结算图标.JPG中国证券登记结算有限责任公司总部项目文档

上线操作手册

项目名称: **内部评级体系建设**

编制单位: **中证信用增进股份有限公司**

编制部门: **研发部**

编制人 : **王振，尚强强，李睿**

日 期 : **2018年6月21日**

目录

[1. 说明 4](#_Toc490835711)

[1.1 部署顺序 4](#_Toc490835712)

[1.2 数据初始化 4](#_Toc490835713)

[2. 数据库部署 4](#_Toc490835714)

[2.1 操作系统内核参数、资源限制配置 4](#_Toc490835715)

[2.2 编译安装PostgreSQL 9.4.10 5](#_Toc490835716)

[2.3 创建数据库管理用户 6](#_Toc490835717)

[2.4 导入数据库文件 6](#_Toc490835718)

[2.5 修改数据库管理用户的环境变量 6](#_Toc490835719)

[2.6 修改数据库参数 7](#_Toc490835720)

[2.7 启动数据库 7](#_Toc490835721)

[3. 内评系统数据中转服务器上创建FTP用户 7](#_Toc490835722)

[4. 结算外网服务器上部署数据拉取程序 8](#_Toc490835723)

[4.1 安装JDK 8](#_Toc490835724)

[4.2 生成SSH密钥 9](#_Toc490835725)

[4.3 部署中证数据文件拉取程序 10](#_Toc490835726)

[5. 数据同步服务器上部署数据同步程序 11](#_Toc490835727)

[5.1 安装JDK 11](#_Toc490835728)

[5.2 创建一个用户用于接收数据仓库导出的可回购债券清单 11](#_Toc490835729)

[5.3 创建一个用户用于部署同步程序 11](#_Toc490835730)

[5.4 安装PostgreSql客户端 11](#_Toc490835731)

[5.5 部署数据同步程序 12](#_Toc490835732)

[5.6 调度配置 14](#_Toc490835733)

[6. 应用程序部署 14](#_Toc490835734)

[6.1 环境准备 14](#_Toc490835735)

[6.1.1 安装python-devel 14](#_Toc490835736)

[6.1.2 安装c++编译器 15](#_Toc490835737)

[6.1.3 安装zlib 15](#_Toc490835738)

[6.1.4 安装libxml2 15](#_Toc490835739)

[6.1.5 安装 pcre 16](#_Toc490835740)

[6.1.6 安装nginx 16](#_Toc490835741)

[6.1.7 安装JDK 8 17](#_Toc490835742)

[6.2 安装和启动redis 17](#_Toc490835743)

[6.3 配置nginx 17](#_Toc490835744)

[6.4 后台部署 18](#_Toc490835745)

[6.5 前台部署 18](#_Toc490835746)

[6.6 清除缓存 19](#_Toc490835747)

[6.7 常见错误日志 19](#_Toc490835748)

[7. 备份数据库上部署数据库备份脚本 19](#_Toc490835749)

[7.1 安装JDK 19](#_Toc490835750)

[7.2 安装PostgreSQL客户端 20](#_Toc490835751)

[7.3 部署数据库备份脚本 21](#_Toc490835752)

[7.4 修改数据库服务器的访问权限 21](#_Toc490835753)

[7.5 调度配置 21](#_Toc490835754)

# 说明

## 部署顺序

请按照以下顺序进行部署：

1. 数据库
2. 数据拉取程序
3. 数据同步程序
4. 应用程序
5. 数据库备份脚本

## 数据初始化

数据库在部署之前可根据客户的需要，对数据做一定的初始化操作，以适应客户的数据定制化需求。

初始化内容包括角色、用户、权限、敞口负责人、组合和预设规则，以及评级相关的指标、字段、模型和主标尺。

初始化的数据将以数据包的形式在部署时进行导入。

# 数据库部署

## 数据库初始化

首先需要将生产环境数据库复制到UAT环境数据库，可以参考以下步骤



## 数据库增量部署

### 软件清单

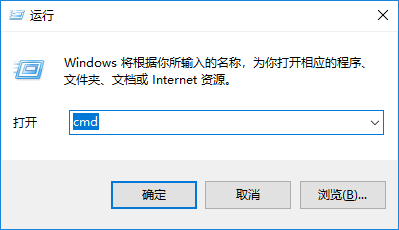
* PostgreSql客户端

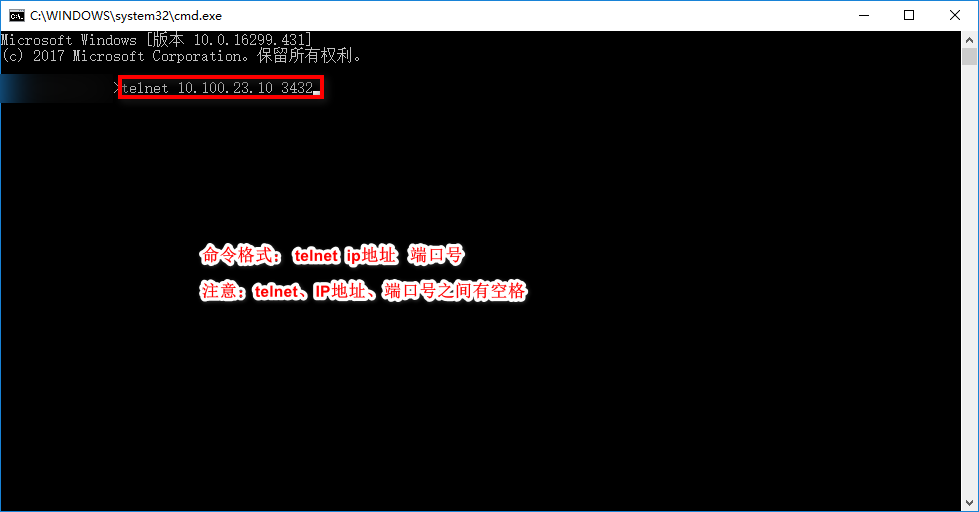
如果客户提供pgadmin软件，优先使用客户提供的软件，如不提供可使用中证方提供的pgadmin软件，也可在线进行下载。

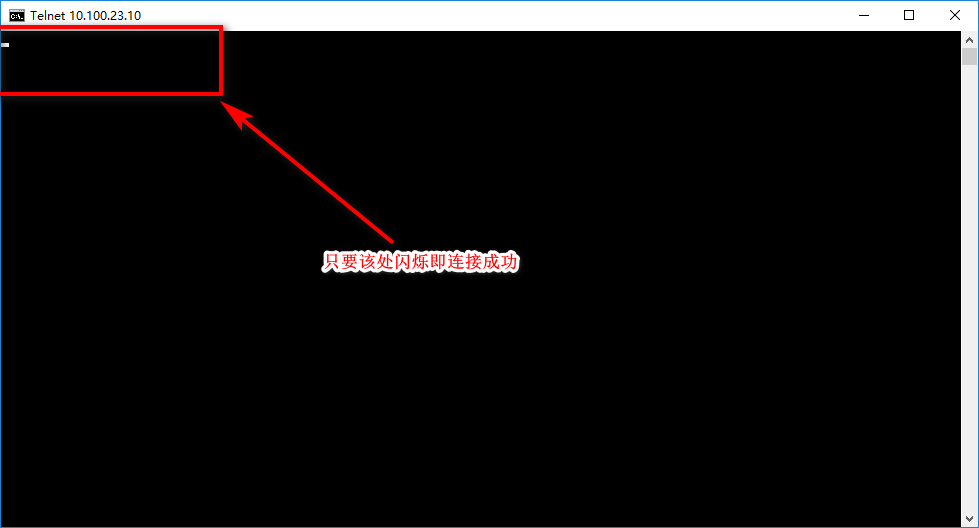
### 部署权限申请及其他事项

执行部署脚本的机器对部署的数据库服务器有访问权限，如无对应的权限则需要申请权限

#### 检查网络是否通畅





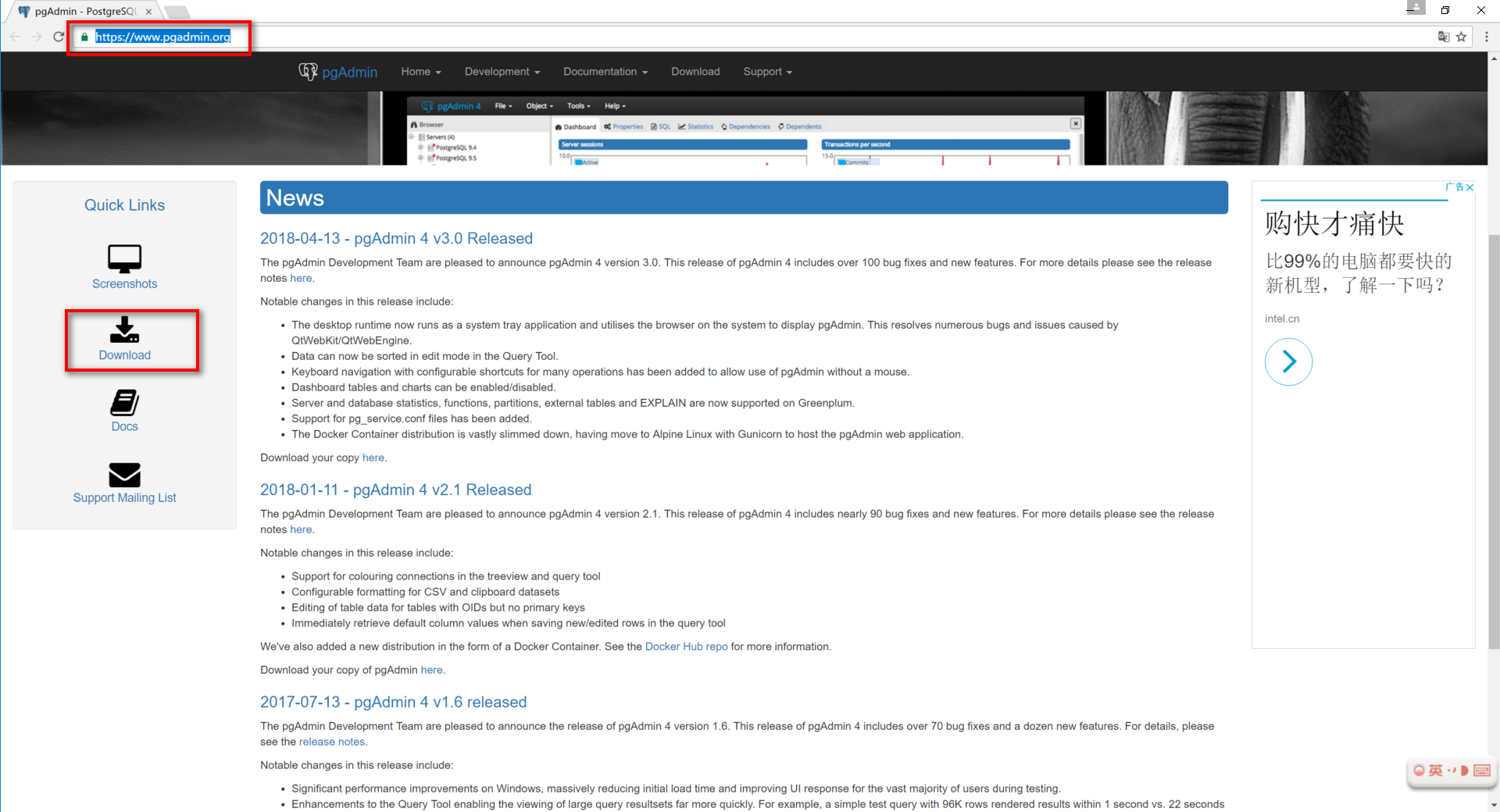


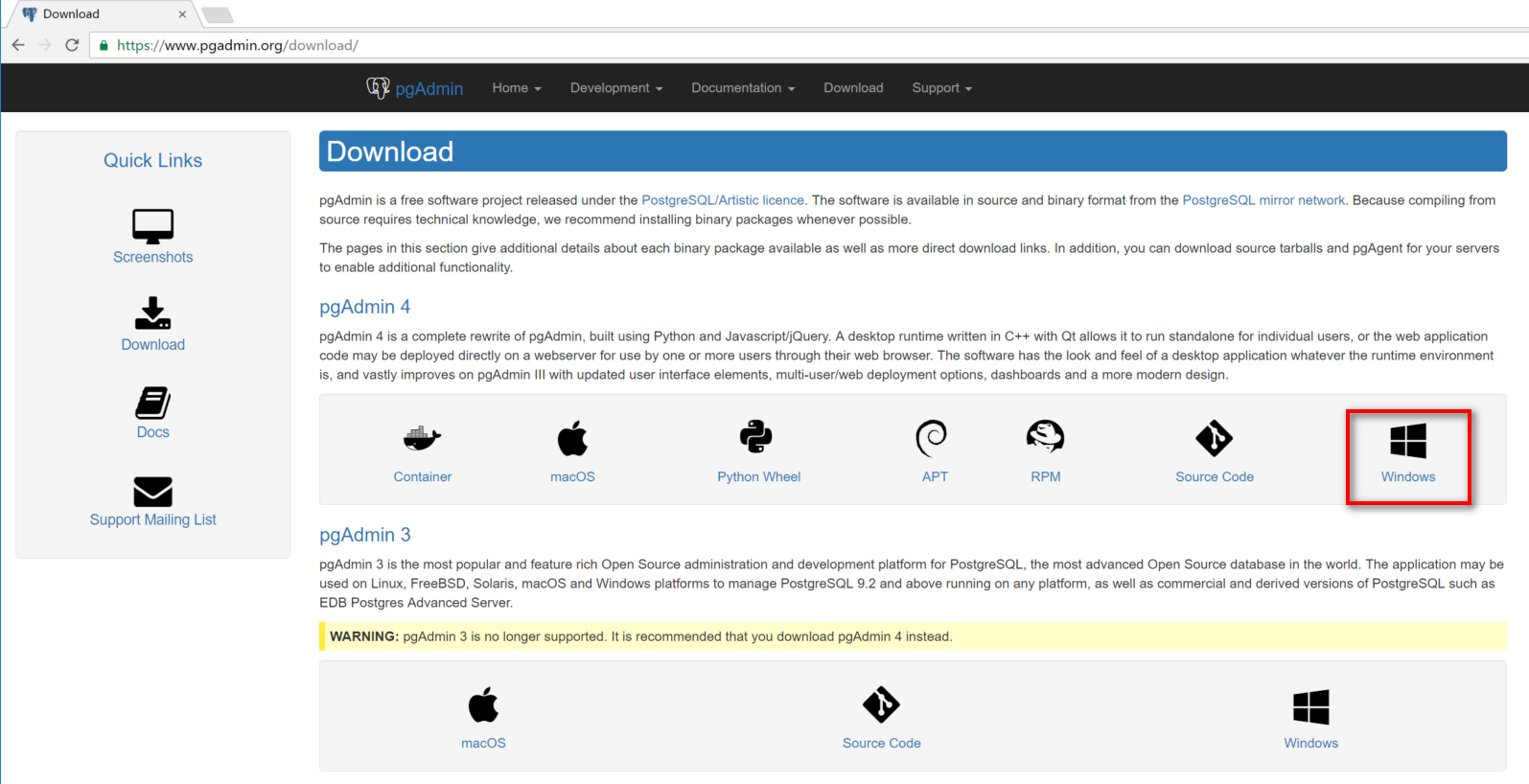
#### 检查应用是否停止

在数据库增量部署时，需要关闭正在连接数据库的应用，避免继续产生业务数据以及避免在升级时造成的不必要的异常。

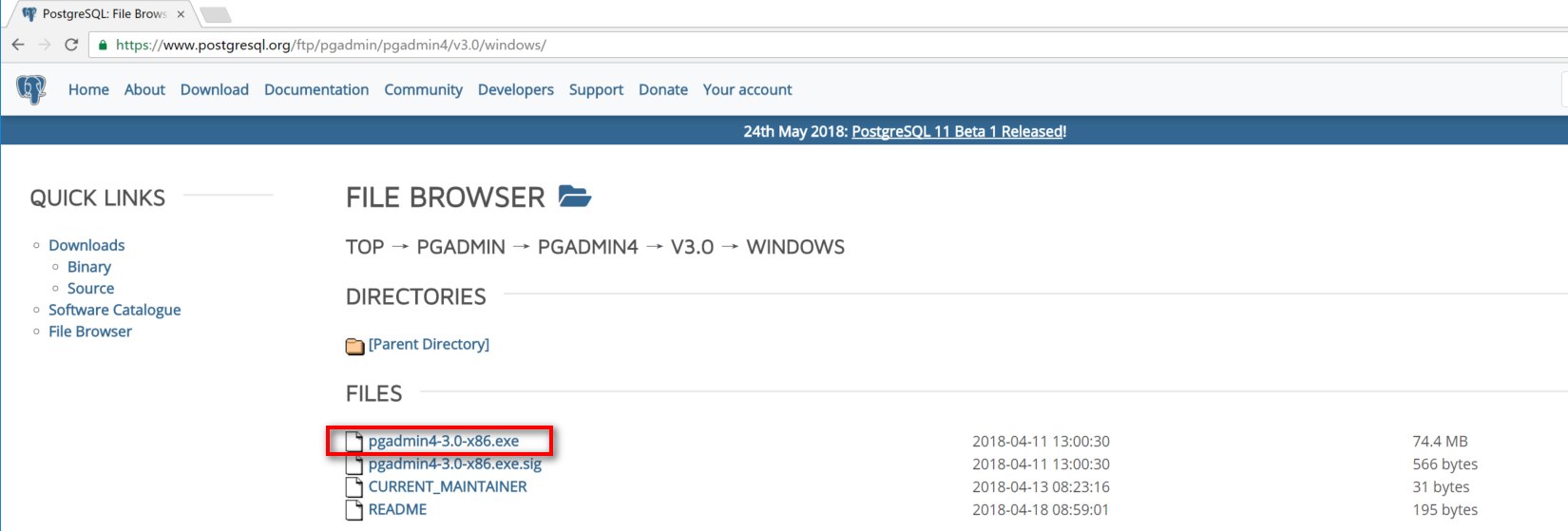
### 软件在线下载

网址： <https://www.pgadmin.org/>

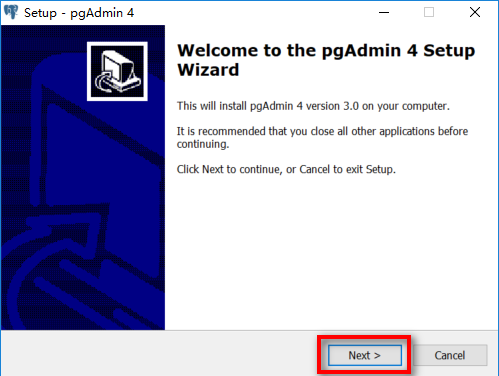


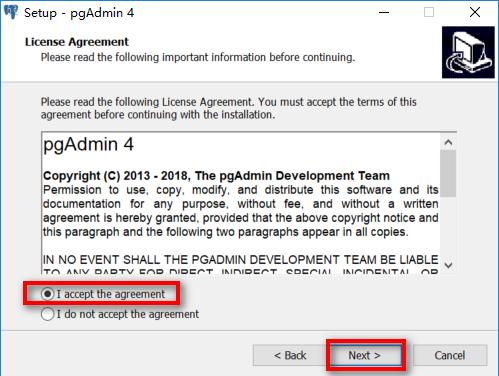


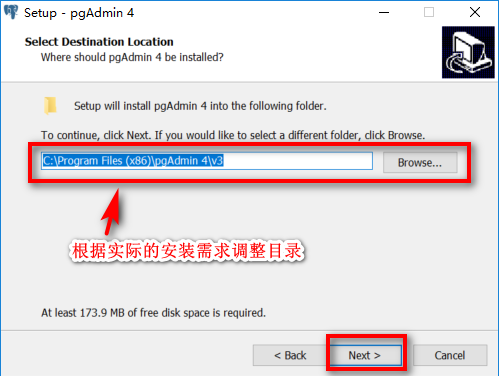


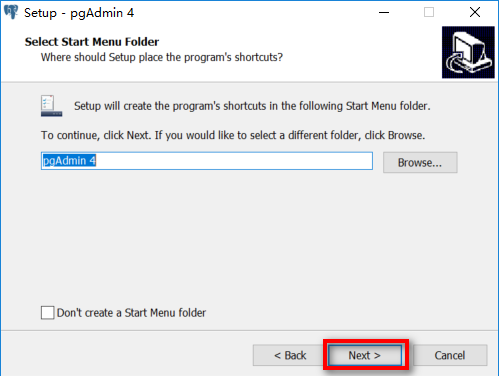


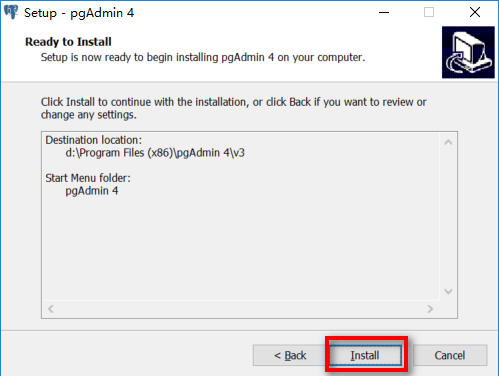
### Pgadmin安装步骤

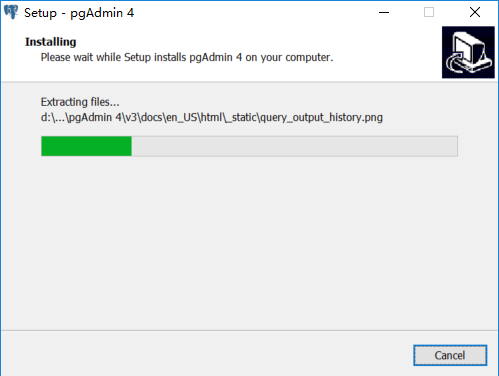




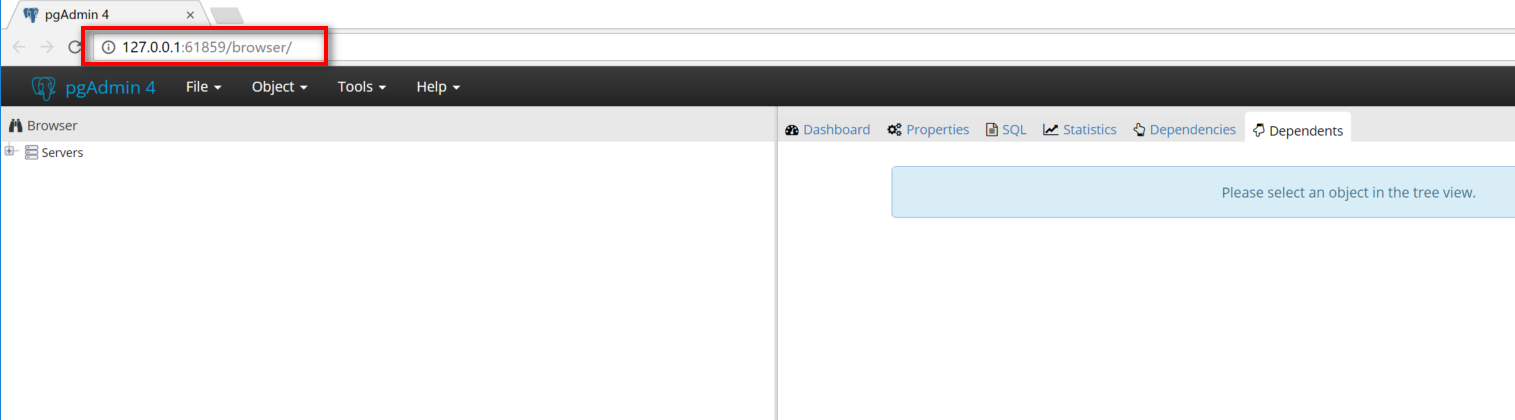






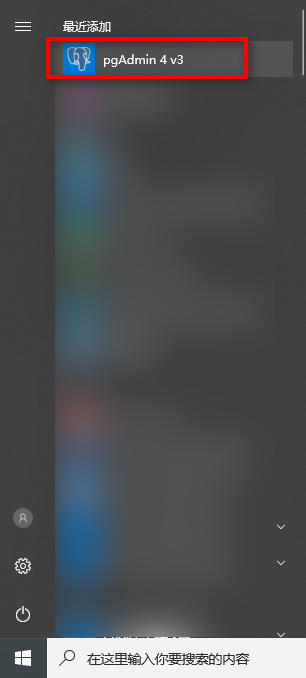


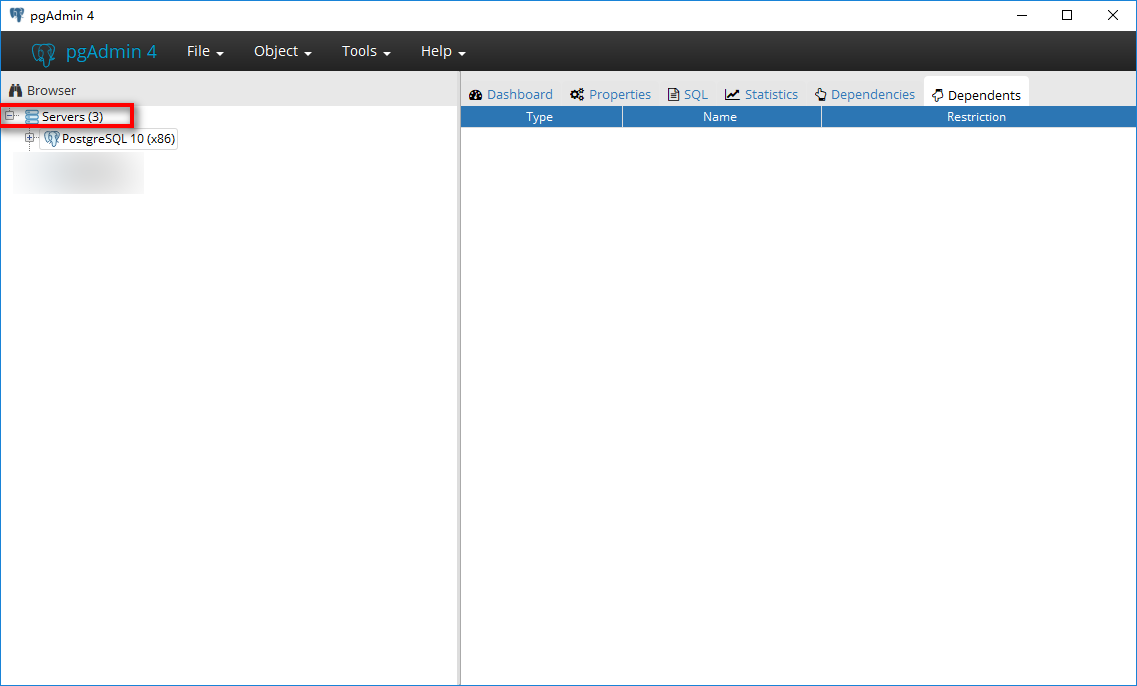


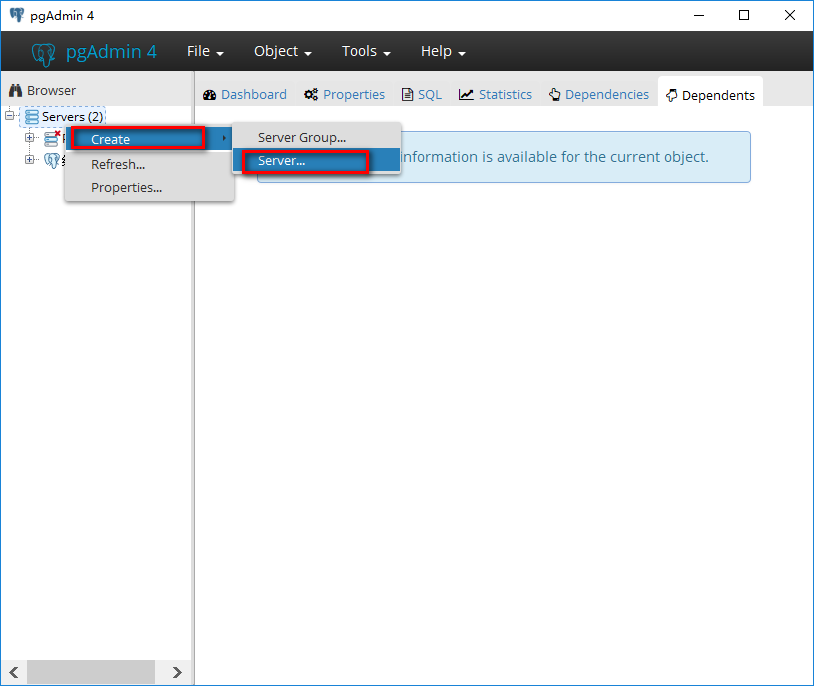


**登录部署脚本的客户端及其执行操作示例**中使用的软件与本次安装的软件实际操作一致。不再基于安装的软件版本不同而调整。

### 登录客户端并配置数据库连接信息

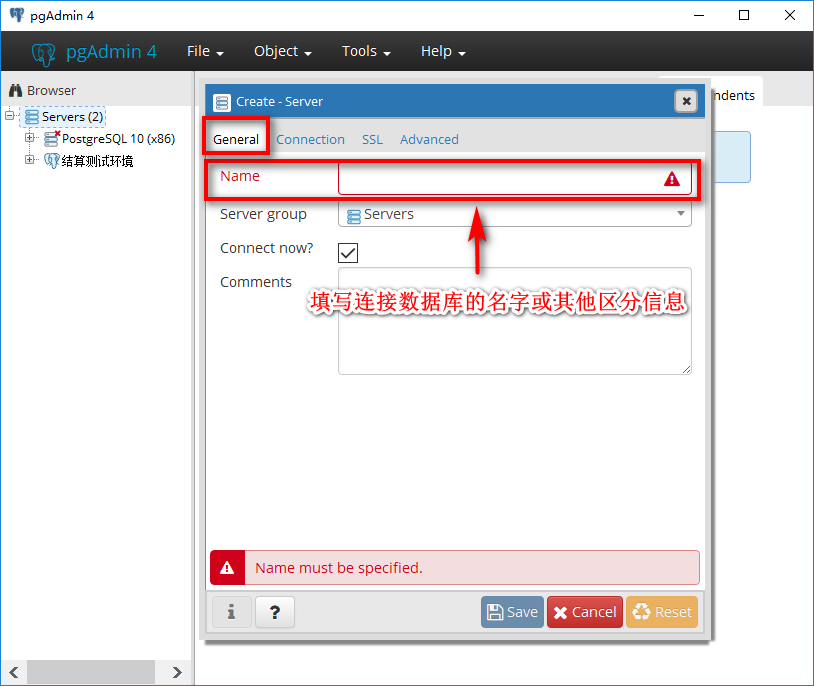






连接数据库名称配置信息

① 填写连接数据库的名字或其他区分信息，比如UAT-stg，UAT-tgt，可以区分出连接的环境以及数据库。



数据库配置界面

① Host name/address: 增量部署脚本对应的数据库服务器IP地址

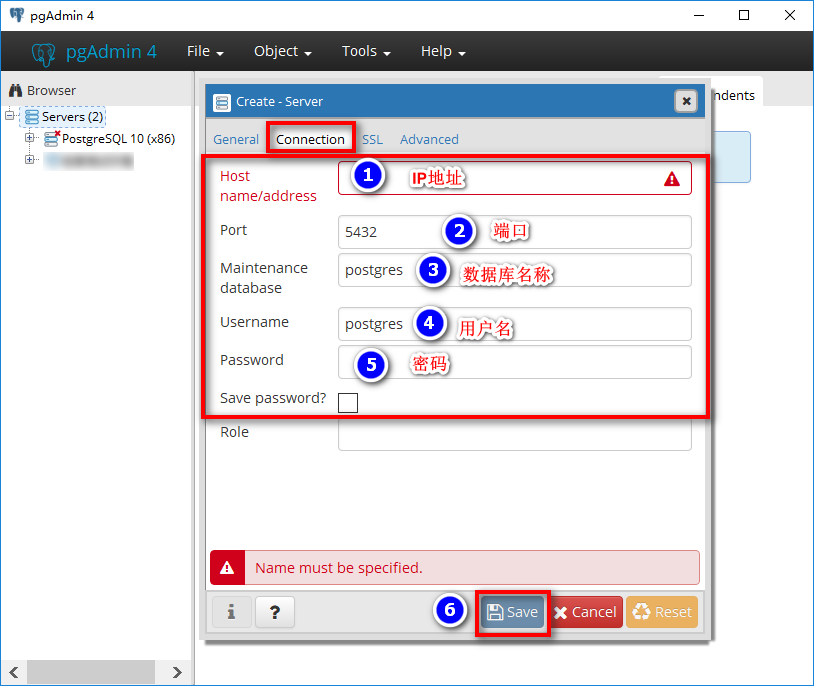
② Port： 增量部署脚本对应的数据库服务器端口号，默认为5432，根据实际情况调整

③ Maintenance database：部署脚本对应的数据库名称，如cs\_master\_stg,cs\_master\_tgt

④ Username：登录对应数据库的用户名

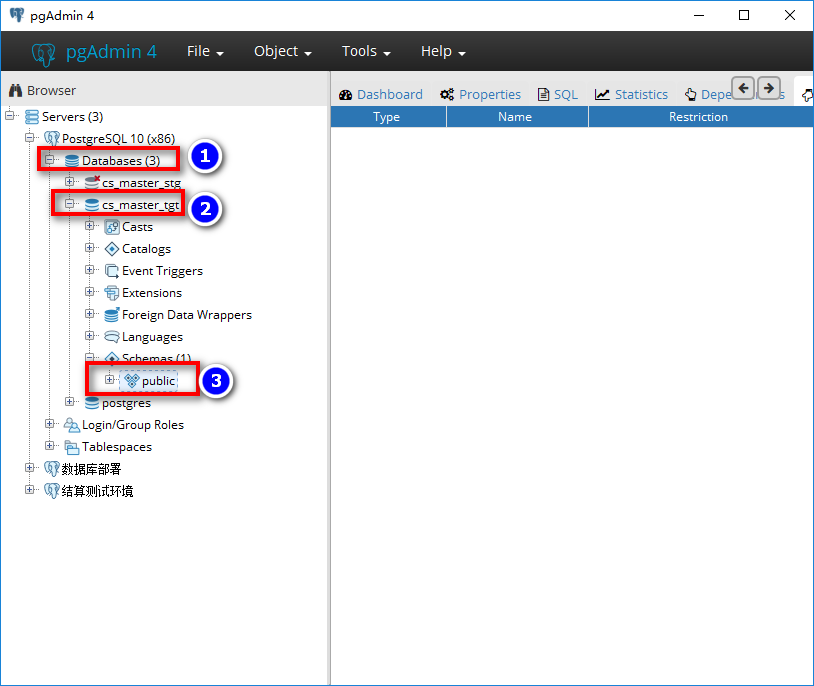
⑤ Password：登录对应数据库的密码

建议：勾选不勾选Save password，以避免跳过密码认证误连接数据库的情况。

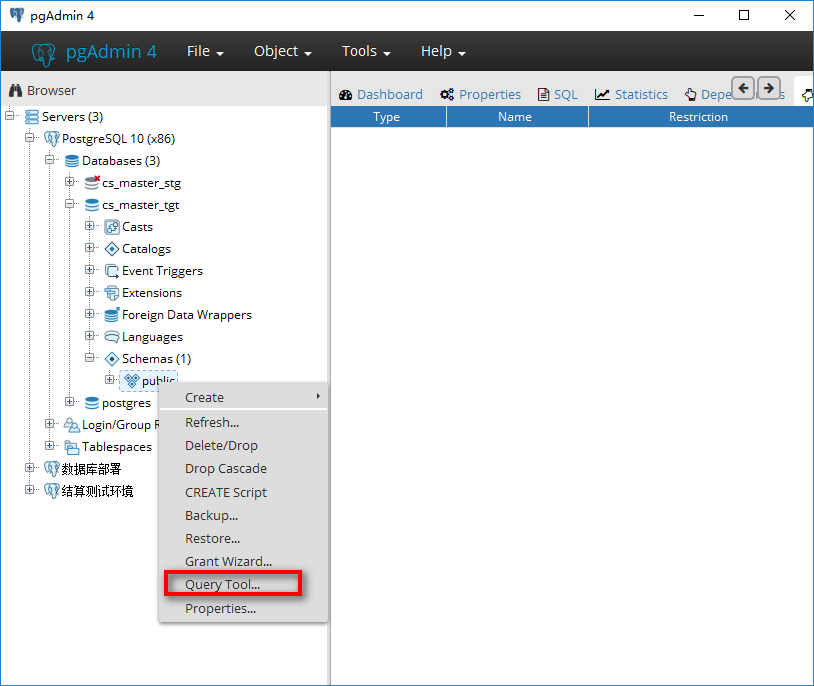


### 客户端执行脚本示例

选择配置的连接对应数据库的数据库下的schema中的public



点击选择QueryTool，打开执行部署脚本的界面



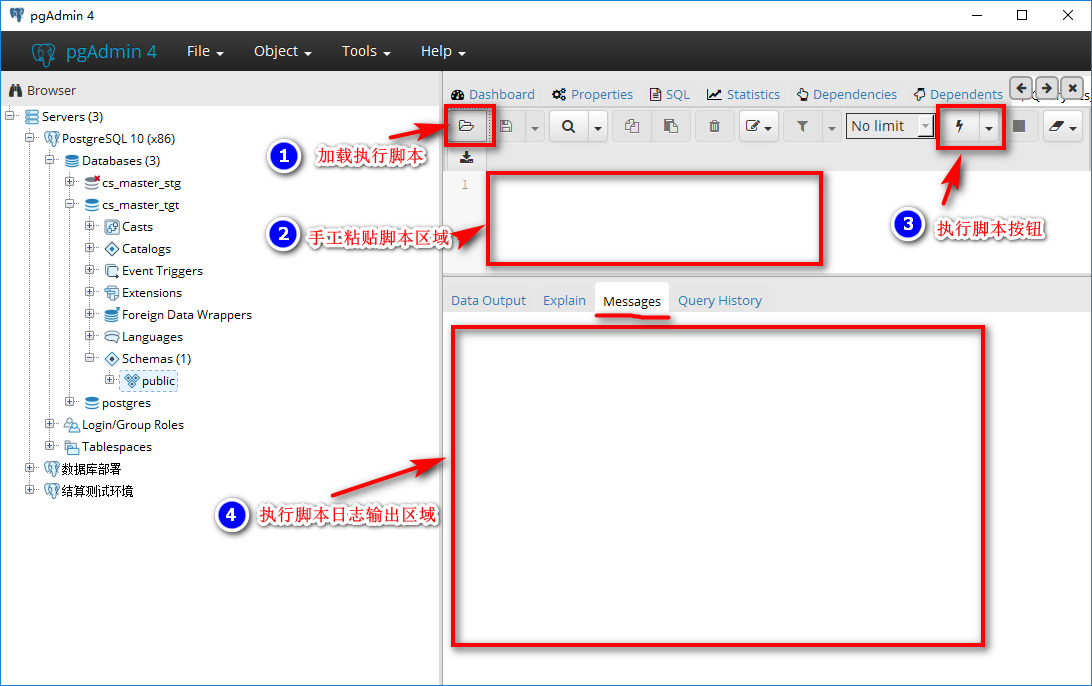
执行脚本界面简单介绍

① 选择数据库增量部署脚本目录下的上传脚本按钮

② 选择从数据库增量部署脚本文件拷贝执行命令到执行区域

③ 执行命令按钮

④ 基于执行命令的实际输出日志区域

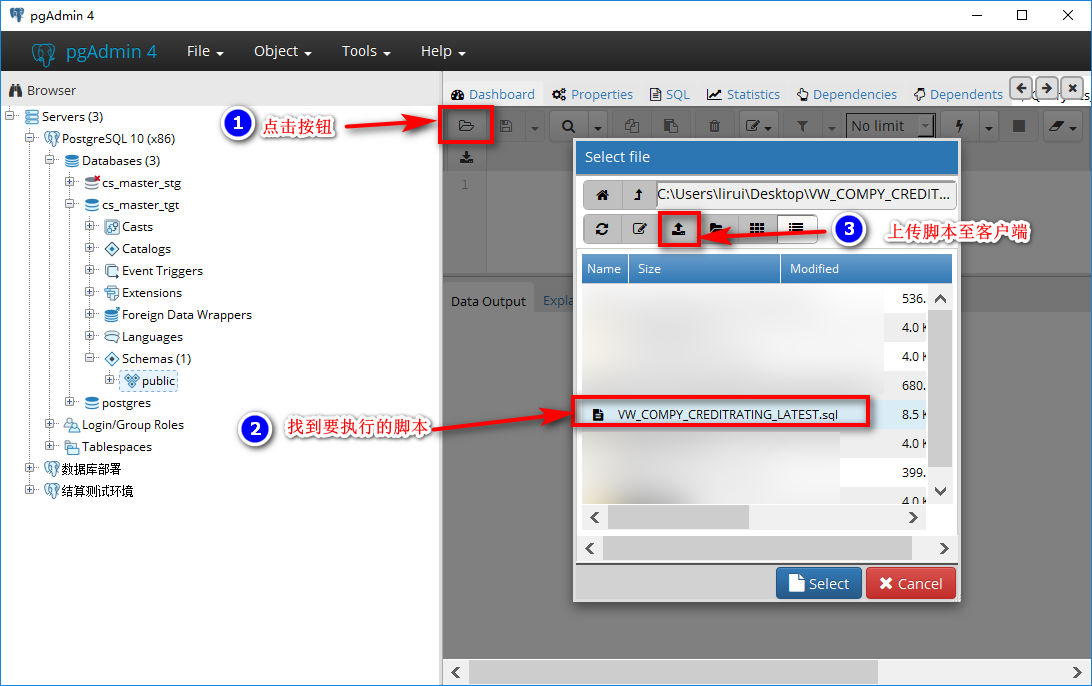


从数据库增量部署脚本上传脚本文件到执行区域简单介绍

① 选择数据库增量部署脚本目录下的上传脚本按钮

② 点击选择要上传的脚本

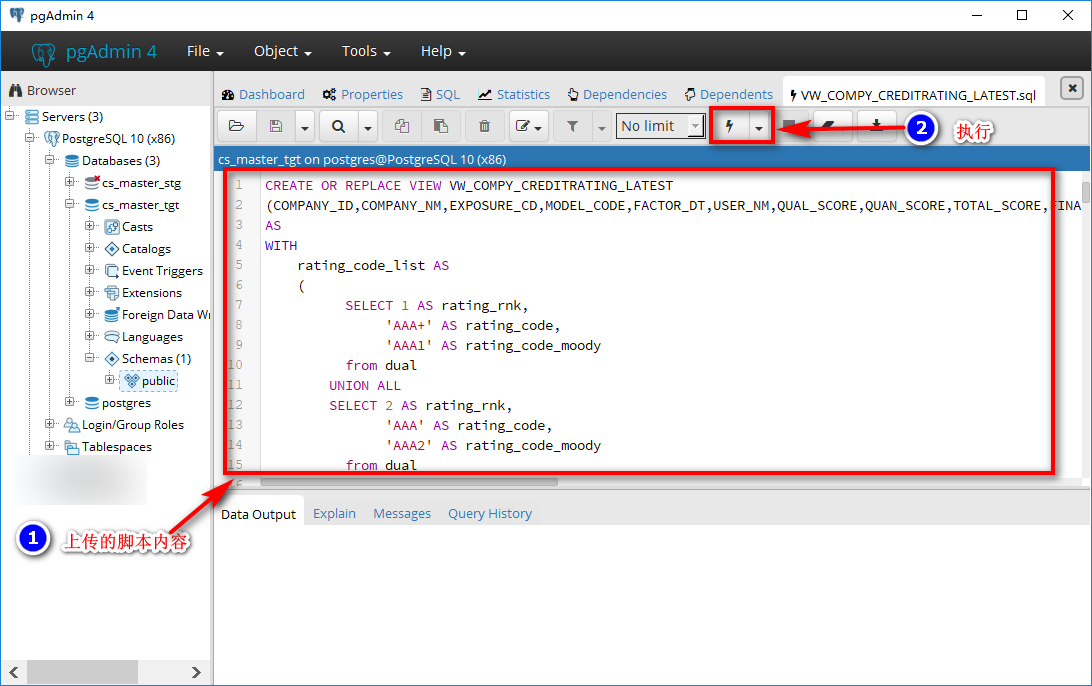
③ 将脚本内容上传至执行区域的按钮



上传结果介绍

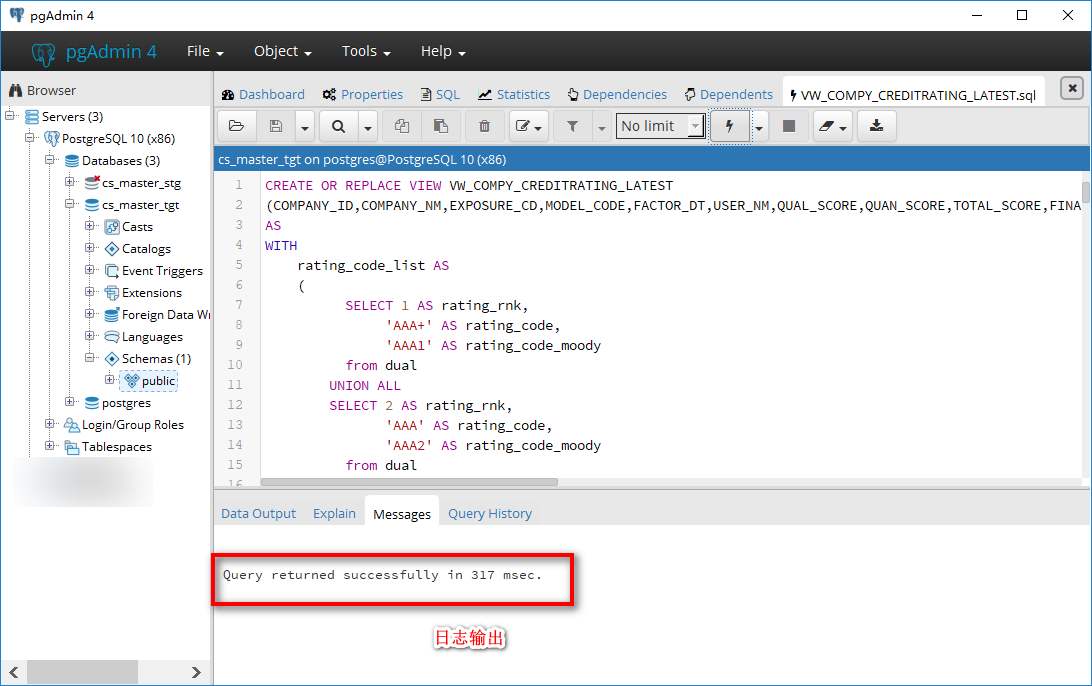
① 上传至执行区域的脚本内容

② 执行按钮



执行结果

① 执行脚本的日志输出

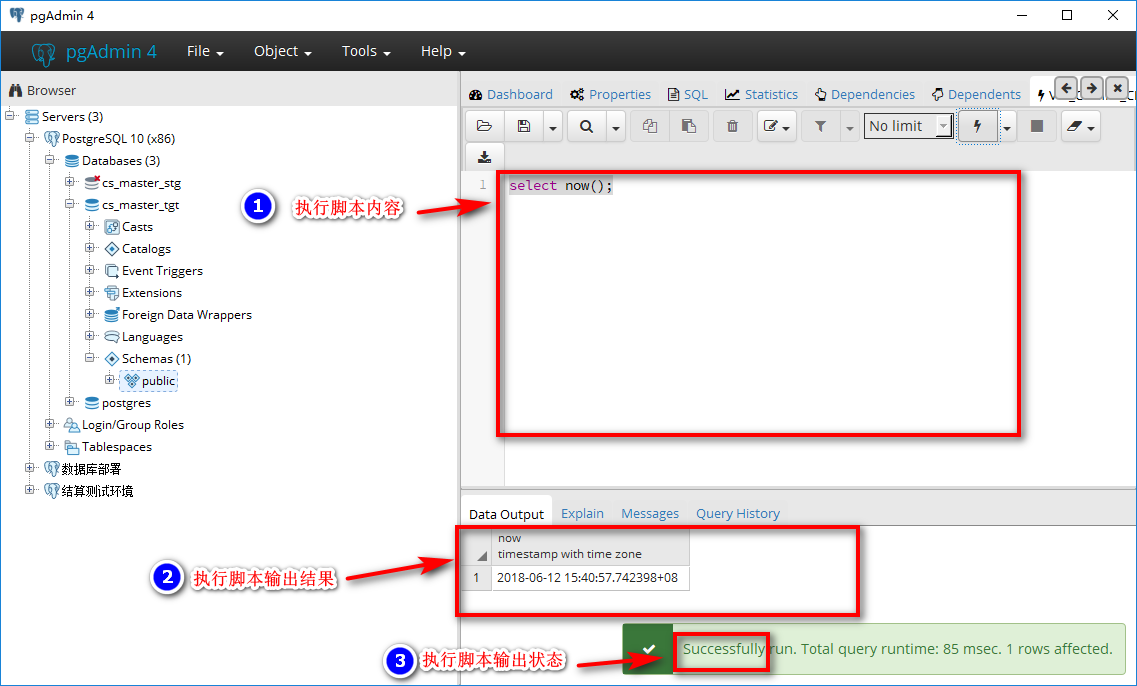


执行查询结果简单介绍

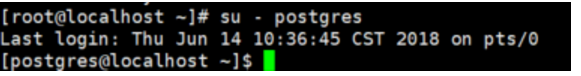
① 执行区域的查询命令

② 查询的结果输出

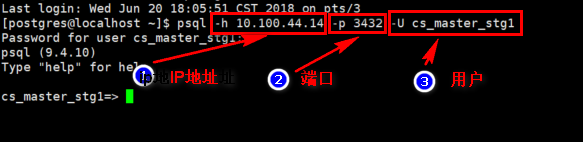
③ 查询执行的状态



### 命令行模式执行脚本示例



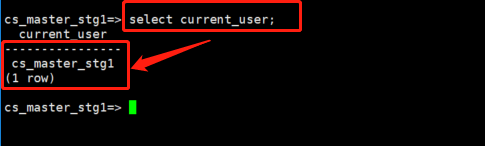
切换至 postgres 用户



① 数据库服务器的IP地址

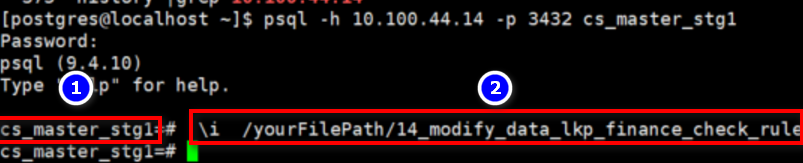
② 数据库的端口号，默认是5432，根据实际情况调整

③ 连接数据库对应的用户名



Select current\_user;

一定是要执行的数据库对应的用户，不可搞到postgres用户下



① 登录的目标数据库

② 在命令行使用 \i 执行脚本的路径+执行的脚本文件名称

Caution

**注意输出的日志内容**

**C:\Users\lirui\AppData\Local\Temp\WeChat Files\666968584712092613.png**

执行完毕，输入 \q 退出命令行

### 增量部署脚本目录结构

|  |
| --- |
| .  ├─cs\_master\_stg  ├─cs\_master\_tgt  ├─QC  │ ├─cs\_master\_stg  │ └─cs\_master\_tgt  └─rollback  ├─cs\_master\_stg  └─cs\_master\_tgt |

### 部署cs\_master\_stg

在增量部署过程中如果出现报错或者其他异常信息，可以查找对应目录下对应对象的回滚脚本进行回滚。

#### 脚本清单

该清单的列举的是全量脚本的目录及文件名称，**不需要执行**。

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | cs\_master\_stg | 00\_clean\_backup\_obj.sql |
| 2 | cs\_master\_stg | 01\_backup\_objects.sql |
| 3 | cs\_master\_stg | 02\_create\_table\_hist\_compy\_bondissuer.sql |
| 4 | cs\_master\_stg | 03\_create\_table\_stg\_compy\_bondissuer.sql |
| 5 | cs\_master\_stg | 04\_recover\_data\_hist\_compy\_bondissuer.sql |
| 6 | cs\_master\_stg | 05\_recover\_data\_stg\_compy\_bondissuer.sql |
| 7 | cs\_master\_stg | 06\_init\_data\_subscribe\_table.sql |
| 8 | cs\_master\_stg | 07\_init\_data\_lkp\_subscribe\_table.sql |
| 9 | cs\_master\_stg | 99\_change\_obj\_owner.sql |
| 10 | rollback\cs\_master\_stg | alter\_table\_hist\_compy\_bondissuer.sql |
| 11 | rollback\cs\_master\_stg | alter\_table\_stg\_compy\_bondissuer.sql |
| 12 | rollback\cs\_master\_stg | alter\_table\_stg\_lkp\_finance\_check\_rule.sql |
| 13 | rollback\cs\_master\_stg | alter\_table\_subscribe\_table.sql |
| 14 | QC\cs\_master\_stg | QC.sql |

#### 执行顺序

该清单下的脚本按照执行顺序依次执行。如执行过程发生异常，则转至2.2.3.4 数据回滚下查找对应对象的回滚脚本。

**CAUTION:**

1. 对于00\_clean\_backup\_obj.sql脚本中包含多个备份对象的清理，部署时需要打开该文件，**逐步执行该脚本，同时记录对应的日志信息**。
2. 对于01\_backup\_objects.sql脚本中包含创建多个备份对象，部署时需要打开该文件，**逐步执行该脚本，同时记录对应的日志信息及其检查备份对象是否成功的语句。**

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | cs\_master\_stg | 00\_clean\_backup\_obj.sql |
| 2 | cs\_master\_stg | 01\_backup\_objects.sql |
| 3 | cs\_master\_stg | 02\_create\_table\_hist\_compy\_bondissuer.sql |
| 4 | cs\_master\_stg | 03\_create\_table\_stg\_compy\_bondissuer.sql |
| 5 | cs\_master\_stg | 04\_recover\_data\_hist\_compy\_bondissuer.sql |
| 6 | cs\_master\_stg | 05\_recover\_data\_stg\_compy\_bondissuer.sql |
| 7 | cs\_master\_stg | 06\_init\_data\_subscribe\_table.sql |
| 8 | cs\_master\_stg | 07\_init\_data\_lkp\_subscribe\_table.sql |
| 9 | cs\_master\_stg | 99\_change\_obj\_owner.sql |
| 10 | QC\cs\_master\_stg | QC.sql |

#### 清理对象

如 2.2.3.2 下**所有**对象执行**成功**，则执行如下脚本。

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | cs\_master\_stg | 00\_clean\_backup\_obj.sql |

#### 数据回滚

如上述步骤2.2.3.2和2.2.3.3执行没有异常，该部分不用执行。如出现异常，则按照异常的对象查找对应的回滚脚本。

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | rollback\cs\_master\_stg | alter\_table\_hist\_compy\_bondissuer.sql |
| 2 | rollback\cs\_master\_stg | alter\_table\_stg\_compy\_bondissuer.sql |
| 3 | rollback\cs\_master\_stg | alter\_table\_stg\_lkp\_finance\_check\_rule.sql |
| 4 | rollback\cs\_master\_stg | alter\_table\_subscribe\_table.sql |

### 部署cs\_master\_tgt

在增量部署过程中如果出现报错或者其他异常信息，可以查找对应目录下对应对象的回滚脚本进行回滚。

#### 脚本清单

该清单的列举的是全量脚本的目录及文件名称，不需要执行。

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | cs\_master\_tgt | 00\_clean\_backup\_obj.sql |
| 2 | cs\_master\_tgt | 01\_backup\_objs\_and\_datas.sql |
| 3 | cs\_master\_tgt | 02\_create\_table\_compy\_bondissuer.sql |
| 4 | cs\_master\_tgt | 03\_recover\_data\_compy\_bondissuer.sql |
| 5 | cs\_master\_tgt | 04\_create\_view\_vw\_bond\_rating\_cacul.sql |
| 6 | cs\_master\_tgt | 05\_init\_data\_lkp\_ratingcd\_xw.sql |
| 7 | cs\_master\_tgt | 99\_change\_obj\_owner.sql |
| 8 | rollback\cs\_master\_tgt | alter\_table\_compy\_bondissuer.sql |
| 9 | QC\cs\_master\_tgt | QC.sql |

#### 执行顺序

该清单下的脚本按照执行顺序依次执行。如执行过程发生异常，则转至2.2.4.4 数据回滚下查找对应对象的回滚脚本。

**CAUTION:**

1. 对于00\_clean\_backup\_obj.sql脚本中包含多个备份对象的清理，部署时需要打开该文件，**逐步执行该脚本，同时记录对应的日志信息。**
2. 对于01\_backup\_objs\_and\_datas.sql脚本中包含创建多个备份对象，部署时需要打开该文件，**逐步执行该脚本，同时记录对应的日志信息及其检查备份对象是否成功的语句。**

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | cs\_master\_tgt | 00\_clean\_backup\_obj.sql |
| 2 | cs\_master\_tgt | 01\_backup\_objs\_and\_datas.sql |
| 3 | cs\_master\_tgt | 02\_create\_table\_compy\_bondissuer.sql |
| 4 | cs\_master\_tgt | 03\_recover\_data\_compy\_bondissuer.sql |
| 5 | cs\_master\_tgt | 04\_create\_view\_vw\_bond\_rating\_cacul.sql |
| 6 | cs\_master\_tgt | 05\_init\_data\_lkp\_ratingcd\_xw.sql |
| 7 | cs\_master\_tgt | 99\_change\_obj\_owner.sql |
| 8 | QC\cs\_master\_tgt | QC.sql |

#### 清理对象

如 2.2.4.2 下**所有**对象执行**成功**，则执行如下脚本。

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | cs\_master\_tgt | 00\_clean\_backup\_obj.sql |

#### 数据回滚

如上述步骤2.2.4.2和2.2.4.3执行没有异常，该部分不用执行。如出现异常，则按照异常的对象查找对应的回滚脚本。

|  |  |  |
| --- | --- | --- |
| **编号** | **目录** | **脚本名称** |
| 1 | rollback\cs\_master\_tgt | alter\_table\_compy\_bondissuer.sql |

# 结算外网服务器上更新数据拉取程序

## 调整拉取数据起始位置

在D:/creditmaster/incoming下创建一个finish后缀的时间戳文件，具体时间戳取决于2.1步初始化时获取的数据包日期，比如数据包日期为20180605则创建的finish文件为20180605000000.finish，表示20180605之前的数据不会再次被拉取

## 调整尝试拉取频率

打开配置文件D:/MasterDataPull/conf/Application.properties，修改sftp.interval参数为1800，然后重启服务。

在cmd中运行jps|findstr DownloadFromSftp查出运行中的进程号（如不存在在说明未启动）

然后cmd中运行taskkill /pid 进程号 /f来结束进程

双击D:/MasterDataPull/bin/DownloadFromSftp.vbe重新启动拉取程序

再次cmd中运行jps|findstr DownloadFromSftp可以看到新启动的拉取程序进程号

conf/Application.properties

# 数据同步服务器上更新数据同步程序

## 更新文件

使用cmsync用户将提供的MasterDataSync.sh和MasterDataSync-1.1.1.jar分别复制到

/CMSYNC/cmapp/MasterDataSync/bin和/CMSYNC/cmapp/MasterDataSync/lib下并覆盖原有文件

然后修改权限并调整换行符

[cmsync@CM-SYNC MasterDataSync-1.1]$ chmod u+x bin/\*

[cmsync@CM-SYNC MasterDataSync-1.1]$ dos2unix bin/\*

## 调整尝试拉取频率

使用cmsync用户执行crontab –e

将其中第一行调度配置修改为

\*/30 \* \* \* 1-5 /CMSYNC/cmapp/MasterDataSync-1.1/bin/MasterDataSyncAll.sh &>> /CMSYNC/cmapp/MasterDataSync-1.1/log/mds.log

修改完成后:wq保存退出

## 增加每日同步数据至结算EDW的调度配置

使用cmsync用户执行crontab –e增加一行内容:

0 7 \* \* 1-5 /CMSYNC/cmapp/MasterDataSync-1.1/bin/ExportToDw.sh

修改完成后:wq保存退出

# 应用程序部署

## 环境准备

操作系统: redhat 7.1

Master版本:v2.0

安装包里面一共有如下依赖包,需要先安装完这些依赖包才能开始部署:

1. zlib-1.2.8.tar.gz
2. libxml2-2.9.4.tar.gz
3. pcre-8.39.tar.gz
4. nginx-1.10.1.tar.gz
5. redis-3.2.3.tar.gz
6. jdk-8u131-linux-x64.rpm
7. python-devel-2.7.5-48.el7.x86\_64.rpm

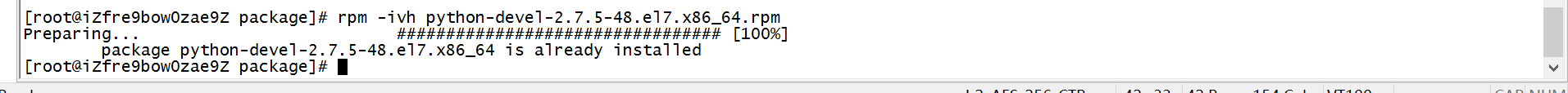
### 安装python-devel

1.如果有网络，可以通过yum去安装pythondevel,yum install python-devel.x86\_64。

2.如果没有网络环境，使用提供的python-devel-2.7.5-48.el7.x86\_64.rpm

yum localinstall python-devel-2.7.5-48.el7.x86\_64.rpm

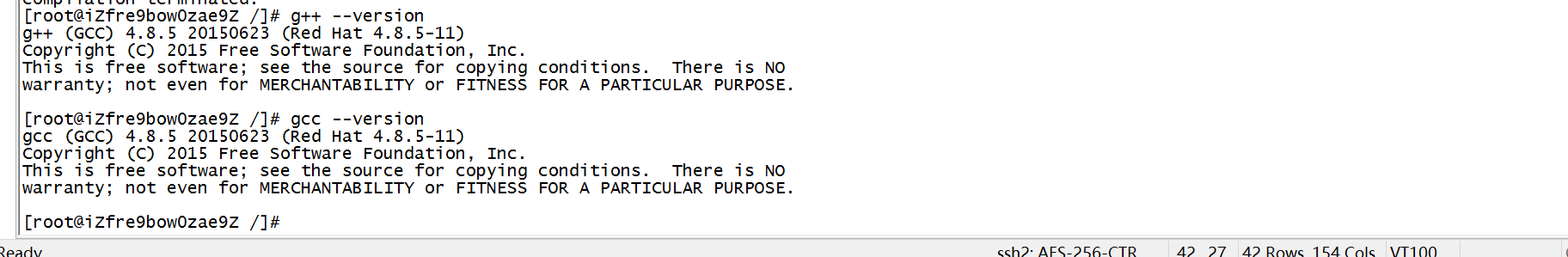
出现如下界面，代表安装成功



### 安装c++编译器

1. 有网络的时候直接运行yum install -y gcc gcc-c++

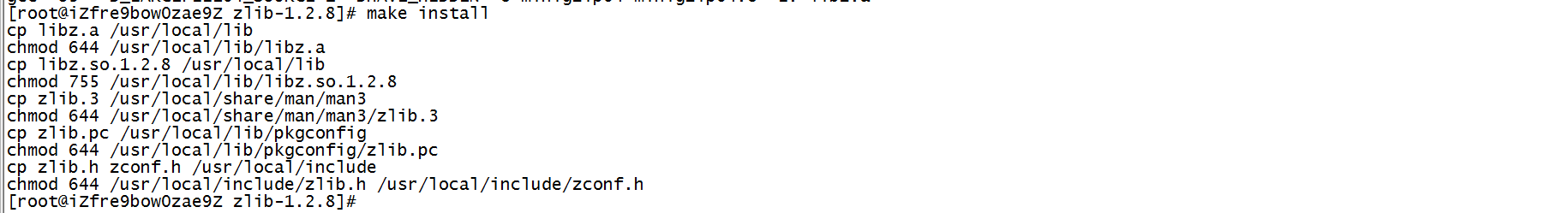
安装完成后可以通过gcc --version 和g++ -- version检查是否安装成功。



### 安装zlib

1. tar zxvf zlib-1.2.8.tar.gz
2. cd zlib-1.2.8
3. ./configure
4. make
5. make install

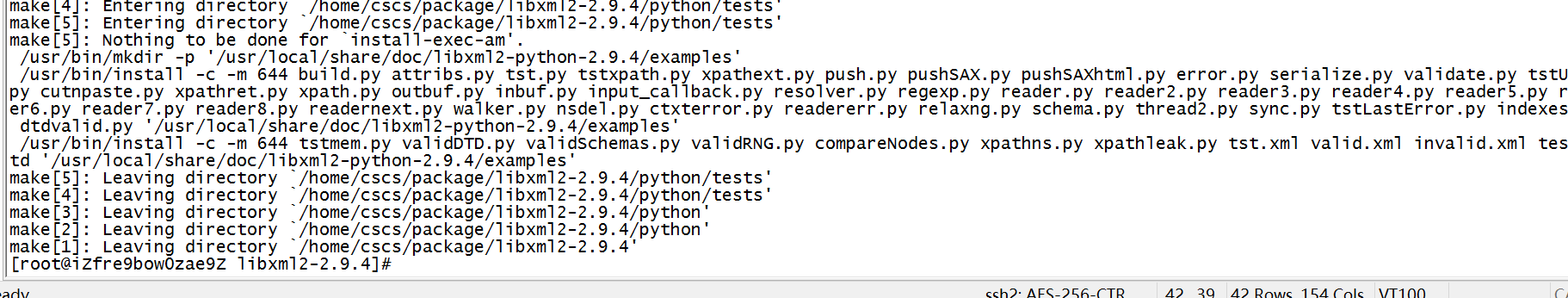
出现如下界面，代表安装成功



### 安装libxml2

1. tar -zxvf libxml2-2.9.4.tar.gz
2. cd libxml2-2.9.4
3. ./configure
4. make
5. make install

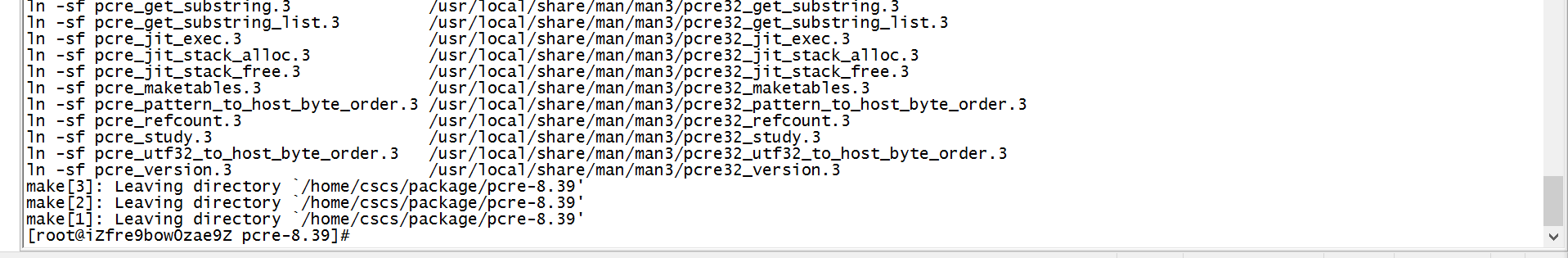
出现如下界面代表安装成功:



### 安装 pcre

1. tar -zxvf pcre-8.39.tar.gz
2. cd pcre-8.39/
3. ./configure
4. make
5. make install

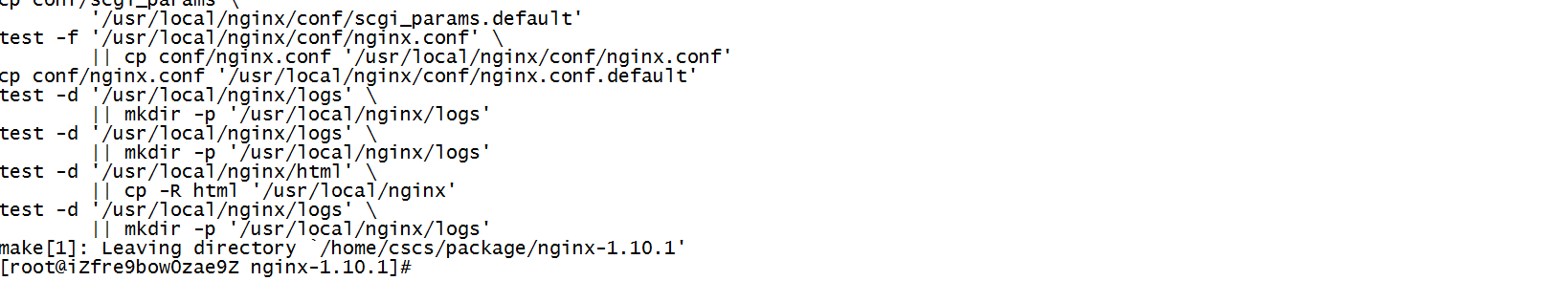
出现如下界面代表安装成功:



### 安装nginx

1. tar -zxvf nginx-1.10.1.tar.gz
2. cd nginx-1.10.1/
3. ./configure
4. make
5. make install

出现如下界面代表安装成功:



### 安装JDK 8

需要JDK8及以上版本。

[root@CM-WEB ~]# rpm -ivh jdk-8u131-linux-x64.rpm

vim /etc/profile

#将下面的内容添加进去：

export JAVA\_HOME=/usr/java/jdk1.8.0\_131

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

export CLASSPATH=.:${JAVA\_HOME}/lib/tools.jar:${JAVA\_HOME}/lib/dt.jar

source /etc/profile

验证是否安装成功: java --version,出现如下信息代表安装成功

Java(TM) SE Runtime Environment (build 1.8.0\_131-b11)

Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)

## 安装和启动redis

安装redis，

1. tar -zxvf redis-3.2.3.tar.gz
2. cd redis-3.2.3
3. 替换redis.conf文件,安装包里有一个已经定义好的redis.conf文件，将它拷贝到解压出来的文件夹里面

cp ../redis.conf ./

1. make
2. make install
3. 启动redis

redis-server /root/redis-3.2.3/redis.conf &

1. 验证是否启动成功

netstat -lnp | grep 6379 有进程代表redis启动成功

## 配置nginx

安装包里面有一个文件nginx.conf，将该文件拷贝到/usr/local/nginx/conf/nginx.conf，并将路径/root/creditmaster修改为/CMWEB/creditmaster，然后启动nginx。

cp nginx.conf /usr/local/nginx/conf/nginx.conf

/usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf

/usr/local/nginx/sbin/nginx -s reload

验证nginx是否启动成功:

ps -aux |grep nginx

注:可以根据服务器的实际情况调整ngnix的配置

* + 第3行 worker\_processes 8; 8代表CPU的核数目
  + 本地后台服务端口 server 127.0.0.1:7777; 代表后台启动的端口为7777
  + 后台对外服务端口 listen 3030; 代表后台对外服务的端口为3030

## 后台部署

1. 创建后台部署包目录/CMWEB/creditmaster/
2. 将cmaster-0.0.1-SNAPSHOT.jar 拷贝到/CMWEB/creditmaster/
3. 将后台启动命令文件startServer.sh 拷贝到/CMWEB/creditmaster/
4. 修改startServer.sh文件中的数据库链接地址和密码为生产系统实际配置
   1. 地址: --spring.datasource.url=jdbc:postgresql://10.128.13.17:3432/cs\_master\_tgt
   2. 数据库用户: --spring.datasource.username=cs\_master\_tgt
   3. 数据库密码: --spring.datasource.password=abc123
5. 进入目录/CMWEB/creditmaster/ (./startServer.sh ) 启动进程
6. 查看startServer.sh 指定的日志输出文件
7. 通过http://10.128.13.16:1010/v2/api.html 即可查看是否部署成功，出现如下界面，代表后端部署成功。



## 前台部署

1. 准备前台发布的包dist.zip
2. 将dist.zip解压到/CMWEB/creditmaster 如果dist目录已经存在请删除
3. 后端的访问地址为**后台部署第7步中的**http://10.128.13.16:3030/v2/
4. 修改main.\*\*\*\*\*.bundle.js 文件，用后端访问地址替换api\_endpoint:"http://ip:3030 /v2中的对应部分
5. 通过<http://10.128.13.16:3030>查看前台发布

出现如下界面，代表前端部署成功



## 清除缓存

在版本部署成功后，需要手动删除nginx缓存保证更新立即生效.

1. rm -rf /usr/local/nginx/client\_body\_temp
2. rm -rf /usr/local/nginx/fastcgi\_temp
3. rm -rf /usr/local/nginx/proxy\_temp
4. rm -rf /usr/local/nginx/scgi\_temp
5. rm -rf /usr/local/nginx/uwsgi\_temp
6. /usr/local/nginx/sbin/nginx -s reload 重启服务

## 常见错误日志

1. 后台错误日志 (./startSever 指定的日志文件)
2. Nginx 错误日志 /usr/local/nginx/logs