Python3 连接 GBase 8s 数据库示例

一、数据库服务器信息

1，数据库环境变量

```
export GBASEDBTDIR=/home/gbasedbt/gbase
export GBASEDBTSERVER=gbase01
export ONCONFIG=onconfig
export PATH=$GBASEDBTDIR/bin:$PATH
export DBDATE=Y4MD-
export DB_LOCALE=zh_CN.utf8
export CLIENT_LOCALE=zh_CN.utf8
```

2，网络连接信息

```
[gbasedbt@a02 ~]$ onstat -g ntt

GBase 8s Database Server Version 12.10.FC4G1AEE -- On-Line -- Up 00:05:51 -- 515416 Kbytes

global network information:
  #netscb connects         read        write    q-free  q-limits  q-exceed alloc/max
   2/   2        0            0            0    0/   0  135/  10    0/   0    0/  -1

Individual thread network information (times):
          netscb thread name    sid     open     read    write address
        46e0fcc8 soctcplst        3 10:44:44                      172.27.180.136|9088|soctcp
        46e09cc8 soctcppoll       2 10:44:44


[gbasedbt@a02 ~]$ more $GBASEDBTDIR/etc/sqlhosts
gbase01 onsoctcp 172.27.180.136 9088
```

二、Python3 通过 DbtPy 连接 GBase 8s 数据库-Linux

1,CSDK 安装及配置

CSDK 需要使用 root 用户权限进行安装。需要预先创建 gbasedbt 用户组及 gbasedbt 用户。

1）、创建 gbasedbt 用户组及 gbasedbt 用户

```
[root@localhost ~]# groupadd -g 1000 gbasedbt
[root@localhost ~]# useradd -g 1000 -d /home/gbasedbt/gbase -m -s /bin/bash gbasedbt
```

2）、解压缩 CSDK 软件包

```
[root@localhost ~]# mkdir csdk
[root@localhost ~]# cd csdk/
[root@localhost csdk]# tar -xvf ../clientsdk_3.0.0_1_93e040_RHLE6_x86_64.tar
installclientsdk
doc/
doc/Glsapi_machine_notes_4.10.txt
doc/ESQLC_machine_notes_4.10.txt
doc/Odbc_machine_notes_4.10.txt
doc/Libcpp_machine_notes_4.10.txt
csdk.properties
```

3）、执行静默安装，自动完成安装

```
[root@localhost csdk]#  ./installclientsdk -i silent \
  -DUSER_INSTALL_DIR=/home/gbase -DLICENSE_ACCEPTED=TRUE
```

表 5 Linux 下 CSDK 安装涉及的参数说明

序号	参数名称	示例参数值	说明信息
1	-i	silent	指定使用静默安装
2	-DUSER_INSTALL_DIR=	/home/gbase	指定安装目录
3	-DLICENSE_ACCEPTED=	TRUE	指定接受协议

CSDK 安装完成后，需要对客户端连接进行设置。以下使用 gbasedbt 用户来说明。

1）、在用户的目录下的用户环境配置文件.bash_profile 中增加数据库的环境。

根据数据库的实现情况设置：

```
# .bash_profile
export GBASEDBTDIR=/home/gbasedbt/gbase
export GBASEDBTSERVER=gbase01
export ONCONFIG=onconfig
export PATH=$GBASEDBTDIR/bin:$PATH
export
LD_LIBRARY_PAT=$GBASEDBTDIR/lib:$GBASEDBTDIR/lib/cli:$GBASEDBTDIR/lib/esql:$LD_
LIBARY_PATH
export DBDATE=Y4MD-
export DB_LOCALE=zh_CN.utf8
export CLIENT_LOCALE=zh_CN.utf8
# DbtPy 需要额外的环境变量 CSDK_HOME
# DbtPy 需要用到的环境变量包括 GBASEDBTDIR/LD_LIBRARY_PATH/PATH/CSDK_HOME
export CSDK_HOME=$GBASEDBTDIR
```

2）、修改 GBASEDBTSQLHOSTS 配置文件

在$GBASEDBTDIR/etc/目录下创建 sqlhosts（默认的 GBASEDBTSQLHOSTS）配置文件，内容为连接到数据库服务器的信息。

```
#GBASEDBTSQLHOSTS
gbase01 onsoctcp a02.gbasedbt.com 9088
```

3）、测试数据库连接

```
[gbasedbt@localhost ~]$ dbaccess - -
> connect to "testdb@gbase01" user "gbasedbt";
   输入密码：<输入用户密码>

已连接。

Elapsed time: 4.978 sec

> select dbservername from sysmaster:sysdual;

(expression)  gbase01

查询到 1 行。

Elapsed time: 0.312 sec
```

2,安装 python3 及 DbtPy

Python3 通过 DbtPy 连接到数据库，需要 python3、python3-devel 和 DbtPy。

1）、确认 python3 和 python3-devel 已经安装

```
[root@localhost python-pyodbc]# rpm -qa python3 python3-devel
python3-devel-3.6.8-13.el7.x86_64
```

```
python3-3.6.8-13.el7.x86_64
```

2）、确认安装 DbtPy，如果没有，则安装之

```
[root@a2 ~]# pip3 list --format=columns
Package    Version
---------- -------
pip        9.0.3
setuptools 39.2.0



[root@a02 ~]# wget https://gbasedbt.com/dl/DbtPy/DbtPy-3.0.5.tar.gz
[root@a02 ~]# tar -zxvf DbtPy-3.0.5.tar.gz && cd DbtPy-3.0.5
[root@a02 DbtPy-3.0.5]# python3 setup.py build
Detected 64-bit Python
Smart Triggers are not available.
running build
running build_py
creating build
creating build/lib.linux-x86_64-3.6
copying DbtPyDbi.py -> build/lib.linux-x86_64-3.6
Fixing build/lib.linux-x86_64-3.6/DbtPyDbi.py
Skipping optional fixer: buffer
Skipping optional fixer: idioms
Skipping optional fixer: set_literal
Skipping optional fixer: ws_comma
Fixing build/lib.linux-x86_64-3.6/DbtPyDbi.py
Skipping optional fixer: buffer
Skipping optional fixer: idioms
Skipping optional fixer: set_literal
Skipping optional fixer: ws_comma
running build_ext
building 'DbtPy' extension
creating build/temp.linux-x86_64-3.6
…… ……
gcc -pthread -shared -Wl,-z,relro -g build/temp.linux-x86_64-3.6/dbtpyc.o
-L/opt/csdk/lib/cli -L/root/DbtPy-3.0.5/Lib -L/usr/lib64 -lifdmr -lthcli
-lpython3.6m -o build/lib.linux-x86_64-3.6/DbtPy.cpython-36m-x86_64-linux-gnu.so

[root@a02 DbtPy-3.0.5]# python3 setup.py install
Detected 64-bit Python
Smart Triggers are not available.
running install
running bdist_egg
running egg_info
writing DbtPy.egg-info/PKG-INFO
writing dependency_links to DbtPy.egg-info/dependency_links.txt
writing top-level names to DbtPy.egg-info/top_level.txt
reading manifest file 'DbtPy.egg-info/SOURCES.txt'
reading manifest template 'MANIFEST.in'
warning: no files found matching '*.pyd'
writing manifest file 'DbtPy.egg-info/SOURCES.txt'
installing library code to build/bdist.linux-x86_64/egg
running install_lib
running build_py
running build_ext
creating build/bdist.linux-x86_64
creating build/bdist.linux-x86_64/egg
copying build/lib.linux-x86_64-3.6/DbtPyDbi.py -> build/bdist.linux-x86_64/egg
copying build/lib.linux-x86_64-3.6/DbtPy.cpython-36m-x86_64-linux-gnu.so ->
build/bdist.linux-x86_64/egg
byte-compiling build/bdist.linux-x86_64/egg/DbtPyDbi.py to
DbtPyDbi.cpython-36.pyc
creating stub loader for DbtPy.cpython-36m-x86_64-linux-gnu.so
byte-compiling build/bdist.linux-x86_64/egg/DbtPy.py to DbtPy.cpython-36.pyc
creating build/bdist.linux-x86_64/egg/EGG-INFO
copying DbtPy.egg-info/PKG-INFO -> build/bdist.linux-x86_64/egg/EGG-INFO
copying DbtPy.egg-info/SOURCES.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
copying DbtPy.egg-info/dependency_links.txt ->
build/bdist.linux-x86_64/egg/EGG-INFO
copying DbtPy.egg-info/top_level.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
writing build/bdist.linux-x86_64/egg/EGG-INFO/native_libs.txt
zip_safe flag not set; analyzing archive contents...
__pycache__.DbtPy.cpython-36: module references __file__
creating dist
creating 'dist/DbtPy-3.0.5-py3.6-linux-x86_64.egg' and adding
'build/bdist.linux-x86_64/egg' to it
removing 'build/bdist.linux-x86_64/egg' (and everything under it)
Processing DbtPy-3.0.5-py3.6-linux-x86_64.egg
creating
/usr/local/lib64/python3.6/site-packages/DbtPy-3.0.5-py3.6-linux-x86_64.egg
Extracting DbtPy-3.0.5-py3.6-linux-x86_64.egg to
/usr/local/lib64/python3.6/site-packages
Adding DbtPy 3.0.5 to easy-install.pth file

Installed
/usr/local/lib64/python3.6/site-packages/DbtPy-3.0.5-py3.6-linux-x86_64.egg
Processing dependencies for DbtPy==3.0.5
Finished processing dependencies for DbtPy==3.0.5
```

3,Python 连接测试

编写测试代码：

```
#!/usr/bin/python3
# filename: TestP3DbtPy.py

import sys
import DbtPy

print("Python DbtPy 测试程序开始运行.\n")
connectStr="PROTOCOL=onsoctcp;HOST=192.168.80.106;SERVICE=9088;SERVER=gbase01;DATABASE=testd
b;DB_LOCALE=zh_CN.utf8;CLIENT_LOCALE=zh_CN.utf8"
conn=DbtPy.connect(connectStr, "gbasedbt", "GBase123")

stmt=DbtPy.exec_immediate(conn, "drop table if exists company")

stmt=DbtPy.exec_immediate(conn, "create table company(coid serial,coname varchar(255),coaddr
varchar(255))")

stmt=DbtPy.prepare(conn,"insert into company(coname,coaddr) values(?,?)")
DbtPy.bind_param(stmt,1,"南大通用")
DbtPy.bind_param(stmt,2,"天津市普天创新园")
DbtPy.execute(stmt)
print("插入表 生效的行数: ", DbtPy.num_rows(stmt))

param="南大通用北京分公司","北京市朝阳区太阳宫",
DbtPy.execute(stmt,param)
print("插入表 生效的行数: ", DbtPy.num_rows(stmt))

'''
bool  fetch_row   : 判断是否获取到行
dict  fetch_assoc : 结果集用字段名称编号
dict  fetch_both  : 结果集使用序号和字段名称同时编号(两份数据)
tuple fetch_tuple : 获取的结果为元组
'''
# 使用 fetch_tuple
stmt=DbtPy.exec_immediate(conn, "select * from company")
tuple=DbtPy.fetch_tuple(stmt)
while tuple != False:
    print(tuple)
    tuple = DbtPy.fetch_tuple(stmt)

# 使用 fetch_both/fetch_assoc
```

```
stmt=DbtPy.exec_immediate(conn, "select * from company")
dict=DbtPy.fetch_both(stmt)
while dict != False:
    print(dict)
    print("COID: ", dict[0], " CONAME: ", dict[1], " COADDR: ", dict[2])
    dict = DbtPy.fetch_both(stmt)

# 使用 fetch_row
stmt=DbtPy.exec_immediate(conn, "select * from company")
while DbtPy.fetch_row(stmt) != False:
    print("COLD: ", DbtPy.result(stmt,0), " CONAME: ", DbtPy.result(stmt,"coname"), " COADDR: 
", DbtPy.result(stmt,"coaddr"))

DbtPy.free_result(stmt)
DbtPy.free_stmt(stmt)
DbtPy.close(conn)

print("\nPython DbtPy 测试程序结束运行.")
sys.exit(0)
```

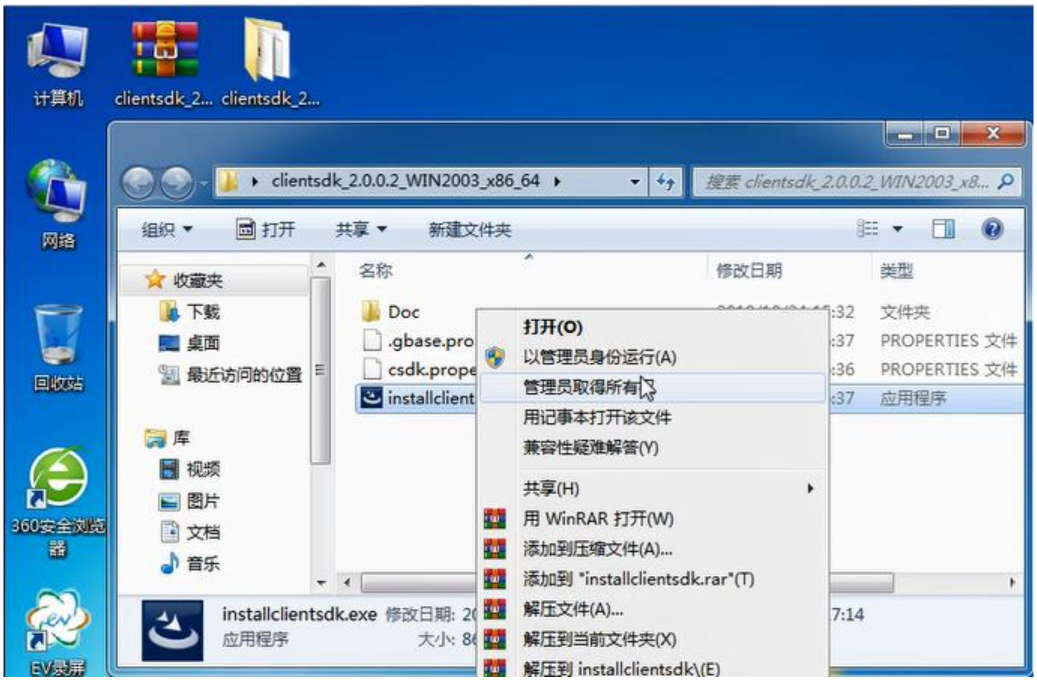
执行测试程序

```
[root@localhost py]# python3 TestPy3DbtPy.py
Python DbtPy 测试程序开始运行.

插入表 生效的行数:  1
插入表 生效的行数:  1
(1, '南大通用', '天津市普天创新园')
(2, '南大通用北京分公司', '北京市朝阳区太阳宫')
{'coid': 1, 0: 1, 'coname': '南大通用', 1: '南大通用', 'coaddr': '天津市普天创新园', 2: '天津
市普天创新园'}
COID:  1  CONAME:  南大通用  COADDR:  天津市普天创新园
{'coid': 2, 0: 2, 'coname': '南大通用北京分公司', 1: '南大通用北京分公司', 'coaddr': '北京市
朝阳区太阳宫', 2: '北京市朝阳区太阳宫'}
COID:  2  CONAME:  南大通用北京分公司  COADDR:  北京市朝阳区太阳宫
COLD:  1  CONAME:  南大通用  COADDR:  天津市普天创新园
COLD:  2  CONAME:  南大通用北京分公司  COADDR:  北京市朝阳区太阳宫

Python DbtPy 测试程序结束运行.
```

三、Python3 通过 DbtPy 连接 GBase 8s 数据库-Windows

1,CSDK 安装及配置

CSDK 需要使用管理员权限进行安装。CSDK 安装包解压，以管理员身份运行 installclientsdk.exe 开始安装。



安装完成后，在开始菜单里找到新安装的 GBase 8t Client-SDK 4.10（64-bit）目录，设置 DB_LOCALE,CLIENT_LOCALE,GBASEDBTSERVER 等环境变量（这些变量与数据库设置的变量应该一致）

在环境（Environment）选项卡，根据数据库的实现情况设置：

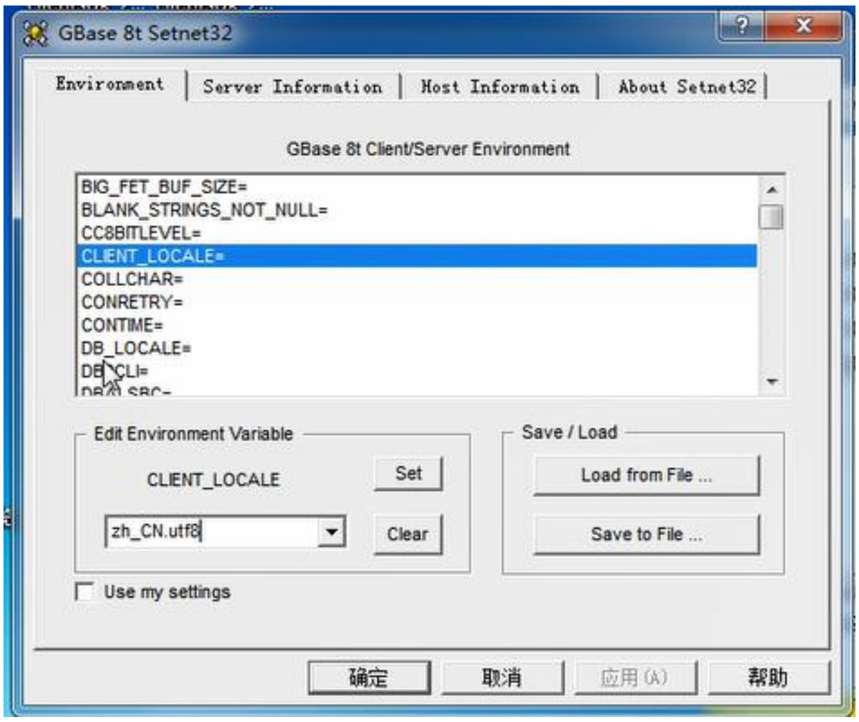
CLIENT_LOCALE：zh_CN.utf8

DB_LOCALE：zh_CN.utf8

GBASEDBTSERVER：gbase01

GL_USEGLU：1

等参数。


GBase 8t Setnet32

Environment | Server Information | Host Information | About Setnet32

GBase 8t Client/Server Environment

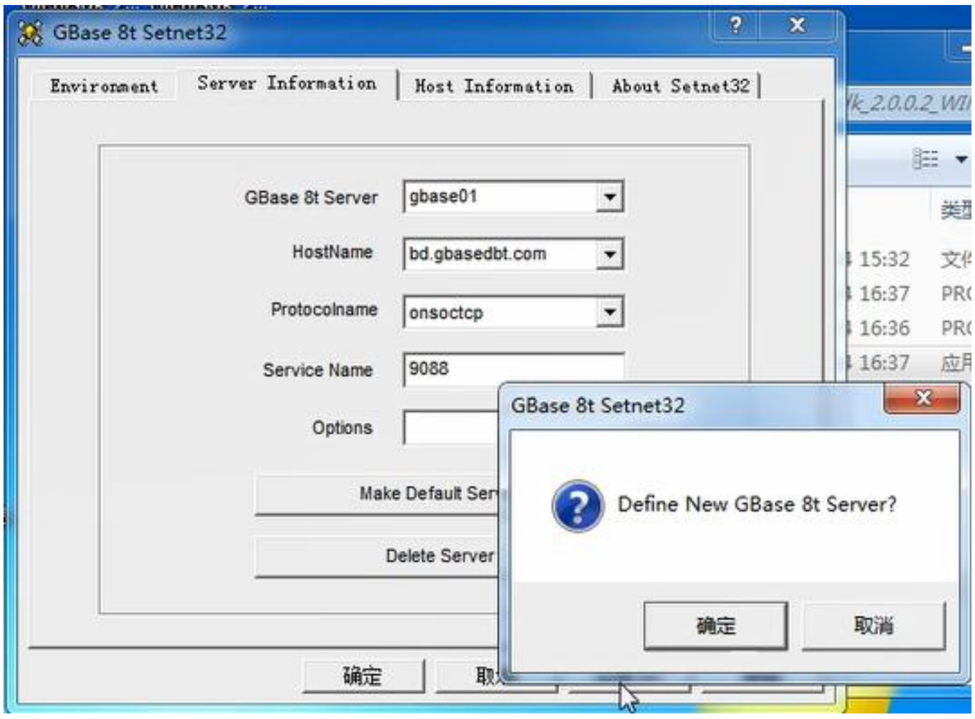
```
BIG_FET_BUF_SIZE=
BLANK_STRINGS_NOT_NULL=
CC8BITLEVEL=
CLIENT_LOCALE=
COLLCHAR=
CONRETRY=
CONTIME=
DB_LOCALE=
DBCLI=
```

Edit Environment Variable: CLIENT_LOCALE = zh_CN.utf8

在服务器信息（Server Information）选项卡中设置数据库服务器信息：

HostName：bd.gbasedbt.com	主机名或者域名
Protocolname：onsoctcp	协议名称
Service Name：9088	数据库使用的端口号

可以设置为默认的数据库服务器


GBase 8t Setnet32

Environment | Server Information | Host Information | About Setnet32

GBase 8t Server: gbase01

HostName: bd.gbasedbt.com

Protocolname: onsoctcp

Service Name: 9088

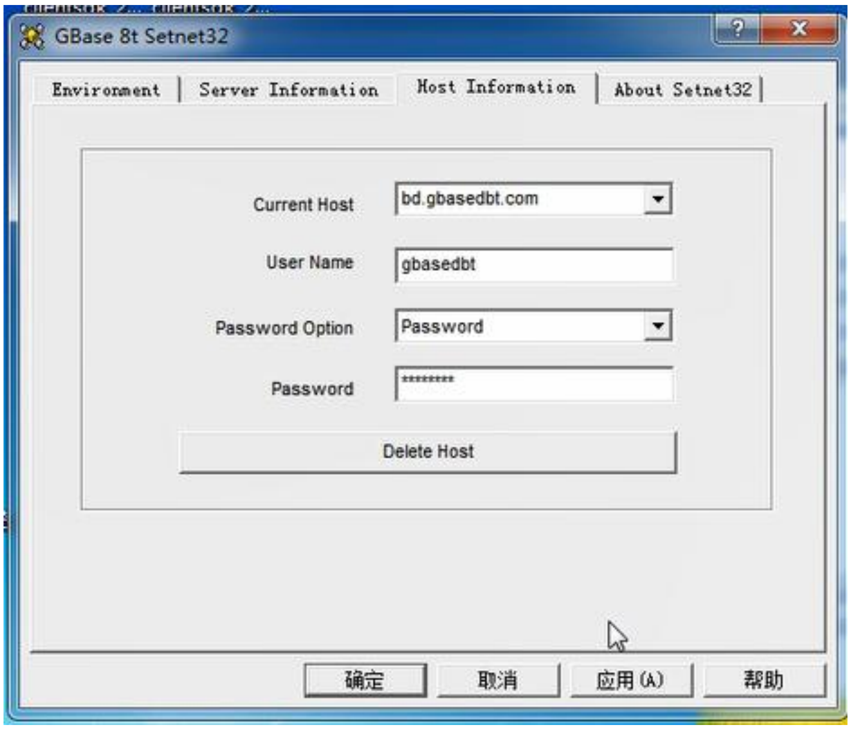
Options:

GBase 8t Setnet32

Define New GBase 8t Server?

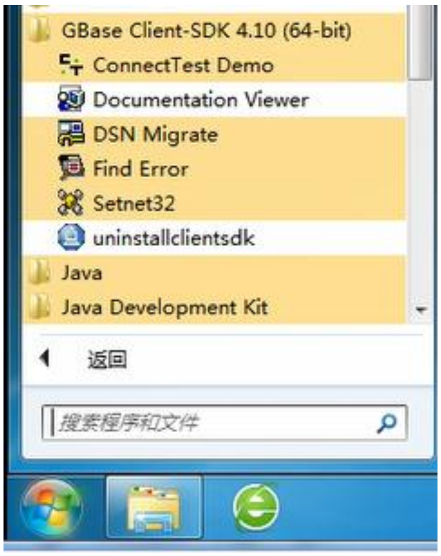
在主机信息（Host Information）选项卡中设置主机信息

设置用户、密码选项及密码

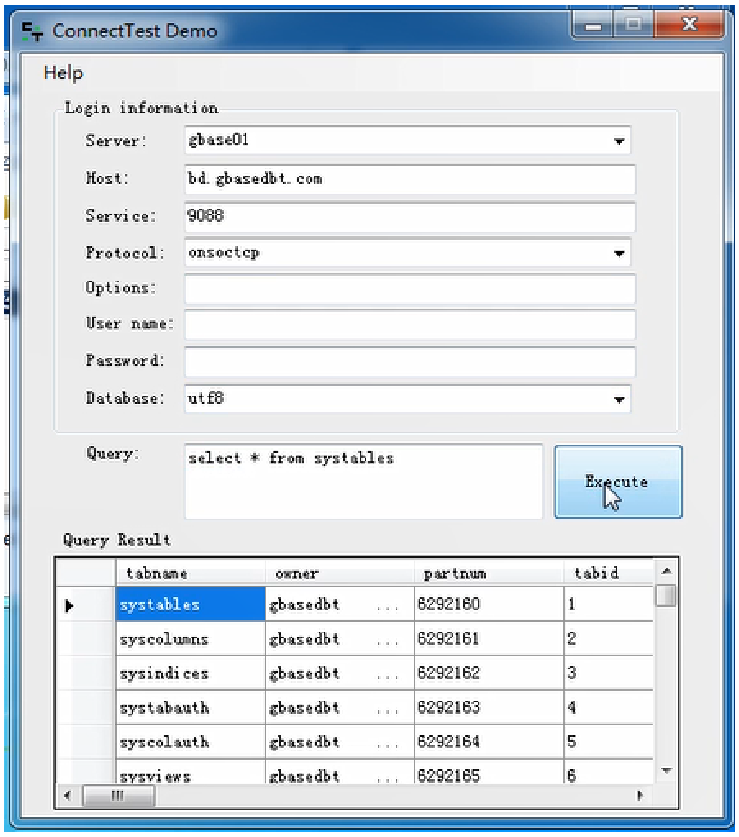


Current Host: bd.gbasedbt.com
User Name: gbasedbt
Password Option: Password
Password: ********

在开始菜单里找到 GBase Client-SDK 4.10(64-bit)目录，使用 管理员权限运行 ConnectTest Demo 进行连接测试



出现的 ConnectTest Demo 界面下，会自动加载配置好的数据库服务器信息，选择 Database（这里我们使用 utf8），填写测试语句 select * from systables 进行测试，能获取到数据即为成功。



Login information

Server:	gbase01
Host:	bd.gbasedbt.com
Service:	9088
Protocol:	onsoctcp
Options:	
User name:	
Password:	
Database:	utf8

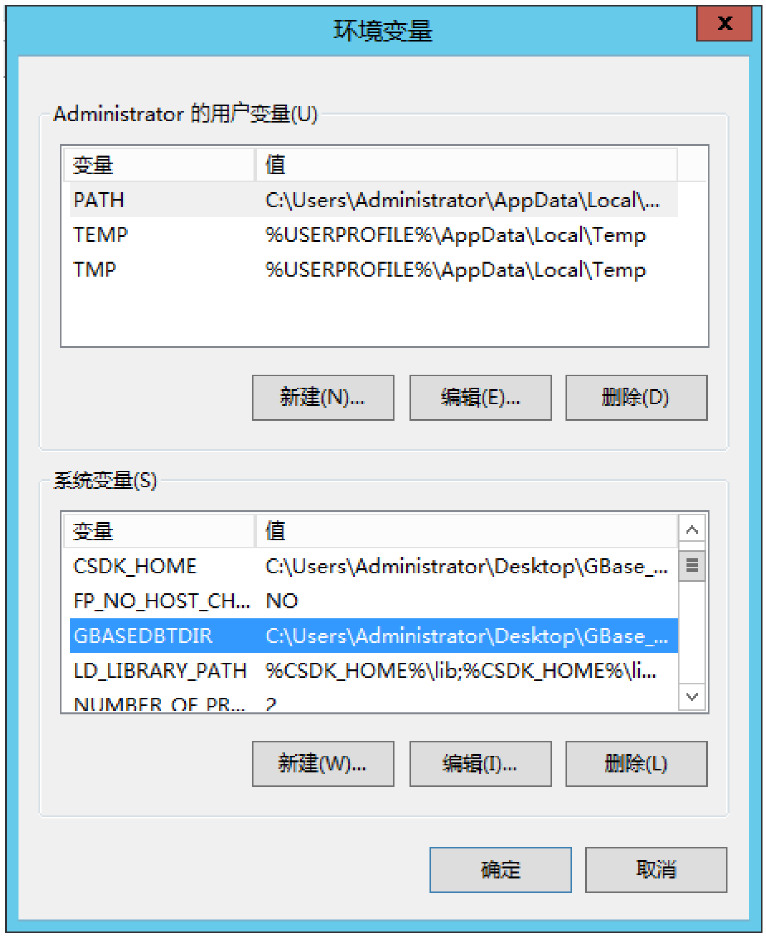
Query:

```
select * from systables
```

Query Result

tabname	owner	partnum	tabid
systables	gbasedbt ...	6292160	1
syscolumns	gbasedbt ...	6292161	2
sysindices	gbasedbt ...	6292162	3
systabauth	gbasedbt ...	6292163	4
syscolauth	gbasedbt ...	6292164	5
sysviews	gbasedbt ...	6292165	6

环境变量配置



Administrator 的用户变量(U)

变量	值
PATH	C:\Users\Administrator\AppData\Local\...
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

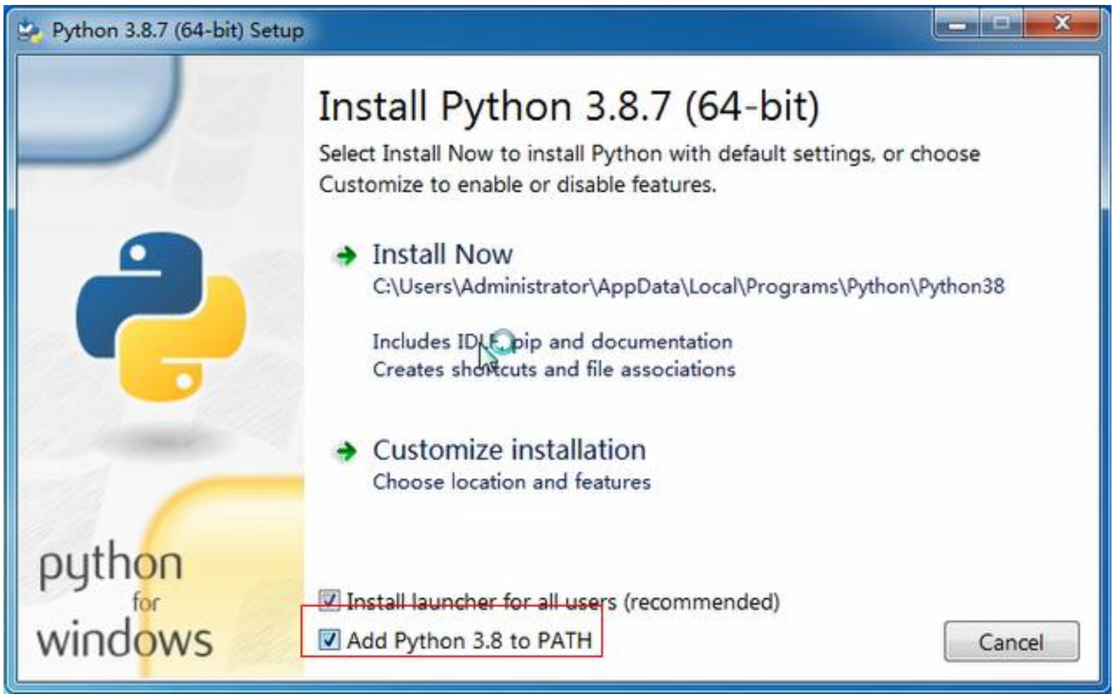
系统变量(S)

变量	值
CSDK_HOME	C:\Users\Administrator\Desktop\GBase_...
FP_NO_HOST_CH...	NO
GBASEDBTDIR	C:\Users\Administrator\Desktop\GBase_...
LD_LIBRARY_PATH	%CSDK_HOME%\lib;%CSDK_HOME%\li...
NUMBER_OF_PR...	2

在系统变量中增加以下内容
GBASEDBTDIR 值为 CSDK 的安装目录，如：D:\GBase_CSDK_3.0.0_1_Win_x86_64
LD_LIBRARY_PATH 值
为 %GBASEDBTDIR%\lib;%GBASEDBTDIR%\lib\esql;%GBASEDBTDIR%\lib\cli;%LD_LIBRARY_PATH%
CSDK_HOME 值为 CSDK 的安装目录

2,安装 python3 及 DbtPy

使用管理员权限安装 python3，勾选上 Add Python 3.8 to PATH


Python 3.8.7 (64-bit) Setup

Install Python 3.8.7 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

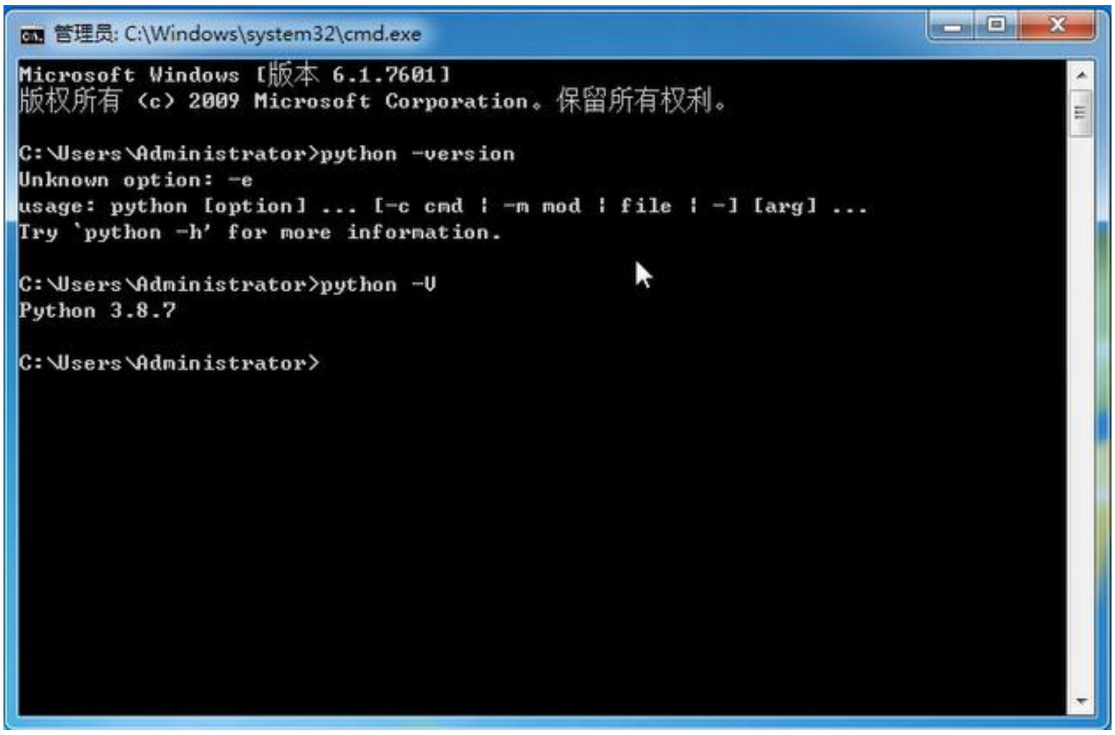
Install Now
C:\Users\Administrator\AppData\Local\Programs\Python\Python38

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

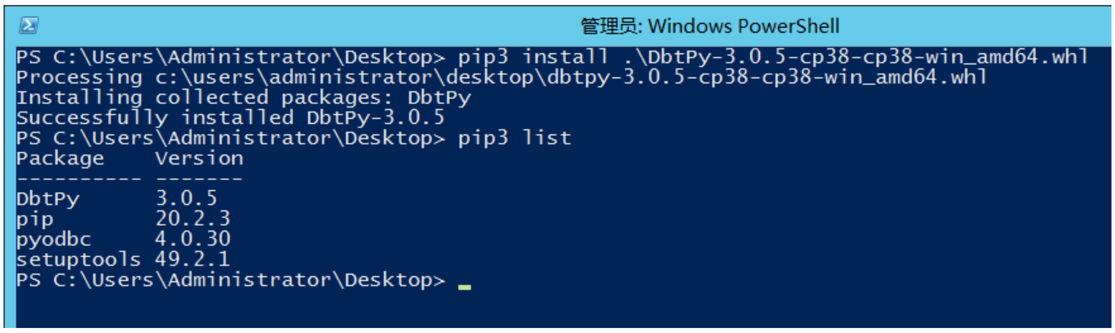
☑ Install launcher for all users (recommended)
☑ Add Python 3.8 to PATH

安装完成后，检查是否安装



```
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python -version
Unknown option: -e
usage: python [option] ... [-c cmd | -m mod | file | -] [arg] ...
Try `python -h' for more information.

C:\Users\Administrator>python -V
Python 3.8.7

C:\Users\Administrator>
```

下载 DbtPy，地址
https://gbasedbt.com/dl/DbtPy/DbtPy-3.0.5-cp38-cp38-win_amd64.whl
通过 pip3 安装 DbtPy



```
PS C:\Users\Administrator\Desktop> pip3 install .\DbtPy-3.0.5-cp38-cp38-win_amd64.whl
Processing c:\users\administrator\desktop\dbtpy-3.0.5-cp38-cp38-win_amd64.whl
Installing collected packages: DbtPy
Successfully installed DbtPy-3.0.5
PS C:\Users\Administrator\Desktop> pip3 list
Package    Version
---------- -------
DbtPy      3.0.5
pip        20.2.3
pyodbc     4.0.30
setuptools 49.2.1
PS C:\Users\Administrator\Desktop> _
```

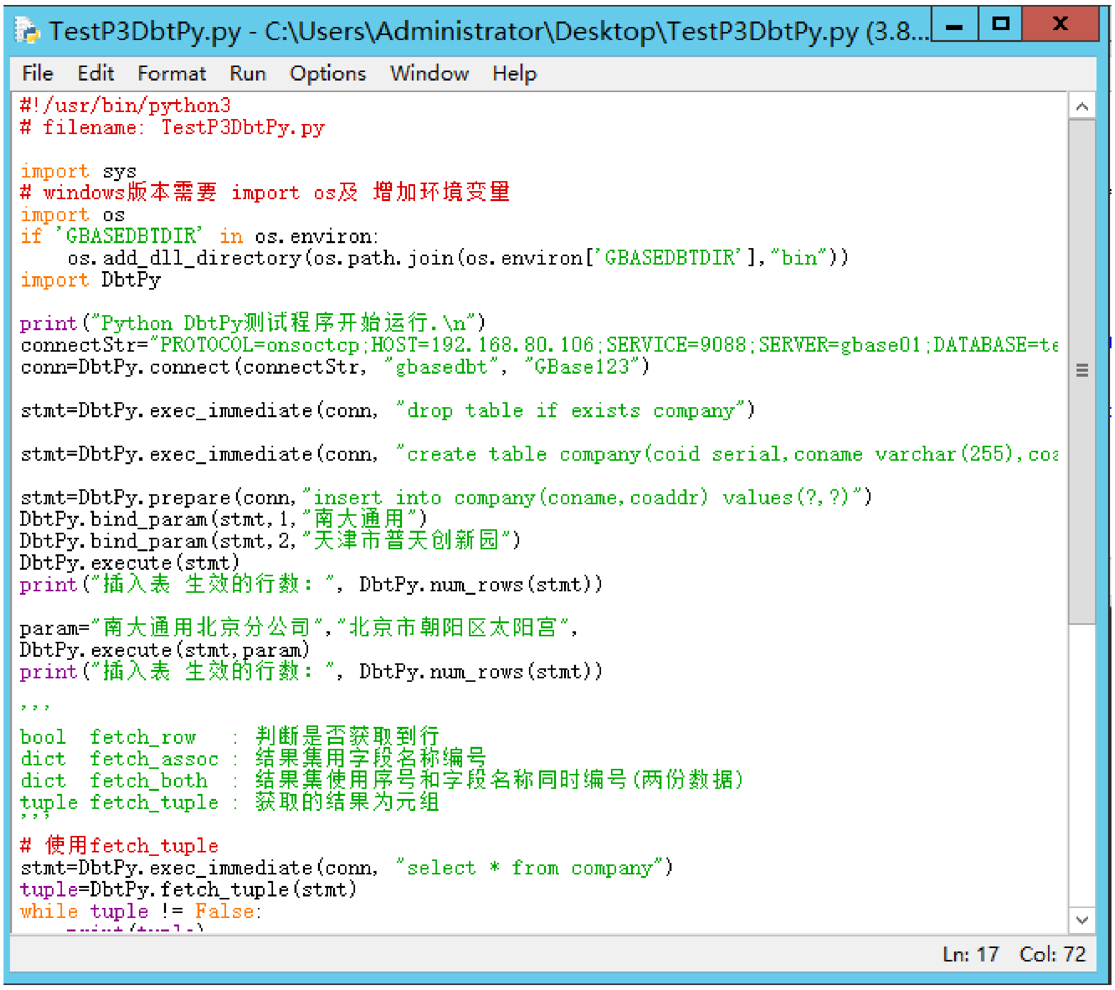
3,Python 连接测试(DbtPy)

编写测试代码：

```
#!/usr/bin/python3
# filename: TestP3DbtPy.py

import sys
# windows 版本需要 import os 及 增加环境变量
import os
if 'GBASEDBTDIR' in os.environ:
    os.add_dll_directory(os.path.join(os.environ['GBASEDBTDIR'],"bin"))
import DbtPy

print("Python DbtPy 测试程序开始运行.\n")
```

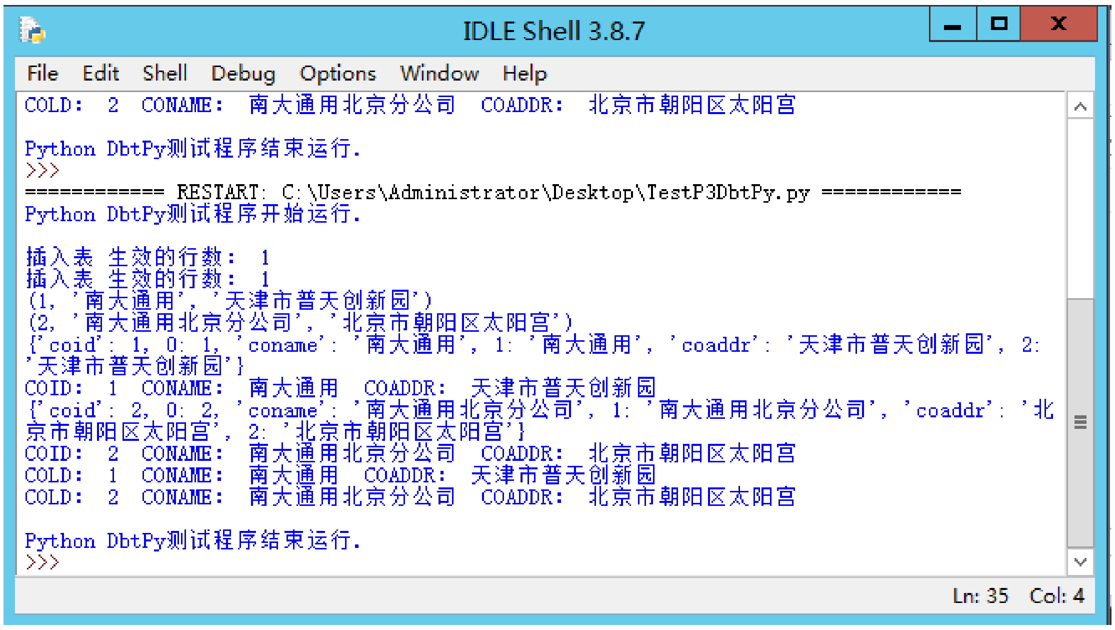
```
connectStr="PROTOCOL=onsoctcp;HOST=192.168.80.106;SERVICE=9088;SERVER=gbase01;DATABASE=testd
b;DB_LOCALE=zh_CN.utf8;CLIENT_LOCALE=zh_CN.utf8"
conn=DbtPy.connect(connectStr, "gbasedbt", "GBase123")

stmt=DbtPy.exec_immediate(conn, "drop table if exists company")

stmt=DbtPy.exec_immediate(conn, "create table company(coid serial,coname varchar(255),coaddr
varchar(255))")

stmt=DbtPy.prepare(conn,"insert into company(coname,coaddr) values(?,?)")
DbtPy.bind_param(stmt,1,"南大通用")
DbtPy.bind_param(stmt,2,"天津市普天创新园")
DbtPy.execute(stmt)
print("插入表 生效的行数: ", DbtPy.num_rows(stmt))

param="南大通用北京分公司","北京市朝阳区太阳宫",
DbtPy.execute(stmt,param)
print("插入表 生效的行数: ", DbtPy.num_rows(stmt))

'''
bool  fetch_row   : 判断是否获取到行
dict  fetch_assoc : 结果集用字段名称编号
dict  fetch_both  : 结果集使用序号和字段名称同时编号(两份数据)
tuple fetch_tuple : 获取的结果为元组
'''
# 使用 fetch_tuple
stmt=DbtPy.exec_immediate(conn, "select * from company")
tuple=DbtPy.fetch_tuple(stmt)
while tuple != False:
    print(tuple)
    tuple = DbtPy.fetch_tuple(stmt)

# 使用 fetch_both/fetch_assoc
stmt=DbtPy.exec_immediate(conn, "select * from company")
dict=DbtPy.fetch_both(stmt)
while dict != False:
    print(dict)
    print("COID: ", dict[0], " CONAME: ", dict[1], " COADDR: ", dict[2])
    dict = DbtPy.fetch_both(stmt)

# 使用 fetch_row
stmt=DbtPy.exec_immediate(conn, "select * from company")
while DbtPy.fetch_row(stmt) != False:
    print("COLD: ", DbtPy.result(stmt,0), " CONAME: ", DbtPy.result(stmt,"coname"), " COADDR:
```

```
", DbtPy.result(stmt,"coaddr"))


DbtPy.free_result(stmt)
DbtPy.free_stmt(stmt)
DbtPy.close(conn)


print("\nPython DbtPy 测试程序结束运行.")
sys.exit(0)
```


TestP3DbtPy.py - C:\Users\Administrator\Desktop\TestP3DbtPy.py (3.8...
File Edit Format Run Options Window Help
Ln: 17 Col: 72

```
#!/usr/bin/python3
# filename: TestP3DbtPy.py

import sys
# windows版本需要 import os及 增加环境变量
import os
if 'GBASEDBTDIR' in os.environ:
    os.add_dll_directory(os.path.join(os.environ['GBASEDBTDIR'],"bin"))
import DbtPy

print("Python DbtPy测试程序开始运行.\n")
connectStr="PROTOCOL=onsoctcp;HOST=192.168.80.106;SERVICE=9088;SERVER=gbase01;DATABASE=te
conn=DbtPy.connect(connectStr, "gbasedbt", "GBase123")

stmt=DbtPy.exec_immediate(conn, "drop table if exists company")

stmt=DbtPy.exec_immediate(conn, "create table company(coid serial,coname varchar(255),coa

stmt=DbtPy.prepare(conn,"insert into company(coname,coaddr) values(?,?)")
DbtPy.bind_param(stmt,1,"南大通用")
DbtPy.bind_param(stmt,2,"天津市普天创新园")
DbtPy.execute(stmt)
print("插入表 生效的行数：", DbtPy.num_rows(stmt))

param="南大通用北京分公司","北京市朝阳区太阳宫",
DbtPy.execute(stmt,param)
print("插入表 生效的行数：", DbtPy.num_rows(stmt))

'''
bool  fetch_row   : 判断是否获取到行
dict  fetch_assoc : 结果集用字段名称编号
dict  fetch_both  : 结果集使用序号和字段名称同时编号(两份数据)
tuple fetch_tuple : 获取的结果为元组
'''
# 使用fetch_tuple
stmt=DbtPy.exec_immediate(conn, "select * from company")
tuple=DbtPy.fetch_tuple(stmt)
while tuple != False:
```

执行测试



```
COLD:  2  CONAME:  南大通用北京分公司  COADDR:  北京市朝阳区太阳宫

Python DbtPy测试程序结束运行.
>>>
============ RESTART: C:\Users\Administrator\Desktop\TestP3DbtPy.py ============
Python DbtPy测试程序开始运行.

插入表 生效的行数:  1
插入表 生效的行数:  1
(1, '南大通用', '天津市普天创新园')
(2, '南大通用北京分公司', '北京市朝阳区太阳宫')
{'coid': 1, 0: 1, 'coname': '南大通用', 1: '南大通用', 'coaddr': '天津市普天创新园', 2:
'天津市普天创新园'}
COID:  1  CONAME:  南大通用  COADDR:  天津市普天创新园
{'coid': 2, 0: 2, 'coname': '南大通用北京分公司', 1: '南大通用北京分公司', 'coaddr': '北
京市朝阳区太阳宫', 2: '北京市朝阳区太阳宫'}
COID:  2  CONAME:  南大通用北京分公司  COADDR:  北京市朝阳区太阳宫
COLD:  1  CONAME:  南大通用  COADDR:  天津市普天创新园
COLD:  2  CONAME:  南大通用北京分公司  COADDR:  北京市朝阳区太阳宫

Python DbtPy测试程序结束运行.
>>>
```