | down   | 0.500000 | standby | 0           |
1       | ux58.com | 5432 | up     | 0.500000 | primary | 0           |
(2 rows)

uxdb=# █
```

3. 主备切换后，如果原来宕机的master机器想作为备机使用，则需要修复节点，修复方法如下所示。

- 同步数据时间线；

如果宕机后导致两个节点数据时间线不一致，需要做数据同步操作，同步master节点上的时间线。

```
./ux_rewind --target-uxdata=/home/uxdb/uxdbinstall/dbsql/bin/master --source-server='host=ux58.com port=5432 user=uxdb dbname=uxdb password=123456'
```

同步时间线时会遇到两种情况。

- 时间线无需同步，如下图所示。

```
[uxdb@ux57 bin]$ ./ux_rewind --target-uxdata=/home/uxdb/uxdbinstall/dbsql/bin/master --source-server='host=ux58.com
servers diverged at WAL location 0/32000098 on timeline 3
no rewind required
[uxdb@ux57 bin]$ cd master
[uxdb@ux57 master]$ ll
```

- 时间线有差异需要同步，则会按照新主库（原salve_1）的数据同步时间线，如下图所示。

```
[uxdb@ux57 bin]$ ./ux_rewind --target-uxdata=/home/uxdb/uxdbinstall/dbsql/bin/master --source-server='host=ux58.com port=5432 user=uxdb dbname=uxdb password=123456'
servers diverged at WAL location 0/E000000 on timeline 3
rewinding from last common checkpoint at 0/D000028 on timeline 3
Done!
[uxdb@ux57 bin]$ █
```

- 重命名recovery.done为recovery.conf，并修改primary_conninfo参数信息。

进入/home/uxdb/uxdbinstall/master目录，执行如下命令。

```
[uxdb@ux57 master]$ mv recovery.done recovery.conf
```

```
[uxdb@ux57 master]$ mv recovery.done recovery.conf
[uxdb@ux57 master]$ vi recovery.conf
[uxdb@ux57 master]$ cd ../
```

[uxdb@ux57 master]\$ vi recovery.conf，修改primary_conninfo参数如下。

```
primary_conninfo = 'application_name=ux57 host=ux58.com port=5432 user=repuser
password=123456'
trigger_file = '/home/uxdb/uxdbinstall/dbsql/bin/master/trigger_file'
```

注意

如果是uxdb-2.1.1.0之前的版本，需要进行该配置。

如果是uxdb-2.1.1.0及其之后的版本，不再适用该配置。

- 创建标志文件standby.signal，添加如下内容。

```
standby_mode = 'on'
```

注意

每次切换都需要创建该文件。

- 将master/uxsinodb.auto.conf下primary_conninfo中的host修改为新主端ip ux58.com。
- 修改ux_hba.conf参数信息。

进入/home/uxdb/uxdbinstall/dbsql/bin/master目录。

```
[uxdb@ux57 master]$ vi recovery.conf
[uxdb@ux57 master]$ vi ux_hba.conf
```

```
# "local" is for Unix domain socket connections only
local    all             all                                md5
# IPv4 local connections:
host     all             all                                0.0.0.0/0          md5
# IPv6 local connections:
host     all             all                                ::1/128           md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
local    replication     all                                md5
host     replication     all                                127.0.0.1/32      md5
host     replication     all                                ::1/128           md5
host     replication     repuser                        ux58.com           md5

"ux_hba.conf" 91L, 4320C
```

- 修复master节点重新加入集群。

启动master集群，并加入集群管理节点后，查询节点状态，节点状态正常，查询主备关系状态，主备关系状态正常。

```
[uxdb@ux57 bin]$ ./ux_ctl -D master/ restart
```

```
[uxdb@ux57 bin]$ ./ux_ctl -D master/ restart
ux_ctl: PID file "master/uxmaster.pid" does not exist
Is server running?
starting server anyway
waiting for server to start.....2018-10-12 19:19:07.21
2018-10-12 19:19:07.216 CST [10015] LOG:  listening on IPv6
2018-10-12 19:19:07.219 CST [10015] LOG:  listening on Unix
2018-10-12 19:19:07.253 CST [10015] LOG:  redirecting log ou
2018-10-12 19:19:07.253 CST [10015] HINT:  Future log output
uxmaster status starting
uxmaster status starting
uxmaster status starting
uxmaster status ready
done
server started
```

修复节点，如下所示。

```
[uxdb@ux57 bin]$ ./pcp_attach_node -d -U uxdb -h uc60.com -p 9898 -n 0
```

```
[uxdb@ux57 bin]$ ./pcp_attach_node -d -U uxdb -h uc60.com -p 9898 -n 0
Password:
DEBUG: recv: tos="m", len=8
DEBUG: recv: tos="r", len=21
DEBUG: send: tos="C", len=6
DEBUG: recv: tos="c", len=20
pcp_attach_node -- Command Successful
DEBUG: send: tos="X", len=4
[uxdb@ux57 bin]$
```

查看修复状态，如下所示。

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
uxdb=# show pool_nodes;
```

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password:
```

```
The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.
```

```
uxdb=# show pool_nodes;
```

node_id	hostname	port	status	lb_weight	role	select_cnt
0	ux57.com	5432	up	0.500000	standby	0
1	ux58.com	5432	up	0.500000	primary	0

(2 rows)

```
uxdb=#
```

6.2.3. 主备切换模式下的高可用

master端uxpool宕机后，停止master主机uxpool后，稍等片刻，继续访问uc60地址，可以访问成功，查看主机节点状态，节点状态正常。

1. 在ux57.com 主机上，进入/home/uxdb/uxdbinstall/uxpool/bin目录下，执行如下命令。

```
[uxdb@ux57 bin]$ ./uxpool -m fast stop
```

```
[uxdb@ux57 bin]$ ./uxpool -m fast stop
2018-10-12 19:27:21: pid 10352: LOG: stop request sent to uxpool. waiting
done.
[1]+ Done ./uxpool -n -D > /home/uxdb/uxdbinstall/ux
[uxdb@ux57 bin]$
```

2. 在ux57.com 主机上，数据库可以正常连接。

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
```

```
[uxdb@ux57 bin]$ ./uxpool -m fast stop
2018-10-12 19:27:21: pid 10352: LOG: stop request sent to uxpool. waiting
done.
[1]+ Done ./uxpool -n -D > /home/uxdb/uxdbinstall/ux
[uxdb@ux57 bin]$
```

3. 在ux58.com 主机上，数据库可以正常连接。

```
[uxdb@ux58 bin]$ ./uxsql -h uc60.com -p 9999
```

```
[uxdb@ux58 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# show pool_nodes;
 node_id | hostname | port | status | lb_weight | role   |
-----+-----+-----+-----+-----+-----+
    0    | ux57.com | 5432 | up     | 0.500000 | primary |
    1    | ux58.com | 5432 | up     | 0.500000 | standby |
(2 rows)

uxdb=#
```

4. 在ux57.com上插入数据。

```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# create table test01 (a int,b char(10));
CREATE TABLE
uxdb=# insert into test01 values (12,'zhangsan');
INSERT 0 1
uxdb=# insert into test01 values (13,'lisi');
INSERT 0 1
uxdb=#
```

5. 在ux58上查看，数据相同。


```
[uxdb@ux57 bin]$ ./uxsql -h uc60.com -p 9999
Password:

The license due date is: 2028-12-01 00:00:00
It's commercial license.
uxsql (10.0)
Type "help" for help.

uxdb=# create table test01 (a int,b char(10));
CREATE TABLE
uxdb=# insert into test01 values (12,'zhangsan');
INSERT 0 1
uxdb=# insert into test01 values (13,'lisi');
INSERT 0 1
uxdb=# █
```

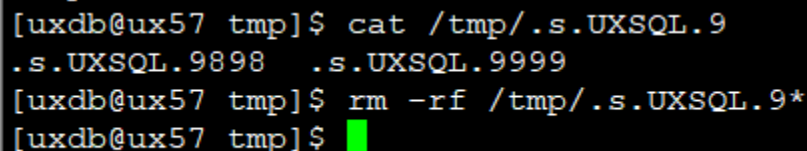
第 7 章 常见问题

7.1. uxpool服务停止

1. uxpool已经配置了主备切换场景之后，为了保证uxpool停止后，不影响主备切换，停止时要按照顺序停止，主备机uxpool停止->主备机集群停止。
2. 安装uxpool的主机关机、重启或异常终止服务后，uxpool想要再次启动时，需要先手动删除两个隐藏文件 /tmp/.s.UXSQL.9999 和 /tmp/.s.UXSQL.9898。删除此文件后，方可启动。如下图所示。

进入cd /tmp目录，执行如下命令。

```
rm -rf /tmp/.s.UXSQL.9*
```



```
[uxdb@ux57 tmp]$ cat /tmp/.s.UXSQL.9
.s.UXSQL.9898 .s.UXSQL.9999
[uxdb@ux57 tmp]$ rm -rf /tmp/.s.UXSQL.9*
[uxdb@ux57 tmp]$
```

7.2. 虚拟IP绑定情况

当主机使用虚拟IP进行访问uxpool时，如果同时启动两个以上的uxpool，则虚拟IP只会绑定在其中的一台机器上，当这台机器的uxpool停止后，虚拟IP自动解绑并自动绑定到另外一台机器上。如下图所示。

1. 同时启动了ux57和ux58两台主机上的uxpool，uxpool只显示绑定到其中的一台机器上，查看ux57上的IP绑定情况。

```
[uxdb@ux57 bin]$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.157 netmask 255.255.254.0 broadcast 192.168.1.255
    inet6 fe80::30c0:584c:ed84:fd91 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:76:23:db txqueuelen 1000 (Ethernet)
    RX packets 278978 bytes 25706833 (24.5 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 82828 bytes 26077013 (24.8 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 19 base 0x2000

ens33:0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.160 netmask 255.255.254.0 broadcast 192.168.1.255
    ether 00:0c:29:76:23:db txqueuelen 1000 (Ethernet)
    device interrupt 19 base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1 (Local Loopback)
    RX packets 62177 bytes 5771496 (5.5 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 62177 bytes 5771496 (5.5 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:19:92:e2 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2. 查看ux58上的IP绑定情况。

```
[uxdb@ux58 bin]$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.158 netmask 255.255.254.0 broadcast 192.168.1.255
    inet6 fe80::94c7:a5c1:428a:fc8a prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:46:30:2f txqueuelen 1000 (Ethernet)
    RX packets 289898 bytes 41353845 (39.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 68228 bytes 10245981 (9.7 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 19 base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1 (Local Loopback)
    RX packets 27643 bytes 2571974 (2.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 27643 bytes 2571974 (2.4 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:4c:90:10 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[uxdb@ux58 bin]$
```

注意

启动时要按照启动顺序启动，主机集群启动--->备机集群启动--->主机
uxpool启动--->备机uxpool启动。

第 8 章 参考

8.1. 服务器命令

8.1.1. uxpool

uxpool — uxpool主服务器。

8.1.1.1. 大纲

uxpool [*option...*]

uxpool [*option...*] stop

uxpool [*option...*] reload

8.1.1.2. 描述

uxpool以3种模式运行：启动、停止和重新加载。默认是启动模式。

8.1.1.3. 选项

这些是 3 种模式的常见选项。

-a hba_config_file

--hba-file= hba_config_file

设置pool_hba.conf配置文件的路径。如果文件放置在标准位置以外，则为必填项。

-f config_file

--config-file= config_file

设置uxpool.conf配置文件的路径。如果文件放置在标准位置以外，则为必填项。

-F pc_config_file

--pcp-file= pcp_config_file

设置pcp.conf配置文件的路径。如果文件放置在标准位置以外，则为必填项。

-k key_file

--key-file=key_file

设置.uxpoolkey文件的路径。如果使用AES256加密密码并且文件放置在标准位置之外并使用，则为必填项。

-h

--help

打印帮助。

8.1.1.4. 启动uxpool主服务器

以下是启动模式的选项。

-d
--debug

在调试模式下运行uxpool。产生了大量调试消息。

-n
--dont-detach

不要在守护进程模式下运行，不分离控制ttys。

-x
--debug-assertions

协助调试。

-C
--clear-oidmaps

当memqcache_method为memcached时清除查询缓存oidmaps。

如果memqcache_method是shmem，uxpool总是在启动时丢弃oidmap。所以这个选项不是必需的。

-D
--discard-status

丢弃uxpool_status文件并且不恢复以前的状态。

8.1.1.5. 停止uxpool主服务器

以下是停止模式的选项。

-m shutdown_mode
--mode=shutdown_mode

停止uxpool。shutdown_mode是smart、fast 或immediate。如果指定了smart，uxpool将等待所有客户端断开连接。如果指定了fast或immediate，uxpool 会立即停止自己，而无需等待所有客户端断开连接。在当前的实现中，fast和immediate之间没有区别。

8.1.1.6. 重新加载uxpool配置文件

重新加载uxpool的配置文件。重新加载模式不存在特定选项。通用选项适用。

8.2. PCP命令

本部分包含PCP命令的参考信息。PCP命令是通过网络操作uxpool的UNIX命令。

- PCP连接认证

PCP用户名和密码必须在\$prefix/etc目录下的pcp.conf中声明。如果pcp.conf放在其他位置，则可以在启动uxpool使用-F选项。

- PCP密码文件

如果没有为pcp连接指定密码，则用户主目录中的文件.pcppass或环境变量PCPPASSFILE引用的文件可以包含要使用的密码。

该文件应该包含如下内容。

```
hostname:port:username:password
```

（可以通过复制上面的行并在其前面加上#来向文件添加提醒注释。）前三个字段中的每一个都可以是文字值或 `*`，它可以匹配任何内容。将使用与当前连接参数匹配的第一行的密码字段。（因此，当使用通配符时，请首先放置更具体的条目。）如果条目需要包含 `:` 或 `\`，请使用 `\` 转义此字符。主机名 `localhost` 匹配来自本地机器的TCP（主机名 `localhost`）和Unix域套接字连接。

`.pcppass`上的权限必须禁止对世界或组的任何访问；通过命令 `chmod 0600 ~/.pcppass` 实现这一点。如果权限不那么严格，该文件将被忽略。

8.2.1. `pcp_common_options`

`pcp_common_options` -- PCP命令中使用的常用选项，

8.2.1.1. 大纲

```
pcp_command [option...]
```

8.2.1.2. 描述

有一些参数对所有PCP命令都是通用的。其中大部分用于身份验证，其余用于详细模式、调试消息等。

8.2.1.3. 选项

```
-h hostname
--host=hostname
```

运行服务器的机器的主机名。如果该值以斜杠开头，则它用作Unix域套接字的目录。

```
-p port
--port=port
```

PCP端口号（默认值：“9898”）。

```
-U username
--username=username
```

PCP认证的用户名（默认：OS 用户名）。

```
-w
--no-password
```

从不提示输入密码。如果 `.pcppass` 文件无法提供密码，则连接尝试将失败。此选项在没有用户输入密码的批处理任务和脚本中很有用。

```
-W
--password
```

强制密码提示（应该自动发生）。

`-d`

`--debug`

启用调试消息。

`-v`

`--verbose`

启用详细输出。

`-V`

`--version`

打印命令版本，然后退出。

`-?`

`--help`

显示命令行参数的帮助，然后退出。

8.2.1.4. 环境

`PCPPASSFILE`

指定pcp密码文件的路径。

8.2.2. `pcp_node_count`

`pcp_node_count --` 显示数据库节点总数，

8.2.2.1. 大纲

`pcp_node_count [option...]`

8.2.2.2. 描述

`pcp_node_count`显示`uxpool.conf` 中定义的数据库节点总数。它不区分节点状态（附加或者分离），统计所有节点。

8.2.2.3. 选项

请参见[pcp_common_options](#)。

8.2.2.4. 示例

如下示例所示。

```
$ pcp_node_count -p 11001
Password:
2
```

8.2.3. `pcp_node_info`

`pcp_node_info --` 显示给定节点ID的信息，

8.2.3.1. 大纲

```
pcp_node_info [option...] [node_id]
```

8.2.3.2. 描述

`pcp_node_info` 显示有关给定节点ID的信息。

8.2.3.3. 选项

```
-n node_id
--node-id=node_id
```

要获取信息的后端节点的索引。

Other options

请参见[pcp_common_options](#)。

8.2.3.4. 示例

如下示例所示。

```
$ pcp_node_info -h localhost -U uxdb 1
/tmp 11003 2 0.500000 up standby 0 streaming async 2019-04-23 13:58:40
```

结果按照以下顺序排列。

1. hostname
2. port number
3. status
4. load balance weight
5. status name
6. backend role
7. replication delay
8. replication state (taken from `ux_stat_replication`, if UXDB is 9.1 or later)
9. sync replication state (taken from `ux_stat_replication`, if UXDB is 9.2 or later)
10. last status change time0

状态由 [0 到 3] 之间的数字表示。要正确显示7、8、9，`sr_check_period`不能为 0。如果 `sr_check_user`不是UXDB超级用户，也不在“`ux_monitor`”组中，则不会显示8、9。

注意

为了使`sr_check_user`在`ux_monitor`组，通过执行以下SQL命令UXDB的超级用户（替换“`sr_check_user`”用的设置`sr_check_user`）。

```
GRANT ux_monitor TO sr_check_user;
```

- 0 - 此状态仅在初始化期间使用。PCP永远不会显示它。
- 1 - 节点已启动。还没有连接。

- 2 - 节点已启动。连接是池化的。
- 3 - 节点已关闭。

负载均衡权重以标准化格式显示。

该`--verbose`选项可以帮助理解输出。如下所示。

```
$ pcp_node_info --verbose -h localhost -U uxdb 1
Hostname      : /tmp
Port          : 11003
Status        : 2
Weight        : 0.500000
Status Name    : up
Role          : standby
Replication Delay : 0
Replication State : streaming
Replication Sync State : async
Last Status Change : 2019-04-23 13:58:40
```

8.2.4. pcp_watchdog_info

`pcp_watchdog_info` -- 显示uxpool的watchdog状态，

8.2.4.1. 大纲

```
pcp_watchdog_info [option...] [watchdog_id]
```

8.2.4.2. 描述

`pcp_node_info` 显示有关给定节点ID的信息。

8.2.4.3. 选项

```
-n watcgdog_id
--node-id=watcgdog_id
```

要获取信息的其他uxpool的索引。

索引 0 获取自己的watchdog信息。

如果省略则获取所有watchdog节点的信息

Other options

请参见[pcp_common_options](#)。

8.2.4.4. 示例

如下示例所示。

```
$ pcp_watchdog_info -h localhost -U uxdb
3 NO Linux_host1.localdomain_9991 host1
```

```
Linux_host1.localdomain_9991 host1 9991 9001 7 STANDBY
Linux_host2.localdomain_9992 host2 9992 9002 4 MASTER
Linux_host3.localdomain_9993 host3 9993 9003 7 STANDBY
```

结果按照以下顺序排列。

The first output line describes the watchdog cluster information:

1. Total watchdog nodes in the cluster
2. Is VIP is up on current node?
3. Master node name
4. Master node host

Next is the list of watchdog nodes:

1. node name
2. hostname
3. uxpools port
4. watchdog port
5. current node state
6. current node state name

该--verbose选项可以帮助理解输出。如下所示。

```
$ pcp_watchdog_info -h localhost -v -U uxdb
Watchdog Cluster Information
Total Nodes      : 3
Remote Nodes     : 2
Quorum state     : QUORUM EXIST
Alive Remote Nodes : 2
VIP up on local node : NO
Master Node Name  : Linux_host2.localdomain_9992
Master Host Name  : localhost
```

```
Watchdog Node Information
Node Name       : Linux_host1.localdomain_9991
Host Name       : host1
Delegate IP     : 192.168.1.10
Uxpools port    : 9991
Watchdog port   : 9001
Node priority   : 1
Status          : 7
Status Name     : STANDBY
```

```
Node Name       : Linux_host2.localdomain_9992
Host Name       : host2
Delegate IP     : 192.168.1.10
Uxpools port    : 9992
Watchdog port   : 9002
Node priority   : 1
Status          : 4
Status Name     : MASTER
```

```
Node Name       : Linux_host3.localdomain_9993
Host Name       : host3
```

```
Delegate IP   : 192.168.1.10
Uxpool port   : 9993
Watchdog port : 9003
Node priority : 1
Status        : 7
Status Name   : STANDBY
```

8.2.5. pcp_proc_count

`pcp_proc_count` -- 显示uxpool子进程ID的列表，

8.2.5.1. 大纲

`pcp_proc_count` [*option...*]

8.2.5.2. 描述

`pcp_proc_count`显示uxpool子进程ID的列表。如果有多个进程，ID将用空格分隔。

8.2.5.3. 选项

请参见[pcp_common_options](#)。

8.2.6. pcp_proc_info

`pcp_proc_info` -- 显示给定uxpool子进程ID的信息。

8.2.6.1. 大纲

`pcp_proc_info` [*option...*] [*uxpool_child_processid*]

8.2.6.2. 描述

`pcp_proc_info`显示有关给定uxpool子进程ID的信息。

8.2.6.3. 选项

`-a`
`--all`

显示所有子进程及其可用的连接槽。

`-P PID`
`--process-id=PID`

uxpool子进程的PID。

Other options

请参见[pcp_common_options](#)。

如果不指定[-a](#)或[-P](#)，则将打印所有连接的uxpool子进程的进程信息。在这种情况下，如果没有连接的uxpool子进程，则只会打印“无可进程信息”消息。

8.2.6.4. 示例

如下示例所示。

```
$ pcp_proc_info -p 11001 1406
test t-ishii 2018-07-09 16:43:53 2018-07-09 16:44:08 3 0 1 1435 1 1406 0
test t-ishii 2018-07-09 16:43:53 2018-07-09 16:44:08 3 0 1 1436 1 1406 1
```

结果按照以下顺序排列。

1. connected database name
2. connected user name
3. process start-up timestamp
4. connection created timestamp
5. protocol major version
6. protocol minor version
7. connection-reuse counter
8. UXDB backend process id
9. 1 if frontend connected 0 if not
10. uxpools child process id
11. UXDB backend id

如果未指定-a或--all选项并且没有与后端的连接，则不会显示任何内容。如果有多个连接，一个连接的信息会在每一行显示多次。时间戳以EPOCH格式显示。

该--verbose选项可以帮助理解输出，如下所示。

```
$ pcp_proc_info -p 11001 --verbose 1406
Database   : test
Username   : t-ishii
Start time  : 2018-07-09 16:43:53
Creation time: 2018-07-09 16:44:08
Major      : 3
Minor      : 0
Counter    : 1
Backend PID : 1435
Connected   : 1
PID        : 1406
Backend ID  : 0
Database   : test
Username   : t-ishii
Start time  : 2018-07-09 16:43:53
Creation time: 2018-07-09 16:44:08
Major      : 3
Minor      : 0
Counter    : 1
Backend PID : 1436
Connected   : 1
PID        : 1406
Backend ID  : 1
```

8.2.7. pcp_pool_status

pcp_pool_status -- 显示uxpool.conf 中定义的参数值。

8.2.7.1. 大纲

`pcp_pool_status [option...]`

8.2.7.2. 描述

`pcp_pool_status`显示uxpool.conf中定义的参数值。

8.2.7.3. 选项

请参见[pcp_common_options](#)。

8.2.7.4. 示例

如下示例所示。

```
$ pcp_pool_status -h localhost -U uxdb
name : listen_addresses
value: localhost
desc : host name(s) or IP address(es) to listen to

name : port
value: 9999
desc : uxpool accepting port number

name : socket_dir
value: /tmp
desc : uxpool socket directory

name : pcp_port
value: 9898
desc : PCP port # to bind
```

8.2.8. pcp_detach_node

`pcp_detach_node --` 从uxpool中分离给定节点。与uxpool的现有连接被强制断开。

8.2.8.1. 大纲

`pcp_detach_node [option...] [node_id] [gracefully]`

8.2.8.2. 描述

`pcp_detach_node`从uxpool中分离给定节点。如果指定了`failover_command`和/或`follow_master_command`，它们也会被执行。与uxpool的现有连接被强制断开。

`pcp_detach_node`只是分离节点，不接触节点后面的运行后端。当管理员需要维护UXDB节点时，此命令很有用。可以根据需要多次关闭或停止后端。

重新附加分离节点的最安全方法是停止后端并应用[pcp_recovery_node](#)。但是，如果确定没有复制延迟（或延迟将在以后恢复）并且节点（主/备用）的角色不会改变，则可以使用[pcp_recovery_node](#)。

8.2.8.3. 选项

```
-n node_id
--node_id=node_id
```

要分离的后端节点的索引。

```
-g
--gracefully
```

等到所有客户端都断开连接（除非`client_idle_limit_in_recovery`为-1或`recovery_timeout`已过期）。

Other options

请参见[pcp_common_options](#)。

8.2.9. pcp_attach_node

`pcp_attach_node` -- 将给定的节点附加到uxpool。

8.2.9.1. 大纲

```
pcp_attach_node [option...] [node_id]
```

8.2.9.2. 描述

`pcp_attach_node`将给定的节点附加到uxpool。

8.2.9.3. 选项

```
-n node_id
--node_id=node_id
```

要附加的后端节点的索引。

Other options

请参见[pcp_common_options](#)。

8.2.10. pcp_promote_node

`pcp_promote_node` -- 将给定节点提升为uxpool的新主节点。

8.2.10.1. 大纲

```
pcp_promote_node [option...] [node_id] [gracefully]
```

8.2.10.2. 描述

`pcp_promote_node` 将给定节点提升为uxpool的新主节点。仅在主/从流复制中。请注意，这个命令实际上并没有提升备用UXDB后端：它只是改变uxpool 的内部状态并触发故障转移，用户必须在uxpool之外提升备用UXDB。

`pcp_promote_node`执行以下操作。请注意，如果设置了`follow_master_command`，该命令将被执行。在执行此命令之前禁用`follow_master_command`是标准建议。

1. 将备用UXDB的状态从备用更改为主。它只是改变了uxpool的内部状态，并没有真正提升UXDB备用服务器。
2. 将此命令的参数未指定的UXDB节点的状态更改为down。它只是改变了uxpool的内部状态，实际上并没有使UXDB备用服务器停机。
3. 如果设置了`follow_master_command`，对UXDB执行`follow_master_command`。

8.2.10.3. 选项

`-n node_id`
`--node_id=node_id`

要提升为新主节点的后端节点的索引。

`-g`
`--gracefully`

等到所有客户端都断开连接（除非`client_idle_limit_in_recovery`为-1或`recovery_timeout`已过期）。

Other options

请参见[pcp_common_options](#)。

8.2.11. pcp_stop_uxpool

`pcp_stop_uxpool --` 终止uxpool进程。

8.2.11.1. 大纲

`pcp_stop_uxpool [option...] [mode]`

8.2.11.2. 描述

`pcp_stop_uxpool`终止uxpool进程。

8.2.11.3. 选项

`-m mode`
`--mode=mode`

用于终止uxpool进程的关闭模式。

可用的模式如下所示。

- s、smart：智能模式
- f、fast：快速模式
- i、立即：立即模式

Other options

请参见[pcp_common_options](#)。

8.2.12. pcp_recovery_node

`pcp_recovery_node` -- 附加具有恢复功能的给定后端节点。

8.2.12.1. 大纲

`pcp_recovery_node` [*option...*] [*node_id*]

8.2.12.2. 描述

`pcp_recovery_node`使用恢复连接给定的后端节点。

8.2.12.3. 选项

`-n node_id`
`--node-id= node_id`

后端节点的索引。

Other options

请参见[pcp_common_options](#)。

8.3. 其他命令

这部分包含各种`uxpool`命令的参考信息。

8.3.1. ux_md5

`ux_md5` -- 在md5中生成加密密码。

8.3.1.1. 大纲

`ux_md5` [*option...*] `-p`
`ux_md5` [*option...*] `password`

8.3.1.2. 描述

`ux_md5`在md5中生成加密密码。

8.3.1.3. 选项

`-p`
`--prompt`

使用标准输入提示密码。

```
-m
--md5auth
```

生成md5认证密码到认证文件pool_passwd。

```
-u your_username
--username=your_username
```

生成md5身份验证密码时，为your_username创建pool_passwd条目。

```
-f config_file
--config-file=config_file
```

指定uxpool.conf配置文件的路径。

8.3.1.4. 示例

以下是将密码加密为pcp.conf的md5哈希格式的示例。

```
$ ux_md5 -p
password: [your password]
```

或者使用如下示例。

```
$ ux_md5 [your password]
acbd18db4cc2f85cedef654fccc4a4d8
```

ux_md5也可用于在pool_passwd认证文件中添加用户名和md5加密密码的条目。

```
$ ux_md5 -m -f /path/to/uxpool.conf -u username -p
password: [your password]
$ cat /path/to/pool_passwd
username:md55a231fdb710d73268c4f44283487ba2
```

要仅显示md5散列字符串，而不是向pool_passwd添加条目，请传递连接密码和用户名的字符串。例如，如果密码是“password”，用户名是“user”，输出如下所示。

```
$ ux_md5 passworduser
4d45974e13472b5a0be3533de4666414
```

注意

要插入到pool_passwd中的实际条目应该在结果字符串的顶部有“md5”。
即：“md54d45974e13472b5a0be3533de4666414”。

8.3.2. ux_enc

ux_enc — AES256密码加密实用程序。

8.3.2.1. 大纲

```
ux_enc [option...] -p
```

```
ux_enc [option... ] password
```

8.3.2.2. 描述

ux_enc AES256密码加密实用程序。

8.3.2.3. 选项

-k KEY_FILE

--key-file=KEY_FILE

设置加密密钥文件的路径。默认是位于用户主目录中的.uxpoolkey文件，它可以被环境变量UXPOOLKEYFILE覆盖。

-K ENCRYPTION_KEY

--enc-key=ENCRYPTION_KEY

用于加密数据库密码的加密密钥。

-f CONFIG_FILE

--config-file=CONFIG_FILE

指定uxpool.conf文件。

-p

--prompt

使用标准输入提示输入数据库密码。

-P

--prompt-for-key

使用标准输入提示输入加密密钥。

-m

--update-pass

在pool_passwd文件中创建加密的密码条目。

-u your_username

--username= your_username

为名为your_username的数据库用户创建pool_passwd条目。

-h

--help

打印ux_enc的帮助。

8.3.2.4. 示例

如下示例所示。

ux_enc -p

db password: [your password]

或者使用如下示例。

```
./ux_enc foo
trying to read key from file /home/uxpool/.uxpoolkey

jglid1QRgiCl/vfhHUDyVA==
pool_passwd string: AESjglid1QRgiCl/vfhHUDyVA==
```

ux_enc可用于pool_passwd密码。

```
ux_enc -m -f /path/to/uxpool.conf -u username -p
db password: [your password]
```

这将使用给定的密码为用户名添加一个条目。

8.3.3. uxproto

uxproto — 测试UXDB或任何其他理解前端/后端协议的服务器。

8.3.3.1. 大纲

```
uxproto [option...]
```

8.3.3.2. 描述

uxproto测试UXDB或任何其他理解前端/后端协议的服务器。

8.3.3.3. 选项

```
-h hostname
--hostname=hostname
```

运行服务器的机器的主机名。如果该值以斜杠开头，则将其用作Unix域套接字（默认值：Unix域套接字）的目录。

```
-p port
--port=port
```

端口号（默认：5432）。

```
-u username
--user=username
```

用户名（默认：操作系统用户名）。

```
-d databasename
--database=databasename
```

数据库名称（默认值：与用户相同）。

```
-f filename
--proto-data-file=filename
```

描述要发送到UXDB的消息数据的文本文件（默认：uxproto.data）。

-r naptime

--read-nap=naptime

以微秒为单位的小睡时间（默认值：0）。大于0将使uxproto在每次从套接字读取数据之间休眠。这对于模拟慢速客户端很有用。

-D

--debug

启用调试消息。

-v

--version

打印命令版本，然后退出。

-?

--help

显示命令行参数的帮助，然后退出。

8.3.3.4. 示例

在下面的示例中，文件中的第一个字符（即“Q”）表示UXDB前端/后端协议中指定的消息类型。

例外是“Y”和“y”。“Y”从后端读取消息，直到收到“准备好查询”为止。’y’从后端读取消息，而消息来自后端，如果消息在 1 秒内不可用则停止。’Y’用于等待’Q’（简单查询）的回复或在扩展查询中的’S’（同步）之后。’y’可用于接收 ’H’（刷新）之后的消息。

如果要在字符串数据类型中包含 “（双引号），例如 “SELECT * FROM ”aaa””，可以使用 \（反斜杠）来限定它，例如 “”SELECT * FROM ”aaa”” A也可以使用 \ 创建分布在多行上的命令。

**'Q' "SELECT * FROM aaa **
WHERE a = 1"

如下是一个示例输入文件。

```
#
# Test data example
#
'Q' "SELECT * FROM aaa"
'Y'
'P' "S1" "BEGIN" 0
'B' "" "S1" 0 0 0
'E' "" 0
'C' 'S' "S1"
'P' "foo" "SELECT 1" 0
'B' "myportal" "foo" 0 0 0
'E' "myportal" 0
'P' "S2" "COMMIT" 0
'B' "" "S2" 0 0 0
'E' "" 0
'C' 'S' "S2"
'S'
'Y'
```

'X'

如下示例输出。

```
$ uxproto -p 11000 -d test -f sample.data
FE=> Query (query="SELECT * FROM aaa")
<= BE RowDescription
<= BE CommandComplete(SELECT 0)
<= BE ReadyForQuery(I)
FE=> Parse (stmt="S1", query="BEGIN")
FE=> Bind(stmt="S1", portal="")
FE=> Execute(portal="")
FE=> Close(stmt="S1")
FE=> Parse(stmt="foo", query="SELECT 1")
FE=> Bind(stmt="foo", portal="myportal")
FE=> Execute(portal="myportal")
FE=> Parse (stmt="S2", query="COMMIT")
FE=> Bind(stmt="S2", portal="")
FE=> 执行(portal="")
FE=> Close(stmt="S2")
FE=> Sync
<= BE ParseComplete
<= BE BindComplete
<= BE CommandComplete(BEGIN)
<= BE CloseComplete
<= BE ParseComplete
<= BE BindComplete
<= BE DataRow
<= BE CommandComplete(SELECT 1)
<= BE ParseComplete
<= BE BindComplete
<= BE CommandComplete(COMMIT)
<= BE CloseComplete
<= BE ReadyForQuery(I)
FE=> 终止
```

其他示例数据文件。

复制

```
#
# Test data example
#

# CopyIn
#
'Q' "COPY t1 FROM STDIN"
# CopyData
'd' "abc"
# CopyDone
'c'
'Y'

# CopyOut
#
```

```
'Q' "COPY t1 TO STDOUT"
'Y'

#
# Copy fail case
#
'Q' "COPY t1 FROM STDIN"
# CopyData
'd' "abc"
# CopyFail
'f' "uxproto copy fail test"
'Y'
'X'
```

函数调用

```
#
# Test data example
#

# Function call (lo_creat)
# from UXDB's src/include/catalog/ux_proc.data
# { oid => '957', descr => 'large object create',
#   pronomes => 'lo_creat', provolatile => 'v', proparallel => 'u',
#   prorotype => 'oid', proargtypes => 'int4', prosrc => 'be_lo_creat' },

'F' 957 1 0 1 1 "0" 0
'Y'
'X'
```

8.3.4. uxpools_setup

uxpools_setup — 创建一个临时安装的uxpools集群。

8.3.4.1. 大纲

```
uxpools_setup [option...]
```

8.3.4.2. 描述

uxpools_setup创建一个uxpools集群的临时安装，其中包括一个uxpools安装和当前目录下指定数量的UXDB安装。在运行uxpools_setup之前，当前目录必须为空。

uxpools_setup仅用于测试目的，不应用于创建生产安装。

uxpools_setup对本地主机执行ssh。需要配置ssh以便它可以在没有密码的情况下登录到localhost。

目前uxpools_setup支持流式复制模式、本机复制模式、原始模式和逻辑复制模式。要支持watchdog，请参见[watchdog_setup](#)了解详细信息。

8.3.4.3. 选项

uxpools_setup接受以下命令行参数。

-m mode

指定运行模式。mode可以是r（本机复制模式）、s（流复制模式）、n（原始模式）、l（逻辑复制模式）或y（slony 模式）。如果省略，则假定为s。

-n num_clusters

指定UXDB安装的数量。如果省略，则使用2。

-p base_port

指定uxpool和UXDB使用的基本端口号。uxpool端口是base_port。pcp端口是base_port+1。第一个UXDB节点的端口是base_port+2，第二个UXDB节点的端口是base_port+3，依此类推。

如果指定了-ux选项，则第一个UXDB节点的端口被分配给ux_base_port，第二个UXDB节点的端口是 ux_base_port+1，依此类推。

如果省略，则使用11000。

-ux ux_base_port

指定UXDB使用的基本端口号。第一个UXDB节点的端口是base_port+2，第二个UXDB节点的端口是base_port+3等等。

如果省略，则使用base_port+2。

--no-stop

工作后不要停止uxpool和UXDB。

-d

以调试模式启动uxpool。

-s

在流式复制模式下，使用复制槽代替存档。由于归档目录是所有UXDB集群共享的，如果提升了一个备用服务器，归档目录中的时间线将发生变化，其他备用服务器将被停止。uxpool_setup使用的复制槽名称是uxpool_setup_slot。

-r

在恢复脚本（basebackup.sh）中使用ux_rewind命令。如果命令失败，请切换到使用顺序rsync命令。在某些情况下，使用ux_rewind进行恢复比rsync快得多，因为它不会复制整个数据库集群。

8.3.4.4. 环境

uxpool_setup识别以下环境变量。

UXPOOL_INSTALL_DIR

指定uxpool安装目录。uxpool二进制文件应该放在UXPOOL_INSTALL_DIR/bin下，uxpool.conf和pool_hba.conf等应该放在UXPOOL_INSTALL_DIR/etc下。默认为 /usr/local。

UXPOOLDIR

指定uxpool配置文件的路径。默认为UXPOOL_INSTALL_DIR/etc。

UXBIN

指定UXDB命令的路径，例如 `initdb`、`ux_ctl`和`uxsql`。默认为`/usr/local/uxsql/bin`。

UXLIB

指定UXDB共享库的路径。默认是`/usr/local/uxsql/lib`。

UXSOCKET_DIR

指定Unix套接字目录的路径。默认为`/tmp`。

INITDBARG

指定`initdb`命令的参数。默认值为“`--no-locale -E UTF_8`”。

USE_REPLICATION_SLOT

如果为“`true`”，在流式复制模式下，使用复制槽而不是存档。这带来与指定“`-s`”选项相同的效果。

USE_UX_REWIND

如果为“`true`”，则在流式复制模式下，在`basebackup.sh`脚本中使用`ux_rewind`。这带来与指定“`-r`”选项相同的效果。

8.3.4.5. 示例**\$ uxpool_setup**

UXDB major version: 124

Starting set up in streaming replication mode

creating startall and shutdownall

creating failover script

creating database cluster /tmp/test/data0...done.

update uxsinodb.conf

creating uxpool_remote_start

creating basebackup.sh

creating recovery.conf

creating database cluster /tmp/test/data1...done.

update uxsinodb.conf

creating uxpool_remote_start

creating basebackup.sh

creating recovery.conf

temporarily start data0 cluster to create extensions

temporarily start uxpool to create standby nodes

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay	replication_state	replication_sync_state	last_status_change
0	/tmp	11002	up	0.500000	primary	0	true	0			2020-08-18 13:50:19
1	/tmp	11003	down	0.500000	standby	0	false	0			2020-08-18 13:50:18

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

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

0	/tmp	11002	up	0.500000	primary	0	true	0			2020-08-18 13:50:19
---	------	-------	----	----------	---------	---	------	---	--	--	---------------------

1	/tmp	11003	down	0.500000	standby	0	false	0			2020-08-18 13:50:18
---	------	-------	------	----------	---------	---	-------	---	--	--	---------------------

(2 rows)

recovery node 1...pcp_recovery_node -- Command Successful

done.

creating follow master script

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay	replication_state	replication_sync_state	last_status_change
---------	----------	------	--------	-----------	------	------------	-------------------	-------------------	-------------------	------------------------	--------------------

0	/tmp	11002	up	0.500000	primary	0	true	0			
											2020-08-18 13:50:19
1	/tmp	11003	up	0.500000	standby	0	false	0			
											2020-08-18 13:50:23

(2 rows)

shutdown all

uxpool setting for streaming replication mode is done.

To start the whole system, use /tmp/test/startall.

To shutdown the whole system, use /tmp/test/shutdownall.

pcp command user name is "t-ishii", password is "t-ishii".

Each UXDB, uxpools and pcp port is as follows:

#1 port is 11002

#2 port is 11003

uxpool port is 11000

pcp port is 11001

The info above is in README.port.

t-ishii\$./startall

waiting for server to start....5744 2020-08-18 13:50:27 JST LOG: starting UXDB 12.4 on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0, 64-bit

5744 2020-08-18 13:50:27 JST LOG: listening on IPv4 address "0.0.0.0", port 11002

5744 2020-08-18 13:50:27 JST LOG: listening on IPv6 address ":::", port 11002

5744 2020-08-18 13:50:27 JST LOG: listening on Unix socket "/tmp/.s.UXSQL.11002"

5744 2020-08-18 13:50:27 JST LOG: redirecting log output to logging collector process

5744 2020-08-18 13:50:27 JST HINT: Future log output will appear in directory "log".

done

server started

waiting for server to start....5757 2020-08-18 13:50:27 JST LOG: starting UXDB 12.4 on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0, 64-bit

5757 2020-08-18 13:50:27 JST LOG: listening on IPv4 address "0.0.0.0", port 11003

5757 2020-08-18 13:50:27 JST LOG: listening on IPv6 address ":::", port 11003

5757 2020-08-18 13:50:27 JST LOG: listening on Unix socket "/tmp/.s.UXSQL.11003"

5757 2020-08-18 13:50:27 JST LOG: redirecting log output to logging collector process

5757 2020-08-18 13:50:27 JST HINT: Future log output will appear in directory "log".

done

server started

t-ishii\$ uxsql -p 11000 test

uxsql (12.4)

Type "help" for help.

test=# show pool_nodes;

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay	replication_state	replication_sync_state	last_status_change
---------	----------	------	--------	-----------	------	------------	-------------------	-------------------	-------------------	------------------------	--------------------

0	/tmp	11002	up	0.500000	primary	0	true	0			
											2020-08-18 13:50:32

```

1 | /tmp | 11003 | up | 0.500000 | standby | 0 | false | 0 | streaming |
async | 2020-08-18 13:50:32
(2 rows)

```

8.3.5. watchdog_setup

`watchdog_setup` — 使用`watchdog`创建一个临时安装的`uxpool`集群。

8.3.5.1. 大纲

`watchdog_setup` [*option...*]

8.3.5.2. 描述

`watchdog_setup`创建一个`uxpool`集群的临时安装并启用了`watchdog`，其中包括一个`uxpool`安装和当前目录下指定数量的UXDB安装。在运行`watchdog_setup`之前，当前目录必须为空。

`watchdog_setup`仅用于测试目的，不应用于创建生产安装。另请注意，未使用心跳。

`watchdog_setup` 使用`uxpool_setup`作为主力。

目前`watchdog_setup`支持流式复制模式、原生复制模式、逻辑复制模式和原始模式。

8.3.5.3. 选项

`watchdog_setup`接受如下命令行参数。

-wn num_uxpool

指定`uxpool`安装的数量。如果省略，则使用3。

-wp watchdog_base_port

指定`uxpool`和UXDB 使用的起始基本端口号。对于第一个`uxpool`，`uxpool`端口是 `watchdog_base_port`。`pcp`端口是`watchdog_base_port` + 1，`watchdog`端口是 `watchdog_base_port`+2。`wd_heartbeat_port`是`watchdog_base_port`+3（虽然没有使用 `heartbeat`）。第一个UXDB节点的端口是 `watchdog_base_port` + 4，第二个UXDB节点的端口是`watchdog_base_port`+5，依此类推。

如果省略，则使用50000。

-m mode

指定运行模式。`mode` 可以是`r`（本机复制模式）、`s`（流式复制模式）或 `n`（原始模式）。如果省略，则使用`s`。

-n num_clusters

指定UXDB安装的数量。如果省略，则使用2。

--no-stop

工作后不要停止`uxpool`和UXDB。

-d

以调试模式启动`uxpool`。

8.3.5.4. 环境

watchdog_setup识别以下环境变量。

UXPOOL_SETUP

指定uxpool_setup命令的路径。默认为“uxpool_setup”，因此假设uxpool_setup在命令搜索路径中。

UXPOOL_INSTALL_DIR

指定uxpool安装目录。uxpool二进制文件应该放在UXPOOL_INSTALL_DIR/bin下，uxpool.conf和pool_hba.conf等应该放在UXPOOL_INSTALL_DIR/etc下。默认为/usr/local。

UXPOOLDIR

指定uxpool配置文件的路径。默认为UXPOOL_INSTALL_DIR/etc。

UXBIN

指定UXDB命令的路径，例如initdb、ux_ctl和uxsql。默认为/usr/local/uxsql/bin。

UXLIB

指定UXDB共享库的路径。默认是/usr/local/uxsql/lib。

UXSOCKET_DIR

指定Unix套接字目录的路径。默认为/tmp。

INITDBARG

指定initdb命令的参数。默认值为“--no-locale -E UTF_8”。

8.3.5.5. 示例

```
$ watchdog_setup
Starting set up
===== setting up uxpoo 0 =====
Starting set up in streaming replication mode
creating startall and shutdownall
creating failover script
creating database cluster /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool0/data0...done.
update uxsinodb.conf
creating uxpoo_remote_start
creating basebackup.sh
creating recovery.conf
creating database cluster /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool0/data1...done.
update uxsinodb.conf
creating uxpoo_remote_start
creating basebackup.sh
creating recovery.conf
temporarily start data0 cluster to create extensions
temporarily start uxpoo to create standby nodes
```

INFO: unrecognized configuration parameter "debug_level"

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay
0	/tmp	51000	up	0.500000	primary	0	true	0
1	/tmp	51001	down	0.500000	standby	0	false	0

(2 rows)

recovery node 1...pcp_recovery_node -- Command Successful
done.

creating follow master script

Pager usage is off.

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay
0	/tmp	51000	up	0.500000	primary	0	false	0
1	/tmp	51001	up	0.500000	standby	0	true	0

(2 rows)

shutdown all

uxpool setting for streaming replication mode is done.

To start the whole system, use /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool0/startall.

To shutdown the whole system, use /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool0/shutdownall.

pcp command user name is "t-ishii", password is "t-ishii".

Each UXDB, uxpools and pcp ports are as follows:

#1 port is 51000

#2 port is 51001

uxpool port is 50000

pcp port is 50001

The info above is in README.port.

===== setting up uxpools 1 =====

Starting set up in streaming replication mode

creating startall and shutdownall

creating failover script

creating database cluster /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool1/data0...done.

update uxsinodb.conf

creating uxpools_remote_start

creating basebackup.sh

creating recovery.conf

creating database cluster /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool1/data1...done.

update uxsinodb.conf

creating uxpools_remote_start

creating basebackup.sh

creating recovery.conf

temporarily start data0 cluster to create extensions

temporarily start uxpools to create standby nodes

INFO: unrecognized configuration parameter "debug_level"

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay
0	/tmp	51000	up	0.500000	primary	0	true	0
1	/tmp	51001	down	0.500000	standby	0	false	0

(2 rows)

recovery node 1...pcp_recovery_node -- Command Successful

done.

creating follow master script

Pager usage is off.

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay
0	/tmp	51000	up	0.500000	primary	0	true	0
1	/tmp	51001	up	0.500000	standby	0	false	0

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

0	/tmp	51000	up	0.500000	primary	0	true	0
---	------	-------	----	----------	---------	---	------	---

1	/tmp	51001	up	0.500000	standby	0	false	0
---	------	-------	----	----------	---------	---	-------	---

(2 rows)

shutdown all

uxpool setting for streaming replication mode is done.

To start the whole system, use /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool1/startall.

To shutdown the whole system, use /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool1/shutdownall.

pcp command user name is "t-ishii", password is "t-ishii".

Each UXDB, uxpools and pcp port is as follows:

#1 port is 51000

#2 port is 51001

uxpool port is 50004

pcp port is 50005

The info above is in README.port.

===== setting up uxpools 2 =====

Starting set up in streaming replication mode

creating startall and shutdownall

creating failover script

creating database cluster /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool2/data0...done.

update uxsinodb.conf

creating uxpools_remote_start

creating basebackup.sh

creating recovery.conf

creating database cluster /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool2/data1...done.

update uxsinodb.conf

creating uxpools_remote_start

creating basebackup.sh

creating recovery.conf

temporarily start data0 cluster to create extensions

temporarily start uxpools to create standby nodes

INFO: unrecognized configuration parameter "debug_level"

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay
0	/tmp	51000	up	0.500000	primary	0	true	0
1	/tmp	51001	down	0.500000	standby	0	false	0

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

0	/tmp	51000	up	0.500000	primary	0	true	0
---	------	-------	----	----------	---------	---	------	---

1	/tmp	51001	down	0.500000	standby	0	false	0
---	------	-------	------	----------	---------	---	-------	---

(2 rows)

recovery node 1...pcp_recovery_node -- Command Successful

done.

creating follow master script

Pager usage is off.

node_id	hostname	port	status	lb_weight	role	select_cnt	load_balance_node	replication_delay
0	/tmp	51000	up	0.500000	primary	0	true	0
1	/tmp	51001	up	0.500000	standby	0	false	0

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

0	/tmp	51000	up	0.500000	primary	0	true	0
---	------	-------	----	----------	---------	---	------	---

1	/tmp	51001	up	0.500000	standby	0	false	0
---	------	-------	----	----------	---------	---	-------	---

(2 rows)

shutdown all

uxpool setting for streaming replication mode is done.

To start the whole system, use /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool2/startall.

To shutdown the whole system, use /home/t-ishii/work/uxpool/current/uxpool2/src/test/a/uxpool2/shutdownall.

pcp command user name is "t-ishii", password is "t-ishii".

Each UXDB, uxpools and pcp ports are as follows:

#1 port is 51000

#2 port is 51001

uxpool port is 50008

pcp port is 50009

The info above is in README.port.

\$ ls

uxpool0 uxpools shutdownall startall

\$ sh startall

waiting for server to start....16123 2016-08-18 16:26:53 JST LOG: redirecting log output to logging collector process

16123 2016-08-18 16:26:53 JST HINT: Future log output will appear in directory "ux_log".

done

server started

waiting for server to start....16136 2016-08-18 16:26:54 JST LOG: redirecting log output to logging collector process

16136 2016-08-18 16:26:54 JST HINT: Future log output will appear in directory "ux_log".

done

server started

t-ishii@localhost: uxsql -p 50000 test

Pager usage is off.

uxsql (9.5.4)

Type "help" for help.

test=# \q

\$ pcp_watchdog_info -p 50001 -v

Password:

Watchdog Cluster Information

Total Nodes : 3

Remote Nodes : 2

Quorum state : QUORUM EXIST

Alive Remote Nodes : 2

VIP up on local node : NO

Master Node Name : Linux_tishii-CF-SX3HE4BP_50004

Master Host Name : localhost

Watchdog Node Information

Node Name : Linux_tishii-CF-SX3HE4BP_50000

Host Name : localhost

Delegate IP : Not_Set

Uxpools port : 50000

```

Watchdog port : 50002
Node priority : 1
Status       : 7
Status Name  : STANDBY

Node Name    : Linux_tishii-CF-SX3HE4BP_50004
Host Name    : localhost
Delegate IP  : Not_Set
Uxpool port  : 50004
Watchdog port : 50006
Node priority : 1
Status       : 4
Status Name  : MASTER

Node Name    : Linux_tishii-CF-SX3HE4BP_50008
Host Name    : localhost
Delegate IP  : Not_Set
Uxpool port  : 50008
Watchdog port : 50010
Node priority : 1
Status       : 7
Status Name  : STANDBY

```

8. 4. SQL类型命令

这部分包含各种SQL类型uxpool命令的参考信息。可以使用标准UXDB客户端（如uxsql）在SQL会话中发出这些命令。它们不会转发到后端数据库：而是由uxpool服务器处理。请注意，SQL 类型的命令不能用于扩展查询模式。将收到来自UXDB的解析错误。

8. 4. 1. UXPOOL SHOW

UXPOOL SHOW — 显示配置参数的值。

8. 4. 1. 1. 大纲

```

UXPOOL SHOW configuration_parameter

UXPOOL SHOW configuration_parameter_group

UXPOOL SHOW ALL

```

8. 4. 1. 2. 描述

UXPOOL SHOW将显示uxpool配置参数的当前值。该命令类似于UXDB中的SHOW命令，但增加了UXPOOL关键字以区别于UXDB的SHOW命令。

8. 4. 1. 3. 参数

表 8.1. UXPOOL SHOW 参数说明

参数	描述
<i>configuration_parameter</i>	uxpool配置参数的名称。

参数	描述
	注意：如果参数包含大写字母（例如 delegate_IP），则参数名称必须用双引号括起来。
configuration_parameter_group	<p>所述名称uxpool配置参数组。目前有三个参数组。</p> <ul style="list-style-type: none"> • backend：所有后端配置参数的配置组。 • other_uxpool：所有watchdog节点配置参数的配置组。 • heartbeat：所有watchdog心跳节点配置参数的配置组。 • health_check：所有健康检查参数的配置组。
ALL	显示所有配置参数的值，并带有说明。

8.4.1.4. 示例

显示参数端口的当前设置，如下所示。

UXPOOL SHOW port;

```
port
-----
9999
(1 row)
```

显示参数black_function_list的当前设置，如下所示。

UXPOOL SHOW black_function_list;

```
black_function_list
-----
nextval,setval
(1 row)
```

显示属于后端组的所有配置参数的当前设置，如下所示。

UXPOOL SHOW backend;

```
item          | value          | description
-----+-----+-----
backend_hostname0 | 127.0.0.1      | hostname or IP address of UXDB backend.
backend_port0    | 5434           | port number of UXDB backend.
backend_weight0  | 0              | load balance weight of backend.
backend_data_directory0 | /var/lib/uxsql/data | data directory of the backend.
backend_flag0    | ALLOW_TO_FAILOVER | Controls various backend behavior.
backend_hostname1 | 127.0.0.1      | hostname or IP address of UXDB backend.
backend_port1    | 5432           | port number of UXDB backend.
backend_weight1  | 1              | load balance weight of backend.
backend_data_directory1 | /home/work/installed/ux | data directory of the backend.
backend_flag1    | ALLOW_TO_FAILOVER | Controls various backend behavior.
(10 rows)
```

显示所有设置，如下所示。

UXPOOL SHOW ALL;

item	value	description
backend_hostname0	127.0.0.1	hostname or IP address of UXDB backend.
backend_port0	5434	port number of UXDB backend.
backend_weight0	0	load balance weight of backend.
backend_data_directory0	/var/lib/uxsql/data	data directory of the backend.
backend_flag0	ALLOW_TO_FAILOVER	Controls various backend behavior.
backend_hostname1	127.0.0.1	hostname or IP address of UXDB backend.
backend_port1	5432	port number of UXDB backend.
backend_weight1	1	load balance weight of backend.
backend_data_directory1	/home/work/installed/ux	data directory of the backend.
backend_flag1	ALLOW_TO_FAILOVER	Controls various backend behavior.
other_uxpool_hostname0	localhost	Hostname of other uxpools node for watchdog connection.
.		
.		
.		
ssl	off	Enables SSL support for frontend and backend connections

(138 rows)

8.4.2. UXPOOL SET

UXPOOL SET — 改变一个配置参数。

8.4.2.1. 大纲

UXPOOL SET *configuration_parameter* { *TO* | = } { *value* | 'value' | DEFAULT }

8.4.2.2. 描述

UXPOOL SET命令更改当前会话的uxpool配置参数的值。该命令类似于UXDB中的SET命令，但增加了UXPOOL关键字以区别于UXDB SET命令。

8.4.2.3. 示例

更改client_idle_limit参数的值，如下所示。

```
UXPOOL SET client_idle_limit = 350;
```

将client_idle_limit参数的值重置为默认值，如下所示。

```
UXPOOL SET client_idle_limit TO DEFAULT;
```

更改log_min_messages参数的值，如下所示。

```
UXPOOL SET log_min_messages TO INFO;
```

8.4.3. UXPOOL RESET

UXPOOL RESET — 将配置参数的值恢复为默认值。

8.4.3.1. 大纲

UXPOOL RESET *configuration_parameter*

UXPOOL RESET ALL

8.4.3.2. 描述

UXPOOL RESET命令将uxpool配置参数的值恢复为默认值。如果在当前会话中没有为它发出UXPOOL SET，则默认值定义为该参数将具有的值。该命令类似于UXDB中的 RESET命令，但增加了UXPOOL关键字以区别于UXDB RESET命令。

8.4.3.3. 参数

表 8.2. UXPOOL RESET 参数说明

参数	描述
<code>configuration_parameter</code>	可设置的uxpool配置参数的名称。
ALL	将所有可设置的uxpool配置参数重置为默认值。

8.4.3.4. 示例

重置`client_idle_limit`参数的值，如下所示。

UXPOOL RESET `client_idle_limit`;

将所有参数的值重置为默认值，如下所示。

UXPOOL RESET ALL;

8.4.4. SHOW POOL_STATUS

SHOW POOL_STATUS — 发送回配置参数列表及其名称、值和描述。

8.4.4.1. 大纲

SHOW POOL_STATUS

8.4.4.2. 描述

SHOW POOL_STATUS显示uxpool配置参数的当前值。

此命令类似于UXPOOL SHOW命令。

8.4.5. SHOW POOL_NODES

SHOW POOL_NODES — 返回所有配置节点的列表。

8.4.5.1. 大纲

SHOW POOL_NODES

8.4.5.2. 描述

SHOW POOL_NODES显示节点id、主机名、端口、状态、权重（只有在使用负载均衡模式时才有意义）、角色、向每个后端发出的SELECT查询计数、每个节点是否是负载均衡节点与否，复制延迟（仅在流复制模式下）和上次状态更改时间。除了此复制状态和同步状态之外，uxpool 或之后的备用节点也会显示。在pcp_node_info参考中解释了状态列中的可能值。如果主机名类似于“/tmp”，则表示uxpool正在使用UNIX域套接字连接到后端。SELECT 计数不包括uxpool使用的内部查询。在uxpool启动时，计数器也被重置为零。上次状态更改时间最初设置为uxpool启动的时间。之后，每当“状态”或“角色”改变时，它都会更新。

如下所示，是一个示例会话。

```
test=# show pool_nodes;
node_id | hostname | port | status | lb_weight | role | select_cnt | load_balance_node | replication_delay
| replication_state | replication_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----
+-----+-----+-----+-----+-----+-----+-----+-----+-----
0 | /tmp | 11002 | up | 0.500000 | primary | 0 | false | 0 | |
| 2019-04-22 16:13:46
1 | /tmp | 11003 | up | 0.500000 | standby | 0 | true | 0 | streaming |
async | 2019-04-22 16:13:46
(2 rows)
```

8.4.6. SHOW POOL_PROCESSES

SHOW POOL_PROCESSES — 返回所有等待连接和处理连接的uxpool进程的列表。

8.4.6.1. 大纲

SHOW POOL_PROCESSES

8.4.6.2. 描述

SHOW POOL_PROCESSES返回所有等待连接和处理连接的uxpool进程的列表。

总共包含6列，如下所示。

- pool_pid是显示的uxpool进程的PID。
- start_time是此进程启动时的时间戳。
- database是此进程当前活动后端的数据库名称。
- username是用于此进程的当前活动后端连接的用户名。
- create_time是连接的创建时间和日期。
- pool_counter计算客户端使用此连接池（进程）的次数。

如下所示，是一个示例会话。

```
test=# show pool_processes;
pool_pid | start_time | database | username | create_time | pool_counter
-----+-----+-----+-----+-----+-----
19696 | 2016-10-17 13:24:17 | uxdb | t-ishii | 2016-10-17 13:35:12 | 1
19697 | 2016-10-17 13:24:17 | | | |
```

19698	2016-10-17 13:24:17			
19699	2016-10-17 13:24:17			
19700	2016-10-17 13:24:17			
19701	2016-10-17 13:24:17			
19702	2016-10-17 13:24:17			
19703	2016-10-17 13:24:17			
19704	2016-10-17 13:24:17			
19705	2016-10-17 13:24:17			
19706	2016-10-17 13:24:17			
19707	2016-10-17 13:24:17			
19708	2016-10-17 13:24:17			
19709	2016-10-17 13:24:17			
19710	2016-10-17 13:24:17			
19711	2016-10-17 13:24:17			
19712	2016-10-17 13:24:17			
19713	2016-10-17 13:24:17			
19714	2016-10-17 13:24:17			
19715	2016-10-17 13:24:17			
19716	2016-10-17 13:24:17			
19717	2016-10-17 13:24:17			
19718	2016-10-17 13:24:17			
19719	2016-10-17 13:24:17			
19720	2016-10-17 13:24:17			
20024	2016-10-17 13:33:46			
19722	2016-10-17 13:24:17 test	t-ishii	2016-10-17 13:34:42	1
19723	2016-10-17 13:24:17			
19724	2016-10-17 13:24:17			
19725	2016-10-17 13:24:17			
19726	2016-10-17 13:24:17			
19727	2016-10-17 13:24:17			

(32 rows)

8.4.7. SHOW POOL_POOLS

SHOW POOL_POOLS — 返回由uxpool处理的池列表。

8.4.7.1. 大纲

SHOW POOL_POOLS

8.4.7.2. 描述

SHOW POOL_POOLS发送回uxpool处理的池列表。

总共包含12列，如下所示。

- pool_pid是显示的uxpool进程的PID。
- start_time是此进程启动时的时间戳。
- pool_id是池标识符（应该在0和max_pool-1之间）。
- backend_id是后端标识符（应该在0和配置的后端数减一之间）。
- database是此进程的池ID连接的数据库名称。

- username是此进程的池ID连接的用户名。
- create_time是连接的创建时间和日期。
- majorversion和minorversion是在此连接中使用的协议版本号。
- pool_counter计算客户端使用此连接池（进程）的次数。
- pool_backendpid是UXDB进程的PID。
- 如果前端当前正在使用此后端，则pool_connected为真（1）。

它总是会返回`num_init_children * max_pool * number_of_backends` 行。如下所示，是一个示例会话。

```
test=# show pool_pools;
pool_pid | start_time | pool_id | backend_id | database | username | create_time | majorversion |
| minorversion | pool_counter | pool_backendpid | pool_connected
-----+-----+-----+-----+-----+-----+-----+-----+-----
+-----+-----+-----+-----+-----+-----+-----+-----+
19696 | 2016-10-17 13:24:17 | 0 | 0 | uxdb | t-ishii | 2016-10-17 13:35:12 | 3 | 0
| 1 | 20079 | 1
19696 | 2016-10-17 13:24:17 | 0 | 1 | uxdb | t-ishii | 2016-10-17 13:35:12 | 3 | 0
| 1 | 20080 | 1
19696 | 2016-10-17 13:24:17 | 1 | 0 | | | | 0 | 0 | 0 |
0 | 0
19696 | 2016-10-17 13:24:17 | 1 | 1 | | | | 0 | 0 | 0 |
0 | 0
19696 | 2016-10-17 13:24:17 | 2 | 0 | | | | 0 | 0 | 0 |
0 | 0
19696 | 2016-10-17 13:24:17 | 2 | 1 | | | | 0 | 0 | 0 |
0 | 0
19696 | 2016-10-17 13:24:17 | 3 | 0 | | | | 0 | 0 | 0 |
0 | 0
19696 | 2016-10-17 13:24:17 | 3 | 1 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 0 | 0 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 0 | 1 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 1 | 0 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 1 | 1 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 2 | 0 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 2 | 1 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 3 | 0 | | | | 0 | 0 | 0 |
0 | 0
19697 | 2016-10-17 13:24:17 | 3 | 1 | | | | 0 | 0 | 0 |
0 | 0
19698 | 2016-10-17 13:24:17 | 0 | 0 | | | | 0 | 0 | 0 |
0 | 0
19698 | 2016-10-17 13:24:17 | 0 | 1 | | | | 0 | 0 | 0 |
0 | 0
```

19698	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19698	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19698	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19698	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19698	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19698	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19699	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19699	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19699	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19699	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19699	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19699	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19699	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19699	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19700	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19700	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19700	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19700	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19700	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19700	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19700	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19700	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19701	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19701	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19701	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19701	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19701	2016-10-17 13:24:17 2	0				0	0	0	
0	0								

19701	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19701	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19701	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19702	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19702	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19702	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19702	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19702	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19702	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19702	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19702	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19703	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19703	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19703	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19703	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19703	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19703	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19703	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19703	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19704	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19704	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19704	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19704	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19704	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19704	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19704	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19704	2016-10-17 13:24:17 3	1				0	0	0	
0	0								

19705 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19705 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19705 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19705 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19705 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19705 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19705 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19705 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19706 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19706 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19706 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19706 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19706 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19706 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19706 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19706 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19707 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19707 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19707 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19707 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19707 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19707 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19707 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19707 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19708 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19708 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19708 0	2016-10-17 13:24:17 1 0	0				0	0	0	

19708	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19708	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19708	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19708	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19708	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19709	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19709	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19709	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19709	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19709	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19709	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19709	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19709	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19710	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19710	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19710	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19710	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19710	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19710	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19710	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19710	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19711	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19711	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19711	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19711	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19711	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19711	2016-10-17 13:24:17 2	1				0	0	0	
0	0								

19711	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19711	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19712	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19712	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19712	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19712	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19712	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19712	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19712	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19712	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19713	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19713	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19713	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19713	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19713	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19713	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19713	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19713	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19714	2016-10-17 13:24:17 0	0				0	0	0	
0	0								
19714	2016-10-17 13:24:17 0	1				0	0	0	
0	0								
19714	2016-10-17 13:24:17 1	0				0	0	0	
0	0								
19714	2016-10-17 13:24:17 1	1				0	0	0	
0	0								
19714	2016-10-17 13:24:17 2	0				0	0	0	
0	0								
19714	2016-10-17 13:24:17 2	1				0	0	0	
0	0								
19714	2016-10-17 13:24:17 3	0				0	0	0	
0	0								
19714	2016-10-17 13:24:17 3	1				0	0	0	
0	0								
19715	2016-10-17 13:24:17 0	0				0	0	0	
0	0								

19715 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19715 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19715 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19715 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19715 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19715 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19715 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19716 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19716 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19716 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19716 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19716 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19716 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19716 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19716 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19717 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19717 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19717 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19717 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19717 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19717 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19717 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19717 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19718 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19718 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19718 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19718 0	2016-10-17 13:24:17 1 0	1				0	0	0	

19718 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19718 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19718 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19718 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19719 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19719 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19719 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19719 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19719 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19719 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19719 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19719 0	2016-10-17 13:24:17 3 0	1				0	0	0	
19720 0	2016-10-17 13:24:17 0 0	0				0	0	0	
19720 0	2016-10-17 13:24:17 0 0	1				0	0	0	
19720 0	2016-10-17 13:24:17 1 0	0				0	0	0	
19720 0	2016-10-17 13:24:17 1 0	1				0	0	0	
19720 0	2016-10-17 13:24:17 2 0	0				0	0	0	
19720 0	2016-10-17 13:24:17 2 0	1				0	0	0	
19720 0	2016-10-17 13:24:17 3 0	0				0	0	0	
19720 0	2016-10-17 13:24:17 3 0	1				0	0	0	
20024 1	2016-10-17 13:33:46 0 22055 1	0	test	t-ishii	2016-10-17 14:30:53 3	0			
20024 1	2016-10-17 13:33:46 0 22056 1	1	test	t-ishii	2016-10-17 14:30:53 3	0			
20024 0	2016-10-17 13:33:46 1 0	0				0	0	0	
20024 0	2016-10-17 13:33:46 1 0	1				0	0	0	
20024 0	2016-10-17 13:33:46 2 0	0				0	0	0	
20024 0	2016-10-17 13:33:46 2 0	1				0	0	0	
20024 0	2016-10-17 13:33:46 3 0	0				0	0	0	

20024	2016-10-17 13:33:46	3	1				0	0	0	
0	0									
20600	2016-10-17 13:46:58	0	0				0	0	0	
0	0									
20600	2016-10-17 13:46:58	0	1				0	0	0	
0	0									
20600	2016-10-17 13:46:58	1	0				0	0	0	
0	0									
20600	2016-10-17 13:46:58	1	1				0	0	0	
0	0									
20600	2016-10-17 13:46:58	2	0				0	0	0	
0	0									
20600	2016-10-17 13:46:58	2	1				0	0	0	
0	0									
20600	2016-10-17 13:46:58	3	0				0	0	0	
0	0									
20600	2016-10-17 13:46:58	3	1				0	0	0	
0	0									
19723	2016-10-17 13:24:17	0	0				0	0	0	
0	0									
19723	2016-10-17 13:24:17	0	1				0	0	0	
0	0									
19723	2016-10-17 13:24:17	1	0				0	0	0	
0	0									
19723	2016-10-17 13:24:17	1	1				0	0	0	
0	0									
19723	2016-10-17 13:24:17	2	0				0	0	0	
0	0									
19723	2016-10-17 13:24:17	2	1				0	0	0	
0	0									
19723	2016-10-17 13:24:17	3	0				0	0	0	
0	0									
19723	2016-10-17 13:24:17	3	1				0	0	0	
0	0									
19724	2016-10-17 13:24:17	0	0				0	0	0	
0	0									
19724	2016-10-17 13:24:17	0	1				0	0	0	
0	0									
19724	2016-10-17 13:24:17	1	0				0	0	0	
0	0									
19724	2016-10-17 13:24:17	1	1				0	0	0	
0	0									
19724	2016-10-17 13:24:17	2	0				0	0	0	
0	0									
19724	2016-10-17 13:24:17	2	1				0	0	0	
0	0									
19724	2016-10-17 13:24:17	3	0				0	0	0	
0	0									
19724	2016-10-17 13:24:17	3	1				0	0	0	
0	0									
19725	2016-10-17 13:24:17	0	0				0	0	0	
0	0									
19725	2016-10-17 13:24:17	0	1				0	0	0	
0	0									

19725	2016-10-17 13:24:17	1	0				0	0	0	
0	0									
19725	2016-10-17 13:24:17	1	1				0	0	0	
0	0									
19725	2016-10-17 13:24:17	2	0				0	0	0	
0	0									
19725	2016-10-17 13:24:17	2	1				0	0	0	
0	0									
19725	2016-10-17 13:24:17	3	0				0	0	0	
0	0									
19725	2016-10-17 13:24:17	3	1				0	0	0	
0	0									
19726	2016-10-17 13:24:17	0	0				0	0	0	
0	0									
19726	2016-10-17 13:24:17	0	1				0	0	0	
0	0									
19726	2016-10-17 13:24:17	1	0				0	0	0	
0	0									
19726	2016-10-17 13:24:17	1	1				0	0	0	
0	0									
19726	2016-10-17 13:24:17	2	0				0	0	0	
0	0									
19726	2016-10-17 13:24:17	2	1				0	0	0	
0	0									
19726	2016-10-17 13:24:17	3	0				0	0	0	
0	0									
19726	2016-10-17 13:24:17	3	1				0	0	0	
0	0									
19727	2016-10-17 13:24:17	0	0				0	0	0	
0	0									
19727	2016-10-17 13:24:17	0	1				0	0	0	
0	0									
19727	2016-10-17 13:24:17	1	0				0	0	0	
0	0									
19727	2016-10-17 13:24:17	1	1				0	0	0	
0	0									
19727	2016-10-17 13:24:17	2	0				0	0	0	
0	0									
19727	2016-10-17 13:24:17	2	1				0	0	0	
0	0									
19727	2016-10-17 13:24:17	3	0				0	0	0	
0	0									
19727	2016-10-17 13:24:17	3	1				0	0	0	
0	0									

(256 rows)

8.4.8. SHOW POOL_VERSION

SHOW POOL_VERSION — 显示一个包含uxpool版本号的字符串。

8.4.8.1. 大纲

SHOW POOL_VERSION

8.4.8.2. 描述

SHOW POOL_VERSION显示一个包含uxpool版本号的字符串。

如下所示，是一个示例会话。

```
test=# show pool_version;
pool_version
-----
3.6.0 (subaruboshi)
(1 row)
```

8.4.9. SHOW POOL_CACHE

SHOW POOL_CACHE — 显示缓存存储统计信息。

8.4.9.1. 大纲

SHOW POOL_CACHE

8.4.9.2. 描述

如果启用内存查询缓存，则SHOW POOL_CACHE显示内存查询缓存统计信息。

如下所示，是一个示例会话。

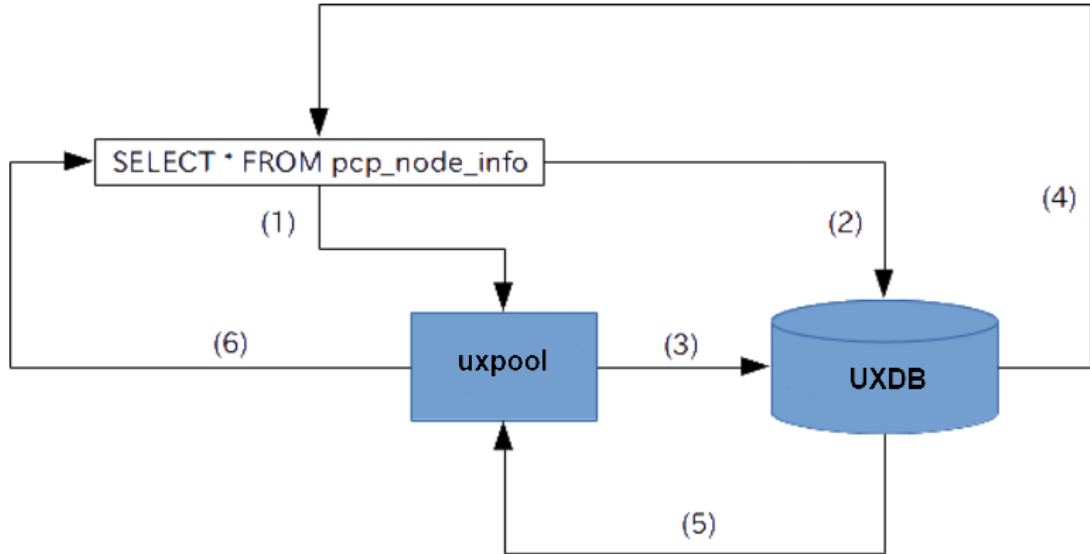
```
test=# \x
\x
Expanded display is on.
test=# show pool_cache;
show pool_cache;
-[ RECORD 1 ]-----+-----
num_cache_hits      | 891703
num_selects         | 99995
cache_hit_ratio      | 0.90
num_hash_entries    | 131072
used_hash_entries    | 99992
num_cache_entries    | 99992
used_cache_entries_size | 12482600
free_cache_entries_size | 54626264
fragment_cache_entries_size | 0
```

注意

如果缓存存储是memcached，则除num_cache_hits、num_selects和cache_hit_ratio之外的所有列的值都显示为0。

8.5. uxpool_adm扩展

uxpool_adm是一组扩展，允许SQL访问[PCP命令](#)（实际上是pcp库）。它使用外部数据包装器，如下图所示。



可以通过uxpool (1) 或UXDB (2) 调用这些函数。在情况 (1) 中，uxpool 接受来自用户 (1) 的查询，然后转发到UXDB (3)。UXDB 连接到uxpool (5)，uxpool 用结果 (3) 回复UXDB。UXDB 将结果返回给uxpool (5)，uxpool 将数据转发给用户 (6)。

在情况 (2) 中，UXDB 接受来自用户 (2) 的查询。UXDB 连接到uxpool (5)，uxpool 用结果 (3) 回复UXDB。UXDB 将数据回复给用户 (4)。

调用 `uxpool_adm` 函数有两种形式：第一种形式接受uxpool主机名（或IP地址）、pcp端口号、pcp用户名、其密码和另一个参数。

在第二种形式中，uxpool服务器名称是必需的。服务器名称必须已经使用UXDB 的“CREATE FOREIGN SERVER”命令定义。pcp端口号被硬编码为9898，假设pcp 用户名与调用者的UXDB用户名相同。密码是从\$HOME/.pcppass中提取的。

安装 `uxpool_adm`

`uxpool_adm`是一个扩展，应该安装在所有UXDB服务器上。

```
$ cd src/sql/uxpool_adm
$ make
$ make install
```

然后为要访问的每个数据库发出以下SQL命令。

```
$ uxsql ...
$ CREATE EXTENSION uxpool_adm
```

8.5.1. uxpool_adm_pcp_node_info

`uxpool_adm_pcp_node_info` — 显示给定节点ID信息的函数。

8.5.1.1. 大纲

`pcp_node_info` returns record(integer node_id, text host, integer port, text username, text password, out status text, out weight float4, out role text, out replication_delay bigint, out replication_state text, out replication_sync_state text, out last_status_change timestamp);

`pcp_node_info` returns record(integer node_id, text pcp_server, out status text, out weight float4, out role text, out replication_delay bigint, out replication_state text, out replication_sync_state text, out last_status_change timestamp);

8.5.1.2. 描述

`pcp_node_info` 显示有关给定节点ID的信息。

8.5.1.3. 参数

表 8.3. `pcp_node_info` 参数说明

参数	描述
<code>node_id</code>	要获取信息的后端节点的索引。
<code>pcp_server</code>	pcp服务器的外部服务器名称。
Other arguments	请参见 pcp_common_options 。

8.5.1.4. 示例

```
test=# SELECT * FROM pcp_node_info(1,"11001",'t-ishii','t-ishii');
 host | port | status | weight | role | replication_delay | replication_state | replication_sync_state |
 last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----
 /tmp | 11003 | Connection in use | 0.5 | Standby | 0 | streaming | async |
 2019-04-23 15:02:46
(1 row)
```

8.5.2. `uxpool_adm_pcp_pool_status`

`uxpool_adm_pcp_pool_status` — 一个在`uxpool.conf`中检索参数的函数。

8.5.2.1. 大纲

`pcp_pool_status` returns record(text host, integer port, text username, text password, out item text, out value text, out description text);

`pcp_pool_status` returns record(text pcp_server, out item text, out value text, out description text);

8.5.2.2. 描述

`pcp_pool_status`检索`uxpool.conf`中的参数。

8.5.2.3. 参数

表 8.4. `pcp_pool_status` 参数说明

参数	描述
<code>pcp_server</code>	pcp服务器的外部服务器名称。
Other arguments	请参见 pcp_common_options 。

8.5.2.4. 示例

```
test=# SELECT * FROM pcp_pool_status('localhost',11001,'t-ishii','t-ishii') WHERE item ~
'backend.*0';
    item    |          value          |          description
-----+-----+-----
backend_hostname0 | /tmp                    | backend #0 hostname
backend_port0    | 11002                   | backend #0 port number
backend_weight0  | 0.500000                | weight of backend #0
backend_data_directory0 | /home/t-ishii/work/uxpool/current/aaa/data0 | data directory for backend #0
backend_status0  | 2                       | status of backend #0
backend_flag0    | ALLOW_TO_FAILOVER       | backend #0 flag
(6 rows)
```

8.5.3. uxpool_adm_pcp_node_count

uxpool_adm_pcp_node_count — 一个检索后端节点数量的函数。

8.5.3.1. 大纲

pcp_node_count returns integer(text host, integer port, text username, text password);

pcp_node_count returns integer(text pcp_server);

8.5.3.2. 描述

pcp_node_count 检索数据库节点的数量。

8.5.3.3. 参数

表 8.5. pcp_node_count 参数说明

参数	描述
pcp_server	pcp服务器的外部服务器名称。
Other arguments	请参见 pcp_common_options 。

8.5.3.4. 示例

```
test=# SELECT * FROM pcp_node_count('localhost',11001,'t-ishii','t-ishii');
 node_count
-----
2
(1 row)
```

8.5.4. uxpool_adm_pcp_attach_node

uxpool_adm_pcp_attach_node — 附加给定节点ID的函数。

8.5.4.1. 大纲

pcp_attach_node returns record(integer node_id, text host, integer port, text username, text password, out node_attached boolean);

`pcp_attach_node` returns record(integer node_id, text pcp_server, out node_attached boolean);

8.5.4.2. 描述

`pcp_attach_node`将节点附加到uxpool。

8.5.4.3. 参数

表 8.6. `pcp_attach_node` 参数说明

参数	描述
<code>node_id</code>	要附加的后端节点的索引。
<code>pcp_server</code>	pcp服务器的外部服务器名称。
Other arguments	请参见 pcp_common_options 。

8.5.4.4. 示例

```
test=# SELECT * FROM pcp_attach_node(1,'localhost',11001,'t-ishii','t-ishii');
node_attached
-----
t
(1 row)
```

8.5.5. `uxpool_adm_pcp_detach_node`

`uxpool_adm_pcp_detach_node` — 一个分离给定节点ID的函数。

8.5.5.1. 大纲

`pcp_detach_node` returns record(integer node_id, boolean gracefully, text host, integer port, text username, text password, out node_detached boolean);

`pcp_detach_node` returns record(integer node_id, boolean gracefully, text pcp_server, out node_detached boolean);

8.5.5.2. 描述

`pcp_detach_node`从uxpool分离一个节点。

8.5.5.3. 参数

表 8.7. `pcp_detach_node` 参数说明

参数	描述
<code>node_id</code>	要分离的后端节点的索引。
<code>gracefully</code>	如果为true, 则等待uxpool的所有会话终止。
<code>pcp_server</code>	pcp服务器的外部服务器名称。
Other arguments	请参见 pcp_common_options 。

8.5.5.4. 示例

```
test=# SELECT * FROM pcp_detach_node(1, 'false', 'localhost',11001,'t-ishii','t-ishii');
node_detached
-----
t
(1 row)
```