

◁ BIT ▷

高斯数据库技术实验指导

时间：2025/3/21

德以明理 学以精工



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY



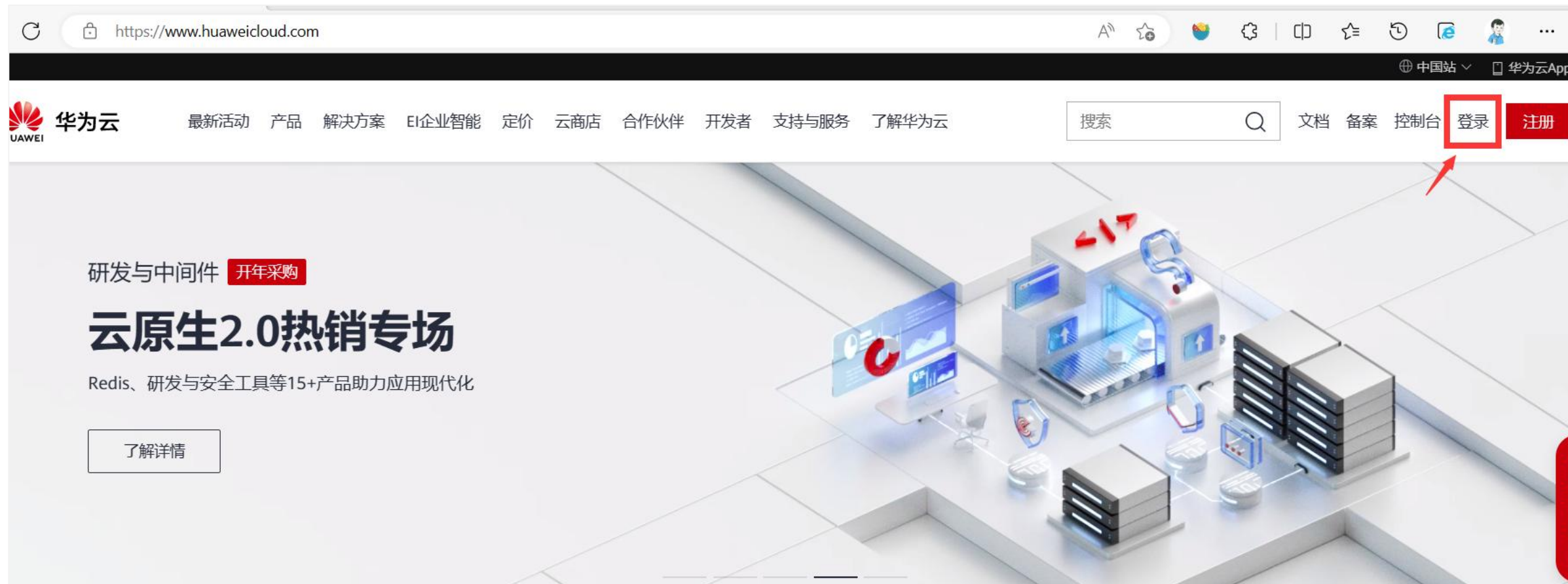
目录 | CONTENTS

- 1 购买弹性云服务器ECS
- 2 安装openGauss数据库
- 3 使用工具连接ECS
- 4 使用简单的SQL语句



1 购买弹性云服务器ECS

进入华为云官网 <https://www.huaweicloud.com/>，单击登录。



输入账号名和密码，单击登录。
如果还没有注册，单击注册，按步骤注册后进行登录。

扫码登录

密码登录

华为帐号登录

手机号/邮件地址/帐号名/原华为云帐号

密码

登录

注册

忘记密码

使用其他帐号登录

IAM用户

企业联邦用户

华为官网帐号

华为企业合作伙伴

华为云帐号

华为帐号注册

国家/地区

中国

+86(中国)

手机号

短信验证码

获取验证码

密码

确认密码

华为帐号服务需要联网，并获取您的帐号、所在区域、浏览器设置信息，以及您主动上传的个人基本资料和身份信息。点击“注册”，即表示您同意上述内容及[华为帐号用户协议](#)、[关于华为帐号与隐私的声明](#)。

注册

在华为云主页 (<https://www.huaweicloud.com/>) 点击产品，选择 " 精选推荐 " 下的 " 计算 "，再选择 " 弹性云服务器ECS "。



The screenshot shows the Huawei Cloud website interface. At the top, the navigation bar includes "华为云", "最新活动", "产品" (highlighted with a red box and a red '1'), "解决方案", "EI企业智能", "定价", "云商店", "合作伙伴", "开发者", "支持与服务", and "了解华为云". A search bar and "文档" and "备案" links are on the right. On the left sidebar, "计算" (highlighted with a red box and a red '2') is selected under the "精选推荐" section. The main content area is titled "计算" and lists various services. "弹性云服务器 ECS HOT" (highlighted with a red box and a red '3') is the first item, with a description "可随时自动获取、弹性伸缩的云服务器". Other services listed include "云耀云服务器 HECS HOT", "GPU加速云服务器 GACS", "裸金属服务器 BMS", "弹性伸缩 AS", "Huawei Cloud EulerOS NEW", "FPGA加速云服务器 FACS", "专属主机 DeH", and "函数工作流 FunctionGraph".

产品	描述
弹性云服务器 ECS HOT	可随时自动获取、弹性伸缩的云服务器
云耀云服务器 HECS HOT	简单上云第一步
GPU加速云服务器 GACS	提供GPU计算资源的弹性云服务器
裸金属服务器 BMS	高性能、高安全的云上物理服务器
弹性伸缩 AS	根据预设策略自动调整计算资源的服务
Huawei Cloud EulerOS NEW	充分发挥华为云优势的操作系统
FPGA加速云服务器 FACS	提供FPGA计算资源的弹性云服务器
专属主机 DeH	专属物理主机创建的云服务器
函数工作流 FunctionGraph	自动运行代码，无需配置或管理服务器

进入弹性云服务器ECS购买界面。

弹性云服务器 ECS

弹性云服务器（Elastic Cloud Server, ECS）是一种云上可随时自助获取、可弹性伸缩的计算服务，可帮助您打造安全、可靠、灵活、高效的应用环境。

立即购买

管理控制台

帮助文档

- 【有奖征文】云耀云服务器（HECS）有奖征文来袭，分享你的上云故事和技术经验，赢千元大礼包！



购买弹性云服务器ECS



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

进行ECS基础配置。完成后，单击“下一步：网络配置”。

购买弹性云服务器

放

1 基础配置

2 网络配置

3 高级配置

4 确认配置

区域

华北-北京四

推荐区域

华北-乌兰察布一

西南-贵阳一

华北-北京四 (2)

华南-广州

华东-上海一

不同区域的云服务产品之间内网互不相通；请就近选择靠近您业务的区域，可减少网络时延，提高访问速度。[如何选择区域](#)

计费模式

包年/包月

按需计费

竞价计费

可用区

随机分配

可用区1

可用区2

可用区3

可用区7

实例筛选

规格类型选型

业务场景选型

CPU架构

x86计算

鲲鹏计算

规格

最新系列

vCPUs

2vCPUs

内存

4GiB

规格名称

隐藏售罄的规格

实例类型	规格名称	vCPUs	内存	CPU	基准 / 最大带宽	内网收发包	规格参考
<input checked="" type="radio"/> 鲲鹏通用计算增强...	kc1.large.2	2vCPUs	4GiB	Huawei Kunpeng 9...	0.8 / 3 Gbit/s	30万PPS	¥0.30/小时

购买弹性云服务器ECS



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

进行ECS网络配置。完成后，单击“下一步：高级配置”。

购买弹性云服务器

1 基础配置

2 网络配置

3 高级配置

4 确认配置

网络

vpc-default(192.168.0.0/16)

C

subnet-default(192.168.0.0/24)

C

自动分配IP地址

可用私有IP数量248个

如需创建新的虚拟私有云，您可前往控制台创建。

扩展网卡

+

增加一块网卡 您还可以增加 1 块网卡

弹性公网IP

☒ 现在购买

☐ 使用已有

☐ 暂不购买

线路

全动态BGP

静态BGP

不低于99.95%可用性保障

公网带宽

按带宽计费

流量较大或较稳定的场景

按流量计费

流量小或流量波动较大场景

加入共享带宽

多业务流量错峰分布场景

指定带宽上限，按实际使用的出公网流量计费，与使用时间无关。

带宽大小

5

10

20

50

100

自定义

-

5

+

带宽范围：1-300 Mbit/s

免费开启DDoS基础防护

购买弹性云服务器ECS



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

记住用户名为root，输入自定义密码和确认密码，其余默认，单击“下一步：确认配置”。

< 购买弹性云服务器

1 基础配置

2 网络配置

3 高级配置

4 确认配置

云服务器名称

ecs-7cf0

☐ 允许重名

购买多台云服务器时，支持自动增加数字后缀命名或者自定义规则命名。

描述

0/85

登录凭证

密码

密钥对

创建后设置

用户名

root

密码

请牢记密码，如忘记密码可登录ECS控制台重置密码。

.....

确认密码

.....

购买弹性云服务器ECS



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

确认设置信息，尤其是配置费用，然后勾选协议，单击“立即购买”。

配置

基础配置

计费模式

按需计费

区域

北京四

规格

鲲鹏通用计算增强型 | kc1.large.2 | 2vCPUs | 4... 镜像

openEuler 20.03 64bit with ARM

系统盘

通用型SSD, 40GiB

网络配置

虚拟私有云

vpc-default(192.168.0.0/16)

安全组

Sys-WebServer

弹性公网IP

全动态BGP | 计费方式: 按流量计费 | 带宽: 5 M...

高级配置

云服务器名称

ecs-7cf0

登录凭证

密码

生成Open API最佳实践脚本

购买数量

-

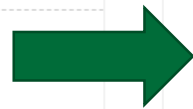
1

+

您最多可以创建198台云服务器。 申请更多云服务器配额请单击[申请扩大配额](#)。

协议

☒ 我已经阅读并同意《[镜像免责声明](#)》



任务提交成功!

您的弹性云服务器ecs-7cf0已经开始创建。

返回购买

返回云服务器列表

查看云服务器列表。状态列显示“运行中”，则表示购买成功。

我的ECS: 华北-北京四 (3) C

开机 关机 重置密码 更多 ▾ C ⚙️ ↗️ ⌵ ☰

🔍 默认按照名称搜索 ⚙️ Q

<input type="checkbox"/>	名称/ID ▾	监控	安全	可用区 ▾	状态 ▾	规格/镜像	IP地址	计费模式 ▾	标签	操作
<input type="checkbox"/>	ecs-7cf0 08cdc88b-4994-4906-...			可用区7	 运行中	2vCPUs 4Gi... openEuler 20....	119.3.187... 192.168.0...	按需计费 2023/03/29 1...	--	远程登录 更多 ▾



2 安装openGauss数据库

使用“CloudShell登录”连接ECS



对购买好的ECS服务器，点击“更多”、“开机”，开机之后，点击“远程登录”。

<input type="checkbox"/>	名称/ID	监控	安全	可用区	状态	规格/镜像	IP地址	计费模式	标签	操作
<input type="checkbox"/>	ecs-7cf0 08cdc88b-4994-4906-...			可用区7	关机	2vCPUs 4Gi... openEuler 20...	119.3.187... 192.168.0...	按需计费 2023/03/29 1...	--	远程登录 1 更多 ▲
<input type="checkbox"/>	ecs-74da ...			可用区7	关机	2vCPUs 4Gi... ...	117.78.9... ...	按需计费 2022/10/25 1...	--	2 购买相同配置 开机

开机

确定要对以下1台云服务器进行开机操作吗？

名称	状态	备注
ecs-7cf0	关机	--

是

否

使用“CloudShell登录”连接ECS



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

点击“远程登录”之后，点击“CloudShell登录”。

登录Linux弹性云服务器

使用CloudShell登录 **New!**

[登录不上?](#)

请确保安全组已放通CloudShell连接实例使用的端口（默认使用22端口）

优势：操作更流畅，命令支持复制粘贴，支持浏览输出历史和多终端分区布局。[了解更多](#)

CloudShell登录

其他方式

1、使用控制台提供的VNC方式登录

立即登录

使用“CloudShell登录”连接ECS

等加载完成后，输入root用户密码后点击“连接”按钮即可登录。

* 区域：华北-北京四

刷新

* 云服务器：ecs-7cf0

☐ 119.3.187.82 (公网)

☐ 192.168.0.130 (私网)

端口：22

* 用户名：root

* 认证方式：密码认证

* 密码：.....

会话名称：root@119.3.187.82

☒ 打开远程主机文件树

注意：

- 为确保连接的安全性，系统将对超过 20分钟 没有活跃的会话进行自动断开。

- 请确认安全组中来源为CloudShell代理IP的远程端口（SSH默认端口为22）已经允许。

- 当远程登录后操作卡顿时，建议查看一下机器的CPU、内存情况，请定义云监控在主机异常时通过短信等多种方式通知。

- 华为云CloudShell不会保存您的密码，请妥善保管以防丢失。

连接

取消

设置字符集参数： 将各数据库节点的字符集设置为相同的字符集。可以在 /etc/profile 文件中添加 “export LANG=XXX”（XXX为Unicode编码），并使配置修改生效。

```
[root@ecs-58ab ~]# cat >> /etc/profile << EOF
>
> export LANG=en_US.UTF-8
> EOF
[root@ecs-58ab ~]# source /etc/profile

Welcome to 4.19.90-2003.4.0.0036.oe1.aarch64

System information as of time:  Wed Oct  5 11:06:23 CST 2022

System load:      0.08
Processes:        116
Memory used:      10.5%
Swap used:        0.0%
Usage On:         9%
IP address:       192.168.0.75
Users online:     1
```

修改python版本并安装libaio包：之后安装过程中需要用到Python-3.7.x命令，但是默认Python版本为Python-2.7.x，所以需要切换Python版本。

输入下面五条命令，分别进行：

- 1.进入/usr/bin目录
- 2.备份python文件
- 3.建立Python3软连接
- 4.验证Python版本
- 5.下载安装libaio包

```
[root@ecs-58ab ~]# cd /usr/bin
[root@ecs-58ab bin]# mv python python.bak
[root@ecs-58ab bin]# ln -s python3 /usr/bin/python
[root@ecs-58ab bin]# python -V
Python 3.7.4
[root@ecs-58ab bin]# yum install libaio* -y
```

以root用户登录待安装openGauss的主机，并创建存放安装包的目录。

```
[root@ecs-58ab bin]# mkdir -p /opt/software/openGauss
```

openGauss安装用户omm须具有/opt/software/openGauss目录的读写权限，所以给该目录赋权限。

```
[root@ecs-58ab bin]# chmod 755 -R /opt/software
```

切换到安装目录。

```
[root@ecs-58ab bin]# cd /opt/software/openGauss
```

使用wget下载数据库安装包到安装目录：**wget https://opengauss.obs.cn-south-1.myhuaweicloud.com/2.0.0/arm/openGauss-2.0.0-openEuler-64bit-all.tar.gz**

```
[root@ecs-58ab openGauss]# wget https://opengauss.obs.cn-south-1.myhuaweicloud.com/2.0.0/arm/openGauss-2.0.0-openEuler-64bit-all.tar.gz
```

XML文件包含部署openGauss的服务器信息、安装路径、IP地址以及端口号等，用于告知openGauss如何部署。

这里以单节点配置的方案为例，说明如何创建XML配置文件。

以root用户登录待安装openGauss的主机，切换到存放安装包的目录。
创建XML配置文件，用于数据库安装。

```
[root@ecs-58ab openGauss]# cd /opt/software/openGauss  
[root@ecs-58ab openGauss]# vi clusterconfig.xml
```


输入 “i” 进入INSERT模式，添加文本（共29行）如下：

```
1.<?xml version="1.0" encoding="UTF-8"?>
2.<ROOT>
3.    <CLUSTER>
4.        <PARAM name="clusterName" value="dbCluster" />
5.        <PARAM name="nodeNames" value="ecs-7cf0" />
6.        <PARAM name="backIp1s" value="192.168.0.130" />
7.        <PARAM name="gaussdbAppPath" value="/opt/gaussdb/app" />
8.        <PARAM name="gaussdbLogPath" value="/var/log/gaussdb" />
9.        <PARAM name="gaussdbToolPath" value="/opt/huawei/wisquery" />
10.        <PARAM name="corePath" value="/opt/opengauss/corefile"/>
11.        <PARAM name="clusterType" value="single-inst"/>
12.    </CLUSTER>
13.
```

这里的nodeNames、backIp1s改为自己的ECS服务器的主机名和**私有**IP地址。

添加文本如下。点击 " Esc " 退出INSERT模式，然后输入 " :wq " 后回车退出编辑并保存文本。

```
14.    <DEVICELIST>
15.
16.        <DEVICE sn="1000001">
17.            <PARAM name="name" value="ecs-7cf0"/>
18.            <PARAM name="azName" value="AZ1"/>
19.            <PARAM name="azPriority" value="1"/>
20.            <PARAM name="backIp1" value="192.168.0.130"/>
21.            <PARAM name="sshIp1" value="192.168.0.130"/>
22.
23.        <!--dbnode-->
24.        <PARAM name="dataNum" value="1"/>
25.        <PARAM name="dataPortBase" value="26000"/>
26.        <PARAM name="dataNode1" value="/gaussdb/data/db1"/>
27.        </DEVICE>
28.    </DEVICELIST>
29.</ROOT>
```

使用vi打开文件 “/etc/profile.d/performance.sh” 。

```
[root@ecs-58ab openGauss]# vi /etc/profile.d/performance.sh
```

输入 “i” ，进入INSERT模式。用#注释 **sysctl -w vm.min_free_kbytes =112640 &> /dev/null** 这行内容。

```
#!/bin/bash
#Copyright (c) [2019] Huawei Technologies Co., Ltd.
#generic-release is licensed under the Mulan PSL v1.
#You can use this software according to the terms and conditions of the Mulan PSL v1.
#You may obtain a copy of Mulan PSL v1 at:
#    http://license.coscl.org.cn/MulanPSL
#THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OF ANY KIND, EITHER
#IMPLIED, INCLUDING BUT NOT LIMITED TO NON-INFRINGEMENT, MERCHANTABILITY OR FIT FOR A
#PARTICULAR
#PURPOSE.
#See the Mulan PSL v1 for more details.

CPUNO=`cat /proc/cpuinfo|grep processor|wc -l`
export GOMP_CPU_AFFINITY=0-[$CPUNO - 1]

#sysctl -w vm.min_free_kbytes=112640 &> /dev/null
sysctl -w vm.dirty_ratio=60 &> /dev/null
sysctl -w kernel.sched_autogroup_enabled=0 &> /dev/null

umask 0077
```

为确保openssl版本正确，执行预安装前加载安装包中lib库。需要执行以下命令：

1.使用vi打开文件/etc/profile。

```
[root@ecs-58ab openGauss]# vi /etc/profile
```

2.在文件底部添加以下内容。

```
export packagePath=/opt/software/openGauss  
export LD_LIBRARY_PATH=$packagePath/script/gspylib/clib:$LD_LIBRARY_PATH
```

3.配置完成后，使用命令 **source /etc/profile** 使设置生效。

在安装目录下，解压以下两个安装包

openGauss-2.0.0-openEuler-64bit-all.tar.gz

openGauss-2.0.0-openEuler-64bit-om.tar.gz

```
[root@ecs-58ab openGauss]# tar -zxvf openGauss-2.0.0-openEuler-64bit-om.tar.gz
```

```
[root@ecs-58ab openGauss]# tar -zxvf openGauss-2.0.0-openEuler-64bit-all.tar.gz
```

安装包解压后，会在/opt/software/openGauss路径下自动生成script子目录，并且在script目录下生成gs_preinstall等各种OM工具脚本。

```
[root@ecs-58ab script]# cd /opt/software/openGauss/script/
[root@ecs-58ab script]# ls
config          gs_checkperf  gs_install    gspylib       gs_upgradectl  local
gs_backup       gs_collector  gs_om         gs_ssh         impl           transfer.py
gs_check        gs_dropnode   gs_postuninstall gs_sshexkey    __init__.py
gs_checkos      gs_expansion  gs_preinstall gs_uninstall   killall
```

切换到script目录，输入以下命令来创建openGauss omm用户互信：

python gs_preinstall -U omm -G dbgrp -X /opt/software/openGauss/clusterconfig.xml

命令中的omm为操作系统用户，
dbgrp为运行openGauss的操作系统用户的群组名称，
/opt/software/openGauss/clusterconfig.xml为openGauss配置文件路径。

```
[root@ecs-58ab script]# python gs_preinstall -U omm -G dbgrp -X /opt/software/openGauss/clusterconfig.xml
Parsing the configuration file.
Successfully parsed the configuration file.
Installing the tools on the local node.
Successfully installed the tools on the local node.
Setting pssh path
Successfully set core path.
Are you sure you want to create the user[omm] and create trust for it (yes/no)?
```

这里提示你是否创建互信。填写 yes。

修改/opt/software/openGauss/script文件的权限，登录到openGauss的主机，并切换到omm用户。

```
[root@ecs-58ab script]# chmod -R 755 /opt/software/openGauss/script
[root@ecs-58ab script]# su - omm
Last login: Wed Oct  5 11:34:00 CST 2022 on pts/0

Welcome to 4.19.90-2003.4.0.0036.oel.aarch64

System information as of time:  Wed Oct  5 11:37:15 CST 2022

System load:      0.01
Processes:        114
Memory used:      11.0%
Swap used:        0.0%
Usage On:         11%
IP address:       192.168.0.75
Users online:     1
```

使用gs_install命令安装openGauss: **gs_install -X /opt/software/openGauss/clusterconfig.xml --gsinit-parameter="--encoding=UTF8" --dn-guc="max_process_memory=4GB" --dn-guc="shared_buffers= 256MB" --dn-guc="bulk_write_ring_size=256MB" --dn-guc="cstore_buffers=16MB"**

在执行过程中，用户需根据提示输入数据库管理员omm用户的密码。成功后如下图。

```
[omm@ecs-58ab ~]$ gs_install -X /opt/software/openGauss/clusterconfig.xml --gsinit-parameter="--encoding=UTF8"
--dn-guc="max_process_memory=4GB" --dn-guc="shared_buffers=256MB" --dn-guc="bulk_write_ring_size=256MB" --d
n-guc="cstore_buffers=16MB"
Parsing the configuration file.
Check preinstall on every node.
Successfully checked preinstall on every node.
Creating the backup directory.
Successfully created the backup directory.
begin deploy..
Installing the cluster.
begin prepare Install Cluster..
Checking the installation environment on all nodes.
begin install Cluster..
Installing applications on all nodes.
Successfully installed APP.
begin init Instance..
encrypt cipher and rand files for database.
Please enter password for database:
Please repeat for database:
begin to create CA cert files
The sslcert will be generated in /opt/gaussdb/app/share/sslcrt/om
Cluster installation is completed.
Configuring.
Deleting instances from all nodes.
Successfully deleted instances from all nodes.
Checking node configuration on all nodes.
Initializing instances on all nodes.
Updating instance configuration on all nodes.
Check consistence of memCheck and coresCheck on database nodes.
Configuring pg_hba on all nodes.
Configuration is completed.
Successfully started cluster.
Successfully installed application.
end deploy..
```

在数据库主节点服务器上，在omm操作系统用户环境下：
使用 **gs_om -t status** 命令查看服务是否启动。

```
[omm@ecs-58ab ~]$ gs_om -t status
```

```
-----  
cluster_name      : dbCluster  
cluster_state     : Normal  
redistributing    : No  
-----
```

如果没有启动，使用 **gs_om -t start** 命令启动数据库服务。

```
[omm@ecs-58ab ~]$ gs_om -t start
```

```
Starting cluster.
```

```
=====
```

```
[SUCCESS] ecs-58ab:
```

```
[2022-10-05 11:47:33.823][15563][][gs_ctl]: gs_ctl started,datadir is /gaussdb/data/d
```

```
[2022-10-05 11:47:33.826][15563][][gs_ctl]: another server might be running; Please
```

```
rt command
```

```
=====
```

```
Successfully started.
```

```
[omm@ecs-58ab ~]$ gsql -d postgres -p 26000 -r
```

```
gsql ((openGauss 2.0.0 build 78689da9) compiled at 2021-03-31 21:03:52 commit 0 last
```

```
Non-SSL connection (SSL connection is recommended when requiring high-security)
```

```
Type "help" for help.
```

```
postgres=#
```

使用 **gsql -d postgres -p 26000 -r** 命令连接数据库。

```
[omm@ecs-58ab ~]$ gsql -d postgres -p 26000 -r
gsql ((openGauss 2.0.0 build 78689da9) compiled at 2021-03-31 21:03:52 commit 0 last
Non-SSL connection (SSL connection is recommended when requiring high-security)
Type "help" for help.

postgres=#
```

出现上图即连接数据库成功。

命令中的postgres是openGauss安装完成后默认生成的数据库，初始可以连接到此数据库进行新数据库的创建。

26000为数据库主节点的端口号，需根据openGauss的实际情况做替换。

默认只有openGauss安装时创建的管理员用户（这里是omm）可以访问初始数据库。还可以使用 **CREATE USER username WITH PASSWORD "password"** ; 命令创建其他数据库用户帐号。下图成功创建了用户john。

```
postgres=# CREATE USER john WITH PASSWORD "REDACTED";  
CREATE ROLE
```

可以使用命令 **CREATE DATABASE database_name OWNER owner_name;** 创建数据库。下图成功创建了数据库newdb，管理员设为john。

```
postgres=# CREATE DATABASE newdb OWNER john;  
CREATE DATABASE
```



3

使用工具连接ECS

使用Data Studio连接ECS

打开ECS云服务器列表，点击“更多”“网络设置”“安全组规则配置”。
下图的云服务器的安全组是default，点击“配置规则”。

计费模式	标签	操作
按需计费	2023/03/29 19...	远程登录 更多 ¹
		<div>购买相同配置</div> <div>开机</div> <div>关机</div> <div>重启</div> <div>重置密码</div> <div>变更规格</div> <div>转包年/包月</div> <div>删除</div> <div>◀ 镜像/磁盘/备份</div> <div>◀ 网络设置 ²</div> <div>迁移云服务器</div>
		<div>更改安全组 3</div> <div>安全组规则配置</div> <div>修改私有IP</div> <div>解绑弹性公网IP</div> <div>修改带宽</div>

安全组规则配置

云服务器名称 ecs-7cf0

网卡 192.168.0.130 (主)

安全组 请输入安全组名称搜索 🔍 ↻

安全组名称	描述	操作
default	Default security group	配置规则

使用Data Studio连接ECS

点击“入方向规则”“添加规则”。

< | default

基本信息 **入方向规则** 出方向规则 关联实例

添加规则 快速添加规则 删除 一键放通 入方向规则: 8 教我设置

安全组规则对不同规格的云服务器生效情况不同, 如果您的安全组规则未生效, 请查看 [安全组规则限制](#)。

输入“优先级”为1, “协议端口”为26000, 作为后面连接数据库的端口。

安全组 default

如您要添加多条规则, 建议单击 [导入规则](#) 以进行批量导入。

优先级 ?	策略 ?	协议端口 ?	类型	源地址 ?	描述
1	允许	基本协议/自定义TCP 26000	IPv4	IP地址 0.0.0.0/0	连接数据库

初始情况下，在配置文件pg_hba.conf中规定了不允许远程登录。所以需要修改该文件允许远程登录。

使用find命令查找文件 **pg_hba.conf** 的位置。

```
[root@ecs-7cf0 ~]# find / -name pg_hba.conf  
/gaussdb/data/db1/pg_hba.conf
```

在pg_hba.conf的合适位置添加以下两行。

host all all 0.0.0.0/0 md5

```
86 # TYPE DATABASE USER ADDRESS METHOD  
87  
88 # "local" is for Unix domain socket connections only  
89 local all all trust  
90 # IPv4 local connections:  
91 host all all 127.0.0.1/32 trust  
92 host all all 192.168.0.130/32 trust  
93 host all all 0.0.0.0/0 md5  
94 # IPv6 local connections:  
95 host all all ::1/128 trust
```

这行命令的意思是：对于主机配置（host），允许所有远程主机（0/0），以所有用户身份（all），以密码md5加密的方式（md5），登录所有的库（all）。

修改同文件夹下的postgresql.conf文件。将listen_addresses的值改为*。

```
59 # listen_addresses = '192.168.0.130'      # what IP address(es) to listen on;  
60 #                                     # comma-separated list of addresses;  
61 #                                     # defaults to 'localhost'; use '*' for all  
62 #                                     # (change requires restart)  
63 listen_addresses = '*'
```

并删掉password_encryption_type前面的#，且赋值0，表示使用md5加密密码。

```
99 #failed login attempts = 10      #Enter the wrong password reached failed_login_attempts times, the current account  
100 password_encryption_type = 0      #Password storage type, 0 is md5 for PG, 1 is sha256 + md5, 2 is sha256 only  
101 #password_min_length = 6         #The minimal password length(6-999)  
102 #password_max_length = 32        #The maximal password length(6-999)
```

使用gs_om -t restart命令重启数据库。

```
[omm@ecs-7cf0 ~]$ gs_om -t restart  
Stopping cluster.  
=====
```

Successfully stopped cluster.

```
=====
```

End stop cluster.
Starting cluster.
=====

[SUCCESS] ecs-7cf0
2023-03-30 11:55:56.081 6425084c.1 [unknown] 281461964865552 [unknown] 0 dn_6001 01000 0 [BACKEND] WARNING: could not create any HA TCP/IP sockets
2023-03-30 11:55:56.087 6425084c.1 [unknown] 281461964865552 [unknown] 0 dn_6001 01000 0 [BACKEND] WARNING: No explicit IP is configured for listen_addresses GUC.
=====

Successfully started.

修改用户密码，使之用默认的加密方式重新加密。

这里使用命令 **alter user john identified by 'new_password'**; 修改之前创建的用户john的密码，注意新旧密码不能相同，否则不能修改成功，也不能改为之前用过的密码。

```
postgres=# alter user john identified by 'ABC123...';  
NOTICE:  The encrypted password contains MD5 ciphertext, which is not secure.  
ALTER ROLE  
postgres=#
```

使用命令 **alter role john with login;** 设置john可以登录。

```
postgres=# alter role "john" with login;  
ALTER ROLE
```

使用Data Studio连接ECS



在本地电脑打开Data Studio，点击“文件” “新建连接”。

比如连接数据库newdb，如右图：

其中

名称：随意输入

主机：ECS的公网IP地址

端口号：26000

数据库：newdb

用户名：john

点击确定建立数据库连接。

通用 SSL 高级

数据库类型 openGauss

名称* connection1

主机* 119.3.187.82

端口号* 26000 最大值 65535

数据库* newdb

用户名* john

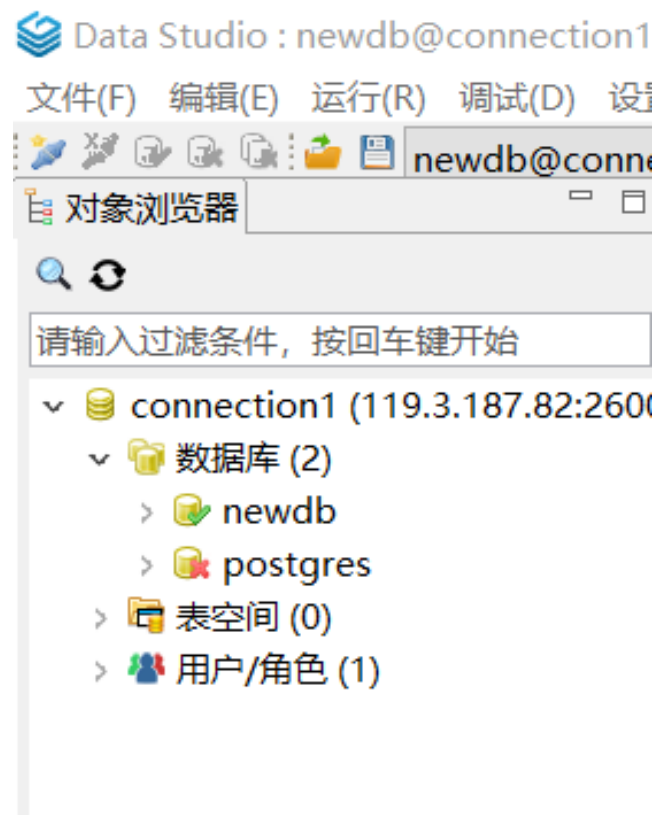
密码*

保存密码 仅当前会话

启用SSL ☐

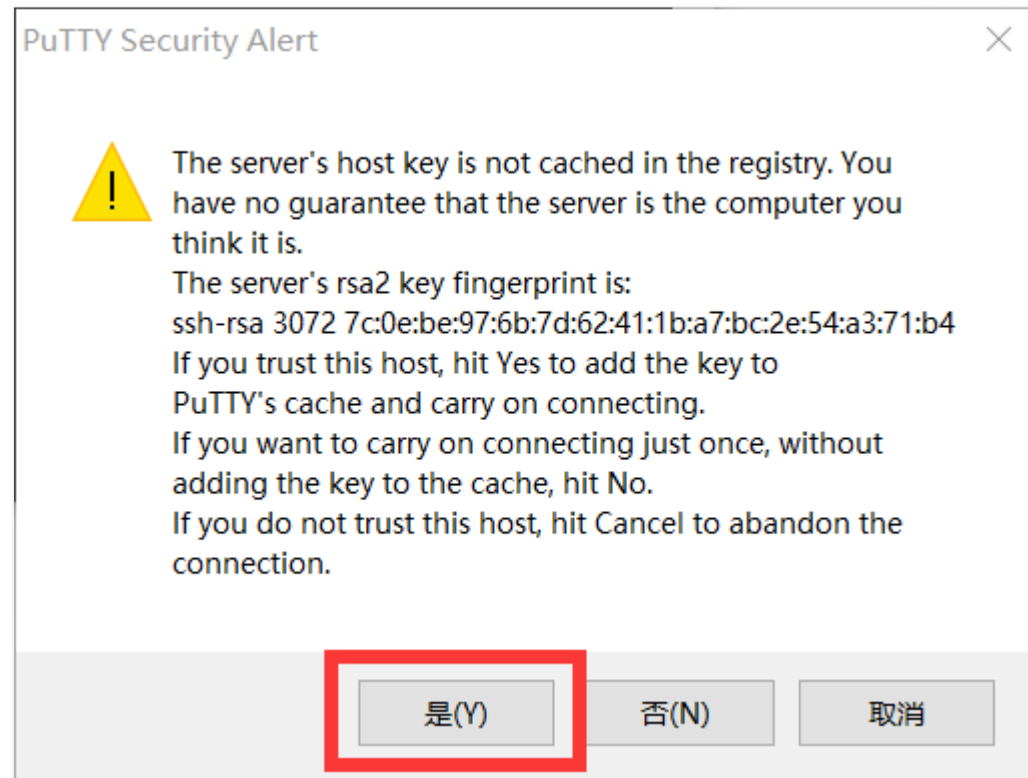
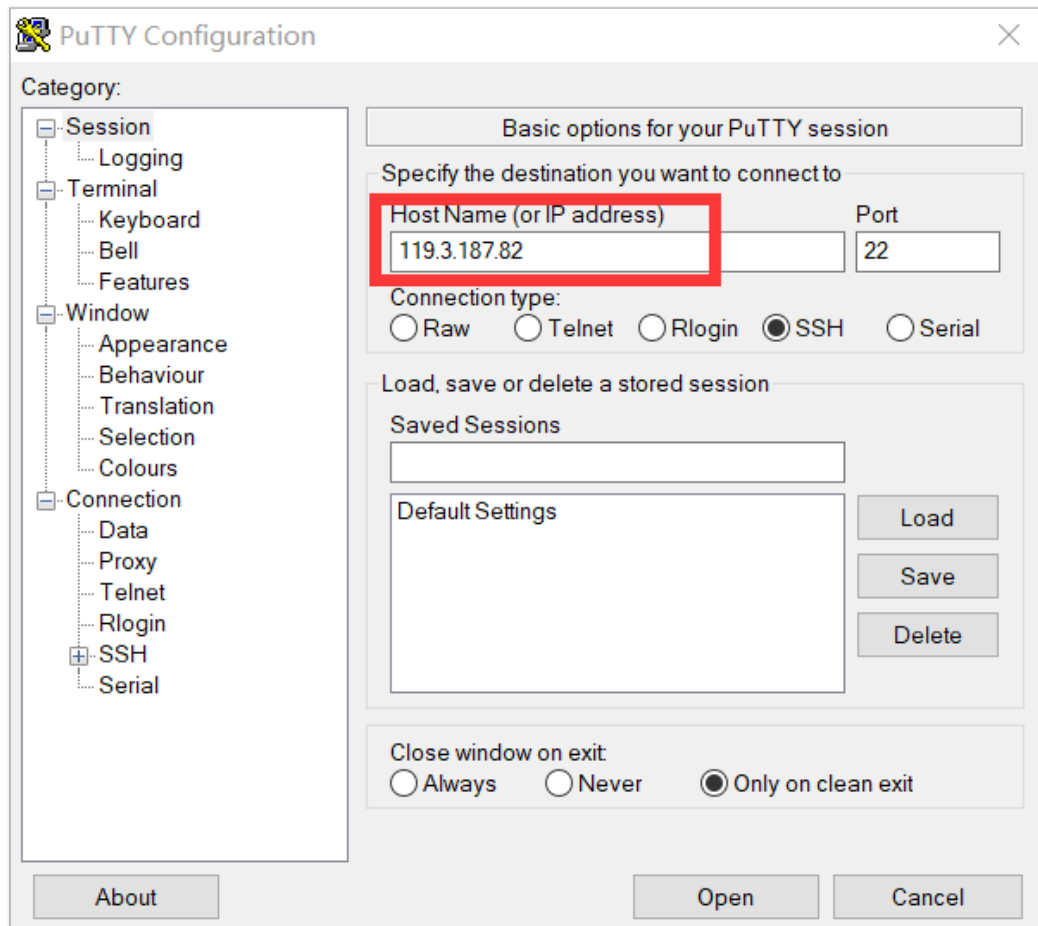
确定 清除 关闭

弹窗提示表示连接成功。



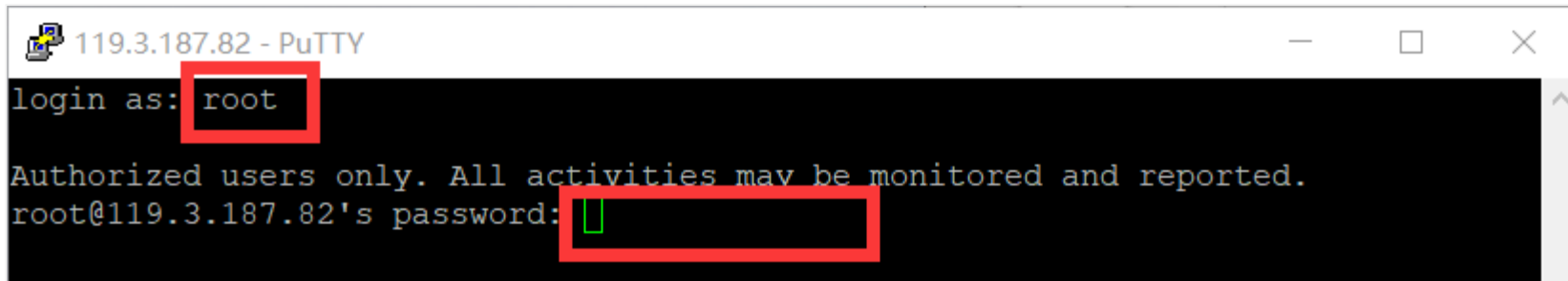
使用putty连接ECS

在本地电脑打开putty，在Host Name的地方输入购买的ECS服务器的公网IP地址，点击“Open”。如果弹出右图的窗口，点击“是”。



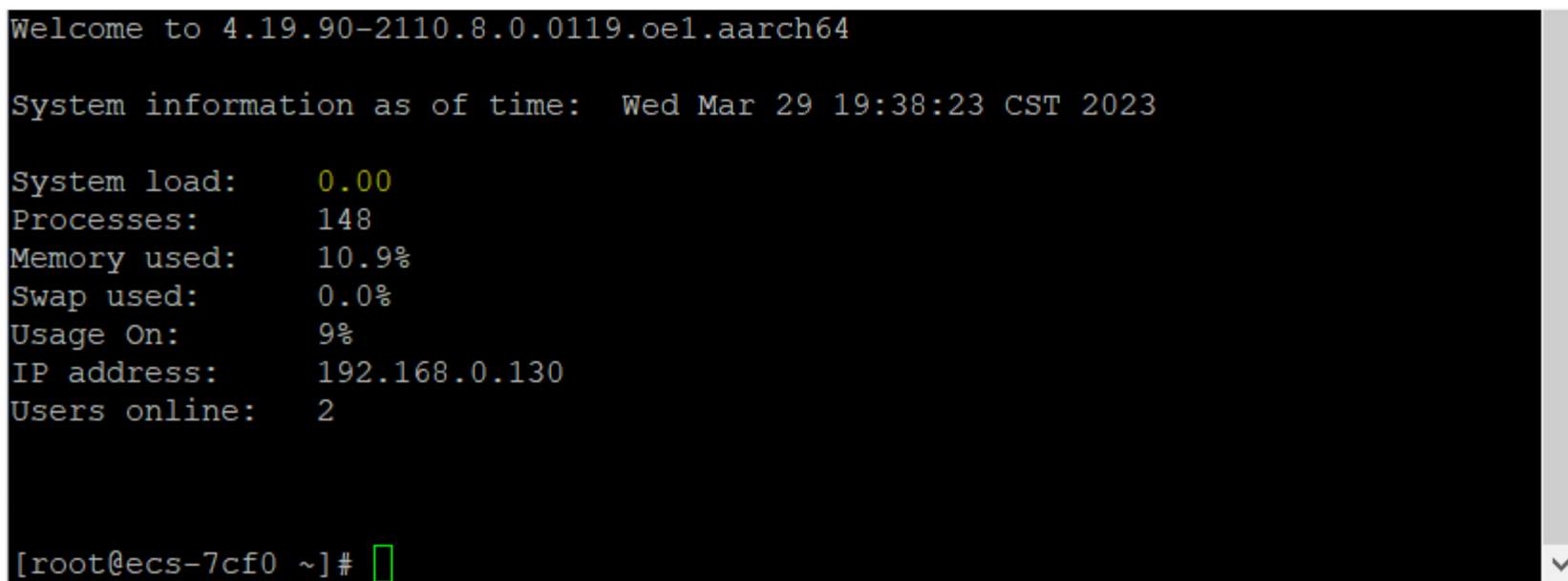
使用putty连接ECS

以root用户来登录。输入密码。



A screenshot of a PuTTY terminal window titled "119.3.187.82 - PuTTY". The terminal shows the login process: "login as: root" (with "root" highlighted by a red box), followed by "Authorized users only. All activities may be monitored and reported." and "root@119.3.187.82's password:" (with the password input field highlighted by a red box).

提示如下，即连接成功。



A screenshot of a PuTTY terminal window showing system information after a successful login. The text displayed is: "Welcome to 4.19.90-2110.8.0.0119.oel.aarch64", "System information as of time: Wed Mar 29 19:38:23 CST 2023", and a list of system statistics: "System load: 0.00", "Processes: 148", "Memory used: 10.9%", "Swap used: 0.0%", "Usage On: 9%", "IP address: 192.168.0.130", and "Users online: 2". The prompt "[root@ecs-7cf0 ~]#" is visible at the bottom.



4 使用简单的SQL语句

创建表空间fastspace。

CREATE TABLESPACE fastspace RELATIVE LOCATION 'tablespace/tablespace_1';

```
postgres=# CREATE TABLESPACE fastspace RELATIVE LOCATION 'tablespace/tablespace_1';  
CREATE TABLESPACE
```

将“fastspace”表空间的访问权限赋予数据用户john。

GRANT CREATE ON TABLESPACE fastspace TO john;

```
postgres=# GRANT CREATE ON TABLESPACE fastspace TO john;  
GRANT
```

创建表。

CREATE TABLE customer_t1

(
 c_customer_sk **integer,**
 c_customer_id **char(5),**
 c_first_name **char(6),**
 c_last_name **char(8)**
);

```
postgres=# CREATE TABLE customer_t1
postgres=# (
postgres=#     c_customer_sk          integer,
postgres=#     c_customer_id         char(5),
postgres=#     c_first_name          char(6),
postgres=#     c_last_name           char(8)
postgres=# );
CREATE TABLE
postgres=#
```

向表中插入单行数据。

**INSERT INTO customer_t1(c_customer_sk, c_customer_id, c_first_name)
VALUES (3769, 'hello', 'Grace');**

```
postgres=# INSERT INTO customer_t1(c_customer_sk, c_customer_id, c_first_name) VALUES (3769, 'hello', 'Grace');
INSERT 0 1
postgres=#
```

向表中插入多行数据。

```
postgres=# INSERT INTO customer_t1 (c_customer_sk, c_customer_id, c_first_name) VALUES
postgres=#     (6885, 'maps', 'Joes'),
postgres=#     (4321, 'tpcds', 'Lily'),
postgres=#     (9527, 'world', 'James');
INSERT 0 3
postgres=#
```

更新表中数据。

UPDATE customer_t1 SET c_customer_sk = 9876 WHERE c_customer_sk = 9527;

```
postgres=# UPDATE customer_t1 SET c_customer_sk = 9876 WHERE c_customer_sk = 9527;
UPDATE 1
postgres=#
```

使用系统表pg_tables查询数据库所有表的信息。

SELECT * FROM pg_tables;

```
postgres=# SELECT * FROM pg_tables;
 schemaname |      tablename      | tableowner | tablespace | hasindexes | hasrules | hastriggers | tablecreator
-----+-----+-----+-----+-----+-----+-----+-----
 pg_catalog | pg_statistic         | omm        |             | t          | f        | f          | 
 public    | customer_t1          | omm        |             | f          | f        | f          | omm
5+08
 pg_catalog | pg_type              | omm        |             | t          | f        | f          | 
 pg_catalog | pg_ts_dict           | omm        |             | t          | f        | f          | 
 pg_catalog | pg_job_proc          | omm        | pg_global  | t          | f        | f          | 
 pg_catalog | pg_authid            | omm        | pg_global  | t          | f        | f          | 
 pg_catalog | pg_statistic_ext     | omm        |             | t          | f        | f          | 
 pg_catalog | gs_wlm_instance_history | omm        |             | f          | f        | f          | 
 pg_catalog | gs_wlm_session_query_info_all | omm        |             | f          | f        | f          | 
 pg_catalog | gs_wlm_user_resource_history | omm        |             | f          | f        | f          | 
 pg_catalog | pg_user_mapping      | omm        |             | t          | f        | f          |
```

创建schema。

CREATE SCHEMA myschema;

```
postgres=# CREATE SCHEMA myschema;  
CREATE SCHEMA
```

在myschema下创建mytable表。

CREATE TABLE myschema.mytable(id int, name varchar(20));

```
postgres=# CREATE TABLE myschema.mytable(id int, name varchar(20));  
CREATE TABLE
```

查询myschema下mytable表的所有数据。

SELECT * FROM myschema.mytable;

```
postgres=# SELECT * FROM myschema.mytable;  
 id | name  
----+-----  
(0 rows)
```

创建普通视图。

```
CREATE OR REPLACE VIEW MyView AS SELECT * FROM customer_t1 WHERE  
c_first_name like 'G%';
```

```
postgres=# CREATE OR REPLACE VIEW MyView AS SELECT * FROM customer_t1 WHERE c_first_name like 'G%';  
CREATE VIEW
```

查询普通视图。

```
SELECT * FROM MyView;
```

```
postgres=# SELECT * FROM MyView;  
 c_customer_sk | c_customer_id | c_first_name | c_last_name  
-----+-----+-----+-----  
          3769 | hello        | Grace       |  
(1 row)
```

创建索引。

```
CREATE INDEX cor_all_std_key ON score(cor_id);
```

```
scg=> CREATE INDEX std_all_cor_key ON score(std_id);  
CREATE INDEX  
scg=> CREATE INDEX cor_all_std_key ON score(cor_id);  
CREATE INDEX
```

创建触发器函数。

```
CREATE OR REPLACE FUNCTION func_name() RETURNS TRIGGER AS
$$
DECLARE
    **函数体**
RETURN NEW/OLD;
END
$$ LANGUAGE PLPGSQL;
```

示例:

```
scg=> CREATE OR REPLACE FUNCTION insert_fail_t_func() RETURNS TRIGGER AS
scg-> $$
scg$> DECLARE
scg$> BEGIN
scg$> IF NEW.grade<60 THEN
scg$> INSERT INTO fail VALUES(NEW.std_id,NEW.cor_id,NEW.grade);
scg$> END IF;
scg$> RETURN NEW;
scg$> END
scg$> $$LANGUAGE PLPGSQL;
CREATE FUNCTION
```


创建INSERT触发器。

```
CREATE TRIGGER trigger_name  
AFTER INSERT ON table_name  
FOR EACH ROW  
EXECUTE PROCEDURE func_name();
```

创建UPDATE触发器。

```
CREATE TRIGGER trigger_name  
AFTER UPDATE ON table_name  
FOR EACH ROW  
EXECUTE PROCEDURE func_name();
```

创建DELETE触发器。

```
CREATE TRIGGER trigger_name  
BEFORE DELETE ON table_name  
FOR EACH ROW  
EXECUTE PROCEDURE func_name();
```

示例：

```
scg=> CREATE TRIGGER insert_fail_t  
scg-> AFTER INSERT ON score  
scg-> FOR EACH ROW  
scg-> EXECUTE PROCEDURE insert_fail_t_func();  
CREATE TRIGGER
```

创建存储过程。

```
create or replace procedure  
procedure_name(var_name var_type)  
as  
declare  
    **核心部分要用的变量的声明**  
    var_name1 var_type1 := 0;  
begin  
    **核心部分**  
end;  
/
```

调用存储过程。

```
call procedure_name(参数值) ;
```

示例：

```
postgres=# create or replace procedure cal_gpa(cur_std_id int)  
postgres=# as  
postgres## declare  
postgres## credit_sum numeric := 0;  
postgres## gpa_sum numeric := 0;  
postgres## begin  
postgres## if (select count(1) from gpa where std_id = cur_std_id) = 0 then  
postgres## insert into gpa values(cur_std_id, gpa_sum/credit_sum);  
postgres## else  
postgres## update gpa set std_gpa = gpa_sum/credit_sum where std_id = cur_std_id;  
postgres## end if;  
postgres## end;  
postgres## /  
CREATE PROCEDURE
```



北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

Thank you!
