2017 年全国职业院校技能大赛（高职组）“云计算技术与应用”C 卷答案

云平台架构

赛项系统架构如图 1 所示，IP 地址规划如表 1 所示。

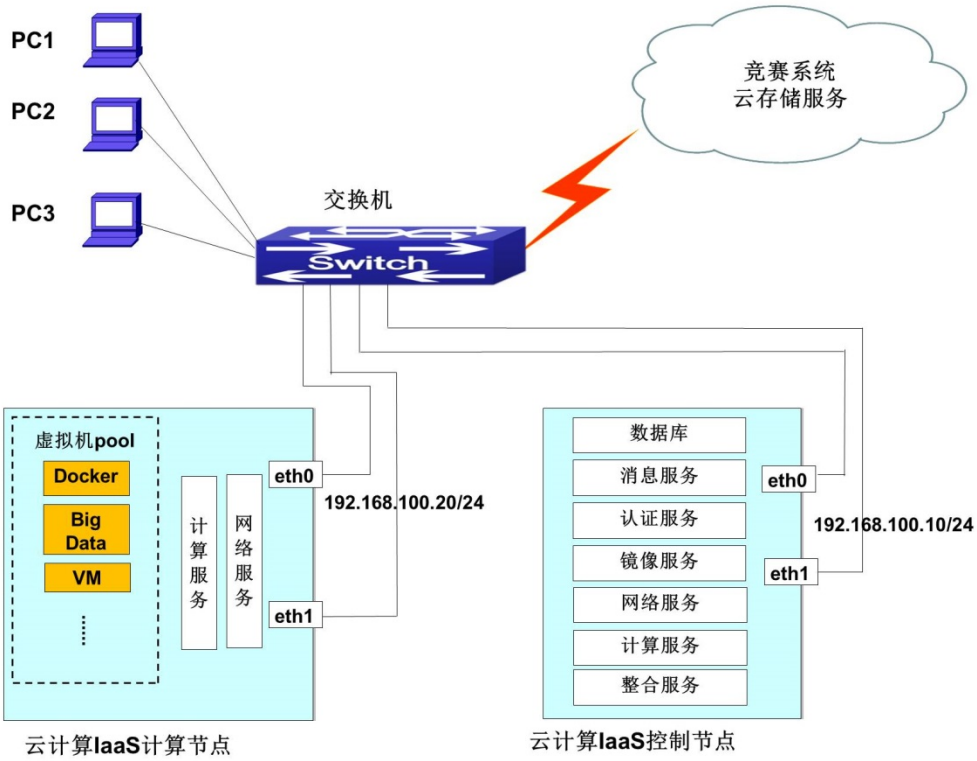


图 1　系统架构图

表 1　IP 地址规划表

设备名称	接口	IP 地址	说明
控制节点服务器	eth0	192.168.100.10/24	Vlan 100
	eth1	192.168.200.10/24（初始 IP）	Vlan 200
计算节点服务器	eth0	192.168.100.20/24	Vlan 100
	eth1	192.168.200.20/24（初始 IP）	Vlan 200
PC-1	本地连接	172.16.x.2/16	Vlan 1
PC-2	本地连接	172.16.x.3/16	Vlan 1
PC-3	本地连接	172.16.x.4/16	Vlan 1
交换机	Vlan 1	172.16.x.1/16	
	Vlan 100	192.168.100.1/24	
	Vlan 200	192.168.200.1/24	

注：表 1 中 x 为考位号。

根据图 1 和表 1 给出的云平台信息，检查硬件设备及网络连接关系。

场景描述

某企业计划搭建私有云平台，以实现计算资源的池化弹性管理、企业应用的集中管理、统一安全认证和授权管理。根据云平台的架构，完成系统部署、云存储网盘 web 开发、客户端开发及大数据开发。

第一部分：云计算平台搭建（20 分）

任务一、IaaS 云计算基础架构平台搭建（15 分）

根据图 1 给出的云平台架构及以下题干中提供的信息，修改云平台 IaaS 各节点的系统配置，完成云计算基础架构平台的搭建及相应的答题。

1. 环境配置(1 分)

设置主机名、防火墙及 selinux 如下：

（1）控制节点主机名 controller；计算节点主机名：compute。

（2）各个节点关闭防火墙，开机不启动。

（3）各个节点 selinux 为 permissive。

使用 sestatus 命令查询 selinux 的状态，以文本形式提交查询命令及结果到答题框。

```
[root@controller ~]# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          permissive
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      28
[root@compute ~]# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          permissive
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      28
```

SELinux 状态 enabled 得 0.5 分
模式为 permissive 的 0.5 分

2. yum 源配置（1 分）

在控制节点上使用 SecureFX 软件上传镜像文件 CentOS-7-x86_64-DVD-1511.iso 和 XianDian-IaaS-v2.1.iso 到/opt 路径下，创建两个目录/opt/centos、/opt/iaas，并将以上镜

像文件分别挂载到两个目录下。配置控制节点本地 yum 源文件 local.repo，搭建 ftp 服务器指向存放 yum 源路径；配置计算节点 yum 源文件 ftp.repo 使用之前配置的控制节点 ftp 作为 yum 源，其中两个节点的地址使用主机名表示。使用 cat 命令查看计算节点的 yum 源全路径配置文件，以文本形式提交查询命令及结果到答题框。

```
[root@compute ~]# cat /etc/yum.repos.d/ftp.repo
[iaas]
name=iaas
baseurl=ftp://controller/iaas/iaas-repo
enabled=1
gpgcheck=0

[centos]
name=centos
baseurl=ftp://controller/centos
enabled=1
gpgcheck=0
```

查看配置文件名称为 ftp.repo 得 0.5 分
配置文件包含 iaas 和 centos 的源正确配置且使用 ftp 格式配置地址得 0.5 分

3. 环境变量配置（1 分）

在控制节点和计算节点分别安装 iaas-xiandian 包，完成配置文件中基本变量的配置。根据表 2 完成指定变量的配置，以文本形式提交控制节点的配置文件到答题框。

表 2　变量配置表

服务	变量	参数/密码
Mysql	root	000000
	Keystone	000000
	Glance	000000
	Nova	000000
	Neutron	000000
	trove	000000
	cinder	000000
	Heat	000000
Keystone	DOMAIN_NAME	demo
	Admin	000000
	Glance	000000
	Nova	000000
	Neutron	000000
	trove	000000
	cinder	000000
	swift	000000
	Heat	000000
Neutron	Metadata	000000
	External Network	enp9s0
RabbitMQ	RabbitMQ user	openstack
	RabbitMQ pass	000000

```
[root@controller ~]# cat /etc/xiandian/openrc.sh
##--------------------system Config--------------------##
##Controller Server Manager IP. example:x.x.x.x
HOST_IP=192.168.100.10

##Controller Server hostname. example:controller
HOST_NAME=controller

##Compute Node Manager IP. example:x.x.x.x
HOST_IP_NODE=192.168.100.20

##Compute Node hostname. example:compute
HOST_NAME_NODE=compute

##--------------------Rabbit Config ------------------##
```

```
##user for rabbit. example:openstack
RABBIT_USER=openstack

##Password for rabbit user .example:000000
RABBIT_PASS=000000

##--------------------MySQL Config---------------------##
##Password for MySQL root user . exmaple:000000
DB_PASS=000000

##--------------------Keystone Config------------------##
##Password for Keystore admin user. exmaple:000000
DOMAIN_NAME=demo
ADMIN_PASS=000000
DEMO_PASS=000000

##Password for Mysql keystore user. exmaple:000000
KEYSTONE_DBPASS=000000

##--------------------Glance Config--------------------##
##Password for Mysql glance user. exmaple:000000
GLANCE_DBPASS=000000

##Password for Keystore glance user. exmaple:000000
GLANCE_PASS=000000

##--------------------Nova Config----------------------##
##Password for Mysql nova user. exmaple:000000
NOVA_DBPASS=000000

##Password for Keystore nova user. exmaple:000000
NOVA_PASS=000000

##--------------------Neturon Config-------------------##
##Password for Mysql neutron user. exmaple:000000
NEUTRON_DBPASS=000000

##Password for Keystore neutron user. exmaple:000000
NEUTRON_PASS=000000

##metadata secret for neutron. exmaple:000000
METADATA_SECRET=000000

##External Network Interface. example:eth1
```

```
INTERFACE_NAME=enp9s0

##First Vlan ID in VLAN RANGE for VLAN Network. exmaple:101
#minvlan=

##Last Vlan ID in VLAN RANGE for VLAN Network. example:200
#maxvlan=

##--------------------Cinder Config--------------------##
##Password for Mysql cinder user. exmaple:000000
CINDER_DBPASS=000000

##Password for Keystore cinder user. exmaple:000000
CINDER_PASS=000000

##Cinder Block Disk. example:md126p3
#BLOCK_DISK=

##--------------------Trove Config--------------------##
##Password for Mysql Trove User. exmaple:000000
TROVE_DBPASS=000000

##Password for Keystore Trove User. exmaple:000000
TROVE_PASS=000000

##--------------------Swift Config--------------------##
##Password for Keystore swift user. exmaple:000000
SWIFT_PASS=000000

##The NODE Object Disk for Swift. example:md126p4.
#OBJECT_DISK=

##The NODE IP for Swift Storage Network. example:x.x.x.x.
#STORAGE_LOCAL_NET_IP=

##--------------------Heat Config--------------------##
##Password for Mysql heat user. exmaple:000000
HEAT_DBPASS=000000

##Password for Keystore heat user. exmaple:000000
HEAT_PASS=000000

##--------------------Ceilometer Config---------------##
##Password for Mysql ceilometer user. exmaple:000000
```

```
#CEILOMETER_DBPASS=

##Password for Keystore ceilometer user. exmaple:000000
#CEILOMETER_PASS=

##--------------------AODH Config----------------##
##Password for Mysql AODH user. exmaple:000000
#AODH_DBPASS=

##Password for Keystore AODH user. exmaple:000000
#AODH_PASS=
```

主机名和 IP 进行配置并配置正确得 0.5 分
表格中的参数全部进行配置并配置正确得 0.5 分

4. keystone 安装（2 分）

根据平台安装步骤安装 keystone 认证服务，admin-openrc.sh 文件在/etc/keystone 路径下。列出数据库 keystone 中的所有表，将其中 keystone 的数据库导出为 keystone.sql 文件，并使用 sed 命令显示文件 keystone.sql 中前 20 行内容。以文本形式提交以上所有命令及结果到答题框。

```
MariaDB [keystone]> show tables;
+------------------------+
| Tables_in_keystone     |
+------------------------+
| access_token           |
| assignment             |
| config_register        |
| consumer               |
| credential             |
| domain                 |
| endpoint               |
| endpoint_group         |
| federated_user         |
| federation_protocol    |
| group                  |
| id_mapping             |
| identity_provider      |
| idp_remote_ids         |
| implied_role           |
| local_user             |
| mapping                |
| migrate_version        |
| password               |
| policy                 |
| policy_association     |
```

```
| project                 |
| project_endpoint        |
| project_endpoint_group |
| region                  |
| request_token           |
| revocation_event        |
| role                    |
| sensitive_config        |
| service                 |
| service_provider        |
| token                   |
| trust                   |
| trust_role              |
| user                    |
| user_group_membership   |
| whitelisted_config      |
+-------------------------+
37 rows in set (0.00 sec)
[root@controller ~]# mysqldump -uroot -p000000 keystone>keystone.sql
[root@controller ~]# sed -n '1,20p' keystone.sql
-- MySQL dump 10.16  Distrib 10.1.17-MariaDB, for Linux (x86_64)
--
-- Host: localhost       Database: keystone
-- ------------------------------------------------------
-- Server version       10.1.17-MariaDB

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT
*/;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION
*/;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
```

-- Table structure for table `access_token` --
使用命令展示 keystone 数据库中所有表得 0.5 分 导出 keystone 数据库得 0.5 分 使用 sed 查询导出文件前 20 行内容的命令得 0.5 分 前 20 行内容一致得 0.5 分

5. nova 安装（2 分）

根据平台安装步骤安装 nova 计算服务，使用 nova 相关命令查询 nova 服务状态列表，以文本形式提交查询命令和结果到答题框。

[root@controller ~]# nova service-list +----+------------------+------------+----------+---------+-------+----------------------------+----------- ------+ | Id | Binary | Host | Zone | Status | State | Updated_at | Disabled Reason | +----+------------------+------------+----------+---------+-------+----------------------------+----------- ------+ | 1 | nova-scheduler | controller | internal | enabled | up | 2017-06-01T04:04:54.000000 | - | | 2 | nova-consoleauth | controller | internal | enabled | up | 2017-06-01T04:04:54.000000 | - | | 3 | nova-conductor | controller | internal | enabled | up | 2017-06-01T04:04:54.000000 | - | | 18 | nova-compute | compute | nova | enabled | up | 2017-06-01T04:04:58.000000 | - | +----+------------------+------------+----------+---------+-------+----------------------------+----------- ------+
命令正确得 1 分 查询结果中服务名称一致、状态正确得 1 分

6. neutron 安装（2 分）

根据平台安装步骤安装 neutron 网络服务，并配置为 GRE 网络。使用 neutron 命令查询网络服务的列表信息，并以表 3 的形式显示出来，以文本形式提交相应的查询命令以及结果到答题框。

表 3　网络服务显示表

binary	agent_type	alive

[root@controller ~]# neutron agent-list -c binary -c agent_type -c alive +--------------------------+--------------------+-------+ | binary | agent_type | alive | +--------------------------+--------------------+-------+ | neutron-openvswitch-agent | Open vSwitch agent | :-) |

```
| neutron-dhcp-agent          | DHCP agent          | :-)   |
| neutron-l3-agent            | L3 agent            | :-)   |
| neutron-metadata-agent      | Metadata agent      | :-)   |
+---------------------------+--------------------+-------+
```

查询命令完全一致得 1 分
查询结果表格格式先后顺序一致、服务名称一致、状态正确得 1 分

7. 网络创建（2 分）

创建云主机外部网络 ext-net，子网为 ext-subnet，云主机浮动 IP 可用网段为 192.168.200.100 ~ 192.168.200.200，网关为 192.168.200.1。创建云主机内部网络 int-net1，子网为 int-subnet1，云主机子网 IP 可用网段为 10.0.0.100 ~ 10.0.0.200，网关为 10.0.0.1；创建云主机内部网络 int-net2，子网为 int-subnet2，云主机子网 IP 可用网段为 10.0.1.100 ~ 10.0.1.200，网关为 10.0.1.1。添加名为 ext-router 的路由器，添加网关在 ext-net 网络，添加内部端口到 int-net1 网络，完成内部网络 int-net1 和外部网络的连通。

使用 neutron 命令查询所创建子网的列表信息，以文本形式提交查询命令及结果到答题框。

```
[root@controller ~]# neutron subnet-list
+--------------------------------------+------+-----------------+-----------------------------------------------
----------+
| id                                   | name | cidr            | allocation_pools
|
+--------------------------------------+------+-----------------+-----------------------------------------------
----------+
| 3c13a12d-37a1-4022-b216-ad10fc4c7f92 | ext-subnet    | 192.168.200.0/24 | {"start":
"192.168.200.100", "end": "192.168.200.200"} |
| 6c671cb2-7241-4891-b560-ad94e6d9b2ae | int-subnet1   | 10.0.0.0/24      | {"start":
"10.0.0.100", "end": "10.0.0.200"}           |
| 6c671cb2-7241-4891-b560-ad94e6d9b2ae | int-subnet2   | 10.0.1.0/24      | {"start":
"10.0.1.100", "end": "10.0.1.200"}           |
+--------------------------------------+------+-----------------+-----------------------------------------------
----------+
```

查询命令完全一致得 1 分
查询结果名称、网段、起始地址和结束地址前后对应正确得 1 分

8. trove 安装（2 分）

在控制节点修改提供的脚本 iaas-install-trove.sh，使其连接的网络为 int-net1，安装数据库 trove 服务。查询所有的数据库实例列表。将以上操作命令及检查结果填入答题框。

```
[root@controller ~]# trove list
+----+------+-----------+-------------------+--------+-----------+------+
| ID | Name | Datastore | Datastore Version | Status | Flavor ID | Size |
+----+------+-----------+-------------------+--------+-----------+------+
+----+------+-----------+-------------------+--------+-----------+------+
```

查询命令一致得 1 分

查询结果完全一致得 1 分

9. trove 管理（2 分）

上传软件包中的镜像文件 MySQL_5.6_XD.qcow2 到云平台，并创建 trove 数据库服务存储类型 mysql，使用上传的镜像更新该数据库类型的版本信息和镜像信息，以文本形式提交操作命令及结果到答题框。

```
# glance image-create --name "mysql-5.6" --disk-format qcow2    --container-format bare
--progress < images/MySQL_5.6_XD.qcow2
[==============================>] 100%
+------------------+--------------------------------------+
| Property               | Value                                                |
+------------------+--------------------------------------+
| checksum               | 156acbb0b92037053face8b0f97674bd         |
| container_format | bare                                                  |
| created_at             | 2017-04-28T18:10:37Z                          |
| disk_format           | qcow2                                                 |
| id                          | 8d9b758e-f22c-4cde-8abb-3aec523564cd |
| min_disk               | 0                                                      |
| min_ram                 | 0                                                      |
| name                     | mysql-5.6                                           |
| owner                    | 541c3d096ba54f948ddc3a74c26ba79c         |
| protected              | False                                                 |
| size                       | 592800256                                           |
| status                   | active                                                |
| tags                       | []                                                     |
| updated_at            | 2017-04-28T18:10:41Z                          |
| virtual_size         | None                                                  |
| visibility             | private                                               |
+------------------+--------------------------------------+
# trove-manage datastore_update mysql "
Option "verbose" from group "DEFAULT" is deprecated for removal.    Its value may be
silently ignored in the future.
2017-04-28 14:10:50.117 21395 DEBUG oslo_db.sqlalchemy.engines [-] MySQL server
mode set to
STRICT_TRANS_TABLES,STRICT_ALL_TABLES,NO_ZERO_IN_DATE,NO_ZERO_D
ATE,ERROR_FOR_DIVISION_BY_ZERO,TRADITIONAL,NO_AUTO_CREATE_USER,
NO_ENGINE_SUBSTITUTION _check_effective_sql_mode
/usr/lib/python2.7/site-packages/oslo_db/sqlalchemy/engines.py:256
Datastore 'mysql' updated.
# Glance_Image_ID=$(glance image-list | awk '/ mysql-5.6 / { print $2 }')
# trove-manage datastore_version_update mysql mysql-5.6 mysql ${Glance_Image_ID} " 1
Option "verbose" from group "DEFAULT" is deprecated for removal.    Its value may be
silently ignored in the future.
2017-04-28 14:13:43.035 21801 DEBUG oslo_db.sqlalchemy.engines [-] MySQL server
```

mode set to STRICT_TRANS_TABLES,STRICT_ALL_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,TRADITIONAL,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION _check_effective_sql_mode /usr/lib/python2.7/site-packages/oslo_db/sqlalchemy/engines.py:256 Datastore version 'mysql-5.6' updated.
上传镜像文件得 0.5 分 创建数据库服务存储类型得 0.5 分 数据库类型版本信息和镜像信息进行更新得 1 分

任务二、PaaS 云计算开发服务平台搭建（5 分）

PaaS 平台架构如图 2 所示。

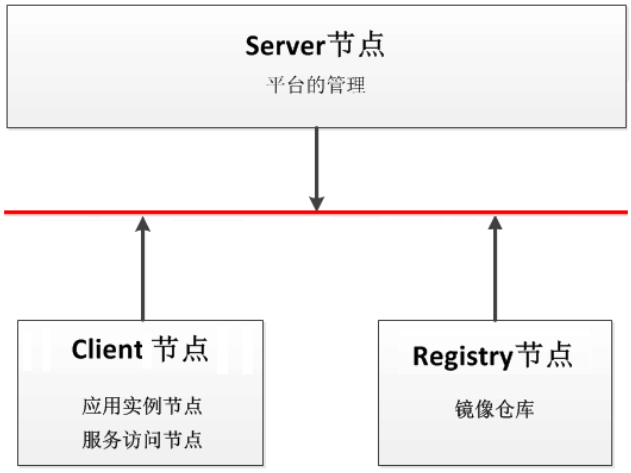


图 2　PaaS 平台架构图

根据 PaaS 平台的部署架构，PaaS 平台部署在 IaaS 平台的 3 台云主机上。采用分开安装的方式部署 PaaS 平台，其中 Registry 云主机上部署 Registry 节点，Server 云主机上部署 Server 节点，Client 云主机上部署 Client 节点。每个云主机配置如下：

(1) 系统配置：Registry 节点：2CPU，4G 内存，60G 硬盘；Server 节点：2CPU，4G 内存，60G 硬盘；Client 节点：2CPU，4G 内存，60G 硬盘。

(2) 镜像文件：centos_7-x86_64_xiandian.qcow2。

(3) IP：Registry、Server 和 Client 节点 IP 动态分配。

(4) 主机名配置：Registry 节点的主机名：Registry；Server 节点的主机名：Server；Client 节点的主机名：Client。

根据配置要求，完成配置文件的定义与修改，搭建 PaaS 平台。

1. Rancher 平台管理（2 分）

进入 Rancher 管理平台“API”菜单，找到“环境 API Keys”的端点 URL 地址，使用 curl 命令进行查询，以文本形式提交查询命令及结果到答题框。

[root@server ~]# curl http://192.168.200.102/v1/projects/1a5
{"id":"1a5","type":"project","links":{"self":"http://192.168.200.102/v1/projects/1a5","agents":"http://192.168.200.102/v1/projects/1a5/agents","auditLogs":"http://192.168.200.102/v1/projects/1a5/auditlogs","backupTargets":"http://192.168.200.102/v1/projects/1a5/backuptargets","backups":"http://192.168.200.102/v1/projects/1a5/backups","certificates":"http://192.168.200.102/v1/projects/1a5/certificates","configItemStatuses":"http://192.168.200.102/v1/projects/1a5/configitemstatuses","containerEvents":"http://192.168.200.102/v1/projects/1a5/containerevents","credentials":"http://192.168.200.102/v1/projects/1a5/credentials","environments":"http://192.168.200.102/v1/projects/1a5/environments","externalEvents":"http://192.168.200.102/v1/projects/1a5/externalevents","healthcheckInstanceHostMaps":"http://192.168.200.102/v1/projects/1a5/healthcheckinstancehostmaps","hosts":"http://192.168.200.102/v1/projects/1a5/hosts","images":"http://192.168.200.102/v1/projects/1a5/images","instanceLinks":"http://192.168.200.102/v1/projects/1a5/instancelinks","instances":"http://192.168.200.102/v1/projects/1a5/instances","ipAddresses":"http://192.168.200.102/v1/projects/1a5/ipaddresses","labels":"http://192.168.200.102/v1/projects/1a5/labels","mounts":"http://192.168.200.102/v1/projects/1a5/mounts","networks":"http://192.168.200.102/v1/projects/1a5/networks","physicalHosts":"http://192.168.200.102/v1/projects/1a5/physicalhosts","ports":"http://192.168.200.102/v1/projects/1a5/ports","projectMembers":"http://192.168.200.102/v1/projects/1a5/projectmembers","serviceConsumeMaps":"http://192.168.200.102/v1/projects/1a5/serviceconsumemaps","serviceEvents":"http://192.168.200.102/v1/projects/1a5/serviceevents","serviceExposeMaps":"http://192.168.200.102/v1/projects/1a5/serviceexposemaps","services":"http://192.168.200.102/v1/projects/1a5/services","snapshots":"http://192.168.200.102/v1/projects/1a5/snapshots","storagePools":"http://192.168.200.102/v1/projects/1a5/storagepools","userPreferences":"http://192.168.200.102/v1/projects/1a5/userpreferences","volumes":"http://192.168.200.102/v1/projects/1a5/volumes","accounts":"http://192.168.200.102/v1/projects/1a5/accounts","addOutputsInputs":"http://192.168.200.102/v1/projects/1a5/addoutputsinputs","amazonec2Configs":"http://192.168.200.102/v1/projects/1a5/amazonec2configs","apiKeys":"http://192.168.200.102/v1/projects/1a5/apikeys","azureConfigs":"http://192.168.200.102/v1/projects/1a5/azureconfigs","azureadconfigs":"http://192.168.200.102/v1/projects/1a5/azureadconfigs","composeProjects":"http://192.168.200.102/v1/projects/1a5/composeprojects","composeServices":"http://192.168.200.102/v1/projects/1a5/composeservices","configItems":"http://192.168.200.102/v1/projects/1a5/configitems","containerExecs":"http://192.168.200.102/v1/projects/1a5/containerexecs","containers":"http://192.168.200.102/v1/projects/1a5/containers","databasechangeloglocks":"http://192.168.200.102/v1/projects/1a5/databasechangeloglocks","databasechangelogs":"http://192.168.200.102/v1/projects/1a5/databasechangelogs","digitaloceanConfigs":"http://192.168.200.102/v1/projects/1a5/digitaloceanconfigs","dnsServices":"http://192.168.200.102/v1/projects/1a5/dnsservices","dockerBuilds":"http://192.168.200.102/v1/projects/1a5/dockerbuilds","environmentUpgrades":"http://192.168.200.102/v1/projects/1a5/environmentupgrades","extensionPoints":"http://192.168.200.102/v1/projects/1a5/extensionpoints","externalDnsEvents":"http://192.168.200.102/v1/projects/1a5/externaldnsevents","externalHandlerExternalHandlerProcessMaps":"http://192.168.200.102/v1/projects/1a5/externalhandlerexternalhandlerprocessmaps","externalHandlerProcesses":"http://192.168.200.102/v1/projects/1a5/externalhandlerprocesses","externalHandlers":"http://192.168.200.102/v1/projects/1a5/externalhandlers","externalHostEvents":"http://192.168.200.102/v1/projects/1a5/externalhostevents","externalServiceEv

ents":"http://192.168.200.102/v1/projects/1a5/externalserviceevents","externalServices":"http://192.168.200.102/v1/projects/1a5/externalservices","externalStoragePoolEvents":"http://192.168.200.102/v1/projects/1a5/externalstoragepoolevents","externalVolumeEvents":"http://192.168.200.102/v1/projects/1a5/externalvolumeevents","githubconfigs":"http://192.168.200.102/v1/projects/1a5/githubconfigs","haConfigInputs":"http://192.168.200.102/v1/projects/1a5/haconfiginputs","haConfigs":"http://192.168.200.102/v1/projects/1a5/haconfigs","hostAccesses":"http://192.168.200.102/v1/projects/1a5/hostaccesses","hostApiProxyTokens":"http://192.168.200.102/v1/projects/1a5/hostapiproxytokens","identities":"http://192.168.200.102/v1/projects/1a5/identities","kubernetesServices":"http://192.168.200.102/v1/projects/1a5/kubernetesservices","ldapconfigs":"http://192.168.200.102/v1/projects/1a5/ldapconfigs","loadBalancerConfigs":"http://192.168.200.102/v1/projects/1a5/loadbalancerconfigs","loadBalancerServices":"http://192.168.200.102/v1/projects/1a5/loadbalancerservices","localAuthConfigs":"http://192.168.200.102/v1/projects/1a5/localauthconfigs","machineDrivers":"http://192.168.200.102/v1/projects/1a5/machinedrivers","machines":"http://192.168.200.102/v1/projects/1a5/machines","openldapconfigs":"http://192.168.200.102/v1/projects/1a5/openldapconfigs","packetConfigs":"http://192.168.200.102/v1/projects/1a5/packetconfigs","passwords":"http://192.168.200.102/v1/projects/1a5/passwords","processDefinitions":"http://192.168.200.102/v1/projects/1a5/processdefinitions","processExecutions":"http://192.168.200.102/v1/projects/1a5/processexecutions","processInstances":"http://192.168.200.102/v1/projects/1a5/processinstances","projects":"http://192.168.200.102/v1/projects/1a5/projects","pullTasks":"http://192.168.200.102/v1/projects/1a5/pulltasks","register":"http://192.168.200.102/v1/projects/1a5/register","registrationTokens":"http://192.168.200.102/v1/projects/1a5/registrationtokens","registries":"http://192.168.200.102/v1/projects/1a5/registries","registryCredentials":"http://192.168.200.102/v1/projects/1a5/registrycredentials","resourceDefinitions":"http://192.168.200.102/v1/projects/1a5/resourcedefinitions","scalePolicys":"http://192.168.200.102/v1/projects/1a5/scalepolicys","schemas":"http://192.168.200.102/v1/projects/1a5/schemas","serviceProxies":"http://192.168.200.102/v1/projects/1a5/serviceproxies","settings":"http://192.168.200.102/v1/projects/1a5/settings","snapshotBackupInputs":"http://192.168.200.102/v1/projects/1a5/snapshotbackupinputs","statsAccesses":"http://192.168.200.102/v1/projects/1a5/statsaccesses","taskInstances":"http://192.168.200.102/v1/projects/1a5/taskinstances","tasks":"http://192.168.200.102/v1/projects/1a5/tasks","typeDocumentations":"http://192.168.200.102/v1/projects/1a5/typedocumentations","virtualMachines":"http://192.168.200.102/v1/projects/1a5/virtualmachines","hostStats":"http://192.168.200.102/v1/projects/1a5/projects/1a5/hoststats"},"actions":{"update":"http://192.168.200.102/v1/projects/1a5/?action=update","deactivate":"http://192.168.200.102/v1/projects/1a5/?action=deactivate","setmembers":"http://192.168.200.102/v1/projects/1a5/?action=setmembers","delete":"http://192.168.200.102/v1/projects/1a5/?action=delete"},"name":"Default","state":"active","allowSystemRole":false,"created":"2017-04-20T05:46:19Z","createdTS":1492667179000,"data":{"fields":{"servicesPortRange":{"startPort":49153,"endPort":65535},"defaultNetworkId":5}},"description":null,"kind":"project","kubernetes":false,"members":null,"mesos":false,"publicDns":false,"removeTime":null,"removed":null,"servicesPortRange":{"endPort":65535,"startPort":49153},"swarm":false,"transitioning":"no","transitioningMessage":null,"transitioningProgress":null,"uuid":"adminProject","virtualMachine":false}

查询命令的网址中除 ip 地址其余完全正确得 1 分
查询结果中除 ip 地址外其余内容一致得 1 分

2. 应用部署（3 分）

通过“应用商店”部署 gogs，服务部署完成后创建名称为 xiandian 的 git 仓库，在 Client 节点使用 git 命令将仓库克隆到本地。以文本形式提交操作命令及结果到答题框。

```
# git clone http://192.168.200.103:10080/xiandian/xiandian.git
Cloning into 'xiandian'...
warning: You appear to have cloned an empty repository.
```

使用 git clone 命令得 1 分
命令中网址除 ip 地址其余内容一致得 1 分
操作结果完全相同得 1 分

第二部分：云计算平台运维管理（35 分）

任务一、IaaS 云平台运维（25 分）

使用 PC 上镜像文件 Xiandian-IaaS-All.qcow2 创建 glance 镜像 iaas-all，格式为 qcow2；在 PC 中解压文件 iaas-error-351.zip，解压密码为 iaaspaicuo，使用解压出来的镜像文件 iaas-error-351.qcow2 创建 glance 镜像 iaas-error-351；在 PC 中解压文件 iaas-error-363.zip，解压密码为 erroriaas，使用解压出来的镜像文件 iaas-error-363.qcow2 创建 glance 镜像 iaas-error-363。按以下初始配置在云平台中创建云主机。

云主机：

（1）名称：iaas。

（2）镜像：iaas-all。

（3）云主机类型：4CPU、8G 内存、100G 硬盘。

（4）网络：网络 1：int-net1，绑定浮动 IP；网络 2：int-net2。

1. keystone 管理（3 分）

在 iaas 云主机中创建用户 testuser，密码为 password。将 testuser 用户分配给 admin 项目，赋予用户 user 的权限。以管理员身份将 testuser 用户的密码修改为 000000。以文本形式提交以上所有操作命令及对应结果到答题框。

```
[root@xiandian ~]# openstack user create --domain xiandian --password password testuser
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | 3ac89594c8e944a9b5bb567fca4e75aa |
| enabled   | True                             |
| id        | 053f1148aefb4d709cf08f51b90a58fd |
| name      | testuser                         |
+-----------+----------------------------------+
[root@xiandian ~]# openstack role add --project admin --user testuser user
[root@xiandian ~]#  openstack user set --password 000000 testuser
```

创建用户命令正确，结果一致得 1 分
权限增加命令正确得 1 分
修改密码命令正确得 1 分

2. ceilometer 管理（4 分）

检查 iaas 云主机中 mongodb 和 ceilometer 服务，确保 ceilometer 组件正常运行，按以下要求使用 ceilometer 命令进行操作。

（1）按提供参数的顺序创建一个基于计算统计的告警，其中名字为：cpu_hi；测量值的名称为：cpu_util；阈值为：70.0；对比的方式为：大于；统计的方式为：平均值；周期为：600s；统计的次数为：3；转为告警状态时提交的 URL 为：'log://'；关键字：resource_id=INSTANCE_ID。

（2）修改告警的告警状态为不生效。

（3）查询告警历史的改变信息。

（4）查询事件列表信息。

按顺序将以上命令和执行结果以文本形式提交到答题框。

```
[root@xiandian ~]# ceilometer alarm-threshold-create --name cpu_hi    --meter-name cpu_util
--threshold 70.0 --comparison-operator gt    --statistic avg --period 600 --evaluation-periods 3
--alarm-action 'log://' --query resource_id=INSTANCE_ID
+--------------------------+-------------------------------------------------------------+
| Property                 | Value
|
+--------------------------+-------------------------------------------------------------+
| alarm_actions            | ["log://"]
|
| alarm_id                 | 44f0d35f-494d-4927-8517-d84628361453
|
| comparison_operator      | gt
|
| description              | Alarm when cpu_util is gt a avg of 70.0 over 600 seconds |
| enabled                  | True
|
| evaluation_periods       | 3
|
| exclude_outliers         | False
|
| insufficient_data_actions | []
|
| meter_name               | cpu_util
|
| name                     | cpu_hi
|
| ok_actions               | []
|
| period                   | 600
|
| project_id               | 0ab2dbde4f754b699e22461426cd0774
|
```

```
| query                          | resource_id == INSTANCE_ID
|
| repeat_actions                 | False
|
| severity                       | low
|
| state                          | insufficient data
|
| statistic                      | avg
|
| threshold                      | 70.0
|
| type                           | threshold
|
| user_id                        | 53a1cf0ad2924532aa4b7b0750dec282
|
+--------------------------+-------------------------------------------------------+
[root@xiandian ~]# ceilometer alarm-update --enabled False
44f0d35f-494d-4927-8517-d84628361453
+--------------------------+-------------------------------------------------------+
| Property                       | Value
|
+--------------------------+-------------------------------------------------------+
| alarm_actions                  | ["log://"]
|
| alarm_id                       | 44f0d35f-494d-4927-8517-d84628361453
|
| comparison_operator            | gt
|
| description                    | Alarm when cpu_util is gt a avg of 70.0 over 600 seconds |
| enabled                        | False
|
| evaluation_periods             | 3
|
| exclude_outliers               | False
|
| insufficient_data_actions | []
|
| meter_name                     | cpu_util
|
| name                           | cpu_hi
|
| ok_actions                     | []
|
```

```
| period                          | 600
|
| project_id                      | 0ab2dbde4f754b699e22461426cd0774
|
| query                           | resource_id == INSTANCE_ID
|
| repeat_actions                  | False
|
| severity                        | low
|
| state                           | insufficient data
|
| statistic                       | avg
|
| threshold                       | 70.0
|
| type                            | threshold
|
| user_id                         | 53a1cf0ad2924532aa4b7b0750dec282
|
+-------------------------+---------------------------------------------------------+
[root@xiandian ~]# ceilometer alarm-history 44f0d35f-494d-4927-8517-d84628361453
+-------------+--------------------------+-------------------------------------------------------------------------
----+
| Type              | Timestamp                          | Detail
|
+-------------+--------------------------+-------------------------------------------------------------------------
----+
| rule change | 2017-06-01T05:38:04.267943 | enabled: False
|
| creation        | 2017-06-01T05:37:34.111994 | name: cpu_high
|
|                   |                                    | description: Alarm when cpu_util is gt a
avg of 70.0 over 600 seconds |
|                   |                                    | type: threshold
|
|                   |                                    | rule: avg(cpu_util) > 70.0 during 3 x
600s                                    |
|                   |                                    | severity: low
|
|                   |                                    | time_constraints: None
|
+-------------+--------------------------+-------------------------------------------------------------------------
----+
```

[root@xiandian ~]# ceilometer event-list +------------+------------+-----------+--------+ | Message ID | Event Type | Generated | Traits | +------------+------------+-----------+--------+ +------------+------------+-----------+--------+
创建告警的命令及结果正确得 1 分 修改告警状态命令及结果正确得 1 分 查询告警历史命令正确，结果中条目一致得 1 分 查询事件列表命令正确，结果格式正确得 1 分

3. 云平台排错（6 分）

使用镜像 iaas-error-351 重建云主机 iaas，使用 glance 命令查询镜像 checksum 值。重建后该主机内有搭建错误的 OpenStack 平台，错误现象为无法创建云主机。排错后使用 OpenStack 平台创建云主机，主机名为 errorhost，云主机类型为 m1.tiny，镜像为 cirros，网络为 sharednet1。以文本形式按顺序将镜像 checksum 值、错误的内容、排错后的内容及通过 nova 命令查询 errorhost 的详细信息提交到答题框。

0c5e74929d91b7eb421b6d75dda7532b bind=10.0.0.1 provider = keystone [root@xiandian ~]# openstack role list --user glance --project service +----------------------------------+------+---------+--------+ | ID | Name | Project | User | +----------------------------------+------+---------+--------+ | 384c446de0dc4c80bc4ef187c555dacd | user | service | glance | +----------------------------------+------+---------+--------+ [root@xiandian ~]# nova service-list +----+-----------------+----------+----------+----------+-------+----------------------------+----------- -----+ | Id | Binary | Host | Zone | Status | State | Updated_at | Disabled Reason | +----+-----------------+----------+----------+----------+-------+----------------------------+-----------

```
-----+

| 1  | nova-conductor    | xiandian | internal | enabled  | up     |
2017-06-01T05:59:19.000000 | -                       |

| 3  | nova-scheduler    | xiandian | internal | enabled  | up     |
2017-06-01T05:59:11.000000 | -                       |

| 4  | nova-consoleauth | xiandian | internal | enabled  | up     |
2017-06-01T05:59:19.000000 | -                       |

| 7  | nova-compute      | xiandian | nova     | disabled | up     |
2017-06-01T05:59:19.000000 | -                       |

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

#bind=10.0.0.1
provider = fernet

[root@xiandian ~]# openstack role list --user glance --project service

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

| ID                               | Name  | Project | User   |

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

| 384c446de0dc4c80bc4ef187c555dacd | user  | service | glance |

| 710e4f3907264eb3b65e97e27990f30b | admin | service | glance |

+----------------------------------+-------+---------+--------+
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY '000000';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY '000000';

[root@xiandian ~]# nova service-enable xiandian nova-compute

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

| Host     | Binary      | Status  |
```

```
+----------+-------------+---------+

| xiandian | nova-compute | enabled |

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

[root@xiandian ~]# nova service-list

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

| Id | Binary                | Host          | Zone          | Status    | State | Updated_at
| Disabled Reason |

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

| 1    | nova-conductor      | xiandian | internal | enabled | up        |
2017-06-01T05:59:39.000000 | -                          |

| 3    | nova-scheduler      | xiandian | internal | enabled | up        | 2017-06-01T05:59:41.000000
| -                          |

| 4    | nova-consoleauth | xiandian | internal | enabled | up        | 2017-06-01T05:59:39.000000
| -                          |

| 7    | nova-compute          | xiandian | nova          | enabled | up        |
2017-06-01T05:59:42.000000 | -                          |

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

[root@xiandian ~]# nova show errorhost

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

| Property                                 | Value
|

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

| OS-DCF:diskConfig                        | AUTO
```

```
|
| OS-EXT-AZ:availability_zone               | nova
|
| OS-EXT-SRV-ATTR:host                      | xiandian
|
| OS-EXT-SRV-ATTR:hostname                  | errorhost
|
| OS-EXT-SRV-ATTR:hypervisor_hostname   | xiandian
|
| OS-EXT-SRV-ATTR:instance_name             | instance-00000001
|
| OS-EXT-SRV-ATTR:kernel_id                 |
|
| OS-EXT-SRV-ATTR:launch_index              | 0
|
| OS-EXT-SRV-ATTR:ramdisk_id                |
|
| OS-EXT-SRV-ATTR:reservation_id            | r-eim5pq9q
|
| OS-EXT-SRV-ATTR:root_device_name          | /dev/vda
|
| OS-EXT-SRV-ATTR:user_data                 | -
|
| OS-EXT-STS:power_state                    | 1
|
| OS-EXT-STS:task_state                     | -
|
```

```
| OS-EXT-STS:vm_state                  | active
|
| OS-SRV-USG:launched_at               | 2017-06-01T06:10:12.000000
|
| OS-SRV-USG:terminated_at             | -
|
| accessIPv4                           |
|
| accessIPv6                           |
|
| config_drive                         |
|
| created                              | 2017-06-01T06:09:25Z
|
| description                          | errorhost
|
| flavor                               | m1.tiny (1)
|
| hostId                               |
991e57844e869d97b120c16afc7f4820ab6361ed1904375f832453e4 |
| host_status                          | UP
|
| id                                   | 0f913798-4fca-4400-a86f-0601972e3ac8
|
| image                                | cirros
(a70696f7-eaa5-4a14-bfee-0e0d62775943)          |
| key_name                             | -
```

```
|

| locked                                | False
|

| metadata                              | {}
|

| name                                  | errorhost
|

| os-extended-volumes:volumes_attached | []
|

| progress                              | 0
|

| security_groups                       | default
|

| sharednet1 network                    | 10.0.1.3
|

| status                                | ACTIVE
|

| tenant_id                             | 0ab2dbde4f754b699e22461426cd0774
|

| updated                               | 2017-06-01T06:10:13Z
|

| user_id                               | 53a1cf0ad2924532aa4b7b0750dec282
|

+--------------------------------------+----------------------------------------------------------+
```

checksum 值正确得 0.5 分
找到数据库配置文件错误并修改正确得 1 分
找到 keystone 配置文件错误并修改正确得 1 分
找到 glance 权限错误配置并修改正确得 1 分
配置 nova 数据库访问权限得 1 分

找到 nova 被禁用服务并修改正确得 1 分
查看新建云主机详细信息且参数一致得 0.5 分

4. 云存储排错（7 分）

使用镜像 iaas-error-363 重建云主机 iaas，使用 glance 命令查询镜像 checksum 值。重建后该主机内有搭建错误的 OpenStack Swift 云存储平台，错误现象为云存储容器内容无法查询。排错后查询云存储账号 demo 中的容器列表信息。以文本形式按顺序将镜像 checksum 值、错误的内容、排错后的内容及查询到的容器列表信息提交到答题框。

db08f10ebdcb969d1bd49bfa37817759
/var/lib/mysql/mysql.sock

Listen 35375

<VirtualHost *:35375>

[root@xiandian ~]# ls -la /etc/swift/

total 1104

drwxr-xr-x. 7 root root   4096 May 25 08:26 .

drwxr-xr-x. 110 root root   8192 Jun 1 06:13 ..

-rw-r--r--. 1 root root 1050252 Feb 23 05:12 account.builder

-rw-r--r--. 1 root root   710 Feb 23 05:13 account.ring.gz

drwxr-xr-x. 2 root root   6 May 17 2016 account-server

-rw-r-----. 1 root root   431 Feb 23 05:13 account-server.conf

drwxr-xr-x. 2 root root   4096 Feb 23 05:12 backups

-rw-r--r--. 1 root root   4714 Feb 23 05:12 container.builder

-rw-r-----. 1 root root   1415 May 17 2016 container-reconciler.conf

-rw-r--r--. 1 root root   192 Feb 23 05:13 container.ring.gz

drwxr-xr-x. 2 root root   6 May 17 2016 container-server

-rw-r-----. 1 root root   465 May 25 08:26 container-server.conf

-rw-r--r--. 1 root root   4714 Feb 23 05:12 object.builder

-rw-r-----. 1 root root 291 May 17 2016 object-expirer.conf

-rw-r--r--. 1 root root 188 Feb 23 05:13 object.ring.gz

drwxr-xr-x. 2 root root 6 May 17 2016 object-server

-rw-r-----. 1 root root 477 Feb 23 05:13 object-server.conf

drwxr-xr-x. 2 root root 6 May 17 2016 proxy-server

-rw-r-----. 1 root root 2254 Feb 23 05:48 proxy-server.conf

-rw-r-----. 1 root root 175 Feb 23 05:13 swift.conf

user=container

rm /var/lib/mysql/mysql.sock

Listen 35357

<VirtualHost *:35357>

openstack endpoint create --region RegionOne identity admin http://xiandian:35357/v3

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

| Field | Value |

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

| enabled | True |

| id | db737247d1c144c4a9dc29dc0ade1a43 |

| interface | admin |

| region | RegionOne |

| region_id | RegionOne |

| service_id | 2b73f27e09ef4f0fa717153f88f8ac8c |

| service_name | keystone |

| service_type | identity |

| url | http://xiandian:35357/v3 |

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

chown -R root:swift /etc/swift

user = swift

[root@xiandian ~]# source /etc/keystone/demo-openrc.sh

[root@xiandian ~]# swift list

error363

checksum 值正确得 0.5 分
找到数据库错误点并修改正确得 1 分
找到 http 配置文件中错误配置并修改正确得 1 分
找到 keystone 中端点缺失项并增加正确得 1 分
查看 swift 文件夹读写权限并修改正确得 1 分
找到 container 配置错误并修改正确得 1 分
查看到 demo 用户的容器为 error363 得 0.5 分

5. SDN 管理（5 分）

在控制节点安装配置 JDK 环境和 Maven 环境，然后完成 OpenDaylight 的安装。安装完成后，查询 JDK 的版本信息、Maven 的版本信息并使用 curl 命令访问网页 http://192.168.100.10:8181/index.html。以文本形式提交查询命令及结果到答题框。

```
[root@controller ~]# java -version
java version "1.7.0_71"
Java(TM) SE Runtime Environment (build 1.7.0_71-b14)
Java HotSpot(TM) 64-Bit Server VM (build 24.71-b01, mixed mode)\
[root@controller ~]# source /etc/profile
[root@controller ~]# mvn -v
Apache Maven 3.0.4 (r1232337; 2012-01-17 16:44:56+0800)
Maven home: /usr/local/apache-maven-3.0.4
Java version: 1.7.0_71, vendor: Oracle Corporation
Java home: /usr/local/jdk1.7.0_71/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-327.el7.x86_64", arch: "amd64", family: "unix"
[root@controller ~]# curl http://192.168.100.10:8181/index.html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>OpenDaylight Dlux</title>

    <meta name="description" content="overview &amp;amp; stats" />
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script type="text/javascript">
var module = ['angular','ocLazyLoad','angular-ui-router','angular-translate', 'angular-sanitize',
'angular-translate-loader-static-files', 'angular-translate-loader-partial', 'angular-css-injector'];
var deps =
['common/config/env.module','app/core/core.module','common/login/login.module','common/
authentification/auth.module','common/navigation/navigation.module','common/topbar/topbar
.module','common/general/common.general.module','app/topology/topology.module','commo
n/layout/layout.module'];
var e = ['oc.lazyLoad', 'ui.router', 'pascalprecht.translate', 'ngSanitize', 'angular.css.injector',
'app','app.core','app.common.login','app.common.auth','app.common.nav','app.common.topbar
','app.common.general','app.topology','app.common.layout'];
        // global variables

    </script>

    <!-- HTML5 shim and Respond.js IE8 support of HTML5 elements and media queries
-->

    <!--[if lt IE 9]>
    <script src="assets/js/html5shiv.js"></script>
    <script src="assets/js/respond.min.js"></script>
    <![endif]-->

    <!-- compiled CSS -->
    <link rel="stylesheet" type="text/css" href="vendor/ng-grid/ng-grid.min.css" />
    <link rel="stylesheet" type="text/css"
href="vendor/select2-bootstrap-css/select2-bootstrap.css" />
    <link rel="stylesheet" type="text/css" href="vendor/footable/css/footable.core.min.css"
/>
    <link rel="stylesheet" type="text/css"
href="vendor/footable/css/footable.standalone.min.css" />
    <link rel="stylesheet" type="text/css" href="vendor/vis/dist/vis.min.css" />
    <link rel="stylesheet" type="text/css" href="vendor/ng-slider/dist/css/ng-slider.min.css"
/>
    <link rel="stylesheet" type="text/css" href="assets/opendaylight-dlux-0.2.0.css" />
    <link rel="stylesheet" href="assets/css/sb-admin.css" />

    <script type="text/javascript" data-main="src/main.js"
src="vendor/requirejs/require.js"></script>

    <link rel="stylesheet" href="assets/css/font-awesome.min.css" />
    <!-- the font-awesome is different from the 'official' one -->
```

```
    <!-- application CSS -->
  </head>

  <body>
  <div ui-view="mainContent"></div>
  </body>
</html>
```

查询 JDK 版本信息命令和结果正确得 1.5 分
查询 Maven 版本信息命令和结果正确得 1.5 分
查询 OpenDaylight 网址命令完全正确得 1 分
查询网页内容完全一致得 1 分

任务二、PaaS 云平台运维（10 分）

1. 网络管理（2 分）

（1）在 Registry 节点创建 xd_br 网桥：192.168.2.1/24，创建完成后启动 xd_br。

（2）使用 ubuntu:14.04.3 镜像运行一个无网络环境的容器。为容器创建独立的网络命名空间 ns；创建虚拟网络接口设备 A，为 A 创建一个映射端设备 B；将设备 A 接入到网桥 xd_br 中。完成后启动设备 A，将 B 设备放入网络空间 ns 中。

（3）在容器中将设备 B 重命名为网络设备名 eth0，并分配 MAC 地址为 1A:2B:3C:4D:5E:6F；根据 xd_br 网桥的地址将该网段的最后一位 IP 地址分配给 eth0，设置路由到 xd_br。

完成后查询 xd_br 网桥详细信息、查询容器 eth0 网卡和路由信息，以文本形式提交查询命令及结果到答题框。

```
# ifconfig xd_br
xd_br: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.2.1  netmask 255.255.255.0  broadcast 0.0.0.0
        ether 7e:cb:e4:ca:78:54  txqueuelen 0  (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
# docker exec 1319914566c4 ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 1a:2b:3c:4d:5e:6f
          inet addr:192.168.2.254  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::182b:3cff:fe4d:5e6f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:648 (648.0 B)  TX bytes:648 (648.0 B)
# docker exec 1319914566c4 ip route
```

default via 192.168.2.1 dev eth0 192.168.2.0/24 dev eth0 proto kernel scope link src 192.168.2.254
查询网桥详细信息的 1 分 查询容器 eth0 网卡信息的 0。5 分 查询容器路由信息得 0.5 分

2. 镜像构建（2 分）

使用 supermin5 命令构建 centos7 系统的 docker 镜像，镜像名称为 centos-7，镜像预装 yum、net-tools、initscripts 和 vi 软件，构建完成后提交到镜像仓库，提交完成后运行 centos-7。

以文本形式提交容器的操作系统版本信息、docker 容器列表信息到答题框。

bash-4.2# cat /etc/redhat-release Derived from Red Hat Enterprise Linux 7.1 (Source) # docker ps -a CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES a39b8f32b18e 192.168.200.12:5000/centos-7 "/bin/bash" About a minute ago Up About a minute romantic_rosalind
查询容器操作系统版本信息内容正确得 1 分 查询容器列表信息正确得 1 分

3. Dockerfile 文件编写（4 分）

使用 git 相关命令将附件中提供的 test.php 文件上传到 gogs 的 xiandian 仓库中；在 Client 节点/root/xd_web 路径下编写 Dockerfile 文件，满足以下 12 个要求。

（1）使用第 2 小题创建的 centos-7 镜像作为基本镜像，删除镜像的原始 yum 源，使用当前系统的 yum 源文件作为镜像的 yum 源。

（2）容器安装 mariadb 数据库服务、java 运行环境、unzip 解压工具、httpd web 服务器、php、php-mysql 和 git。

（3）将提供的 build_table.sh、start.sh 文件添加到系统的/root 路径下。

（4）添加 apache-tomcat.zip 到/root 路径，添加 index.html 为 tomcat 的主页，并创建/var/log/httpd，/var/www/html 目录。

（5）利用 git 克隆 gogs 仓库内的 test.php 文件并移动到/var/www/html 路径下，完成后删除原克隆下来的文件夹。

（6）使用 mysql 用户初始化数据库，设置环境变量中 MYSQL_USER 为 xiandian、MYSQL_PASS 为 000000、MYSQL_ADDR 为 localhost，要求数据库支持 UTF-8，运行并修改提供的 build_table.sh 和 start.sh 文件，暴露端口 3306。

（7）解压 apache-tomcat.zip 到/root 路径

（8）添加 build_table.sh 执行权限，完成后执行该程序。

（9）添加 start.sh 文件执行权限。

（10）添加/root/apache-tomcat-6/bin 路径内的文件执行权限。

（11）暴露端口 80、3306、8080。

（12）容器运行时要求自动执行 start.sh 脚本文件。

编写完成后按顺序将本题中使用 git 命令上传的具体操作命令、执行结果及编写的 Dockerfile 文件内容以文本形式提交到答题框。

```
# git clone http://10.0.6.81:10080/xiandian/xiandian.git
Cloning into 'xiandian'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
# git add .
# git commit -am "add changes"
[master 1cbbae2] add changes
 1 file changed, 31 insertions(+)
 create mode 100755 test.php
# git push
warning: push.default is unset; its implicit value is changing in
Git 2.0 from 'matching' to 'simple'. To squelch this message
and maintain the current behavior after the default changes, use:
   git config --global push.default matching
To squelch this message and adopt the new behavior now, use:
   git config --global push.default simple
See 'git help config' and search for 'push.default' for further information.
(the 'simple' mode was introduced in Git 1.7.11. Use the similar mode
'current' instead of 'simple' if you sometimes use older versions of Git)
Username for 'http://10.0.6.81:10080': xiandian
Password for 'http://xiandian@10.0.6.81:10080':
Counting objects: 4, done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 616 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To http://10.0.6.81:10080/xiandian/xiandian.git
    8aad82a..1cbbae2    master -> master
FROM 10.0.6.83:5000/centos-7
MAINTAINER Xiandian
RUN rm -fv /etc/yum.repos.d/*
ADD local.repo /etc/yum.repos.d/
RUN yum install -y mariadb-server java unzip httpd php php-mysql    git
ADD build_table.sh /root/build_table.sh
ADD start.sh /root/start.sh
ADD apache-tomcat.zip    /root/apache-tomcat.zip
RUN mkdir -p /var/log/httpd
RUN mkdir -p /var/www/html
ADD index.html /root/apache-tomcat-6/webapps/ROOT/index.html
RUN git clone http://10.0.6.81:10080/xiandian/xiandian.git
RUN mv xiandian/test.php /var/www/html/test.php
RUN rm -rf xiandian
```

```
RUN mysql_install_db --user=mysql
ENV LC_ALL en_US.UTF-8
ENV MYSQL_USER xiandian
ENV MYSQL_PASS 000000
ENV MYSQL_ADDR localhost
ENV TERM linux
RUN unzip /root/apache-tomcat.zip -d /root/
RUN chmod +x /root/build_table.sh
RUN /root/build_table.sh
RUN chmod +x /root/start.sh
RUN chmod u+x /root/apache-tomcat-6/bin/*
EXPOSE 80
EXPOSE 3306
EXPOSE 8080
ENTRYPOINT ["/root/start.sh"]
CMD ""
```

git 操作命令正确得 1 分
git 操作结果正确得 1 分
dockerfile 文件内容一致得 2 分

4. 持续集成环境构建（2 分）

安装 jenkins 并根据 gogs 的相关设置，配置 jenkins 使用的 git 代码库，创建名称为 xiandian_project 的 jenkins 项目，配置项目源码管理编写触发器，使用 ssh 命令构建持续集成 DevOps 环境。要求如下：

（1）检测主机是否存在名称为 xd_web 的容器，如果存在则删除。

（2）检测是否存在 xd_web 的镜像，如果存在则删除。

（3）构建 Client 节点/root/xd_web 路径下的 Dockerfile 文件，镜像名称为 xd_web。

（4）运行该容器，端口监听规则为：8585 端口监听内部 80 端口、8586 端口监听内部 8080 端口，名称为 xd_web。

gogs 的 xiandian 仓库中 test.php 代码存在错误，将“con1”连接数据库的代码部分修改正确，完成后在 jenkins 项目中点击“立即构建”该项目，构建完成后，按顺序将下列信息以文本形式提交到答题框。

a. 配置构建环境相关命令。

b. 构建完成后的 console output（控制台）中内容。

c. 使用 curl 命令获取 8585 端口的 test.php 网页内容。

```
docker ps -a |grep -w "xd_web" |awk '{print $1}' | xargs -i docker rm -f {}
docker images   |grep -w "xd_web"   |awk '{print $3}' |xargs docker rmi -f
docker build -t xd_web   /root/xd_web
docker run -itd -p 8585:80 –p 8586:8080 --name xd_web xd_web
    Started by user anonymous
    Building in workspace /var/jenkins_home/jobs/xiandian_project/workspace
     > git rev-parse --is-inside-work-tree # timeout=10
    Fetching changes from the remote Git repository
     > git config remote.origin.url http://10.0.6.81:10080/xiandian/xiandian.git # timeout=10
```

```
Fetching upstream changes from http://10.0.6.81:10080/xiandian/xiandian.git
 > git --version # timeout=10
 > git fetch --tags --progress http://10.0.6.81:10080/xiandian/xiandian.git
+refs/heads/*:refs/remotes/origin/*
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision ae4718dda876d8896695e924e26da9e8924c4061
(refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f ae4718dda876d8896695e924e26da9e8924c4061
 > git rev-list ae4718dda876d8896695e924e26da9e8924c4061 # timeout=10
[SSH] executing pre build script:

docker ps -a |grep -w "xd_web" |awk '{print $1}' | xargs -i docker rm -f {}
docker images    |grep -w "xd_web"    |awk '{print $3}' |xargs docker rmi -f
docker build -t xd_web    /root/xd_web
docker run -itd -p 8585:80 -p 8586:8080 --name xd_web xd_web
docker: "rmi" requires a minimum of 1 argument.
See '/usr/bin/docker-current rmi --help'.

Usage:          docker rmi [OPTIONS] IMAGE [IMAGE...]

Remove one or more images
Sending build context to Docker daemon 557.1 kB
Sending build context to Docker daemon 1.114 MB
Sending build context to Docker daemon 1.671 MB
Sending build context to Docker daemon 2.228 MB
Sending build context to Docker daemon 2.785 MB
Sending build context to Docker daemon 3.342 MB
Sending build context to Docker daemon 3.899 MB
Sending build context to Docker daemon 4.456 MB
Sending build context to Docker daemon 5.014 MB
Sending build context to Docker daemon 5.033 MB

Step 1 : FROM 10.0.6.83:5000/centos-7
 ---> e40242986ac3
Step 2 : MAINTAINER Xiandian
 ---> Using cache
 ---> 24b611d226ea
Step 3 : RUN rm -fv /etc/yum.repos.d/*
 ---> Using cache
 ---> d62ed5af44e1
Step 4 : ADD local.repo /etc/yum.repos.d/
 ---> Using cache
```

```
 ---> 44e2c15bd456
Step 5 : RUN yum install -y mariadb-server java unzip httpd php php-mysql    git
 ---> Using cache
 ---> e1421d1304e3
Step 6 : ADD build_table.sh /root/build_table.sh
 ---> Using cache
 ---> c539b88deb60
Step 7 : ADD apache-tomcat.zip /root/apache-tomcat.zip
 ---> Using cache
 ---> 1f792c85f8da
Step 8 : ADD start.sh /root/start.sh
 ---> Using cache
 ---> 8f497885b767
Step 9 : ADD index.html /root/apache-tomcat-6/webapps/ROOT/index.html
 ---> Using cache
 ---> 700801bd9a07
Step 10 : RUN git clone http://10.0.6.81:10080/xiandian/xiandian.git
 ---> Using cache
 ---> 549eafebd047
Step 11 : RUN mv xiandian/test.php /var/www/html/test.php
 ---> Using cache
 ---> 76e05ee5ae91
Step 12 : RUN rm -rf xiandian
 ---> Using cache
 ---> ea53a87b348c
Step 13 : RUN mysql_install_db --user=mysql
 ---> Using cache
 ---> 51b441645ef3
Step 14 : ENV LC_ALL en_US.UTF-8
 ---> Using cache
 ---> 4d7ad227a3ad
Step 15 : ENV MYSQL_USER xiandian
 ---> Using cache
 ---> 1f62e38ee332
Step 16 : ENV MYSQL_PASS 000000
 ---> Using cache
 ---> dec21a93aa83
Step 17 : ENV MYSQL_ADDR localhost
 ---> Using cache
 ---> e05ed410f1c0
Step 18 : ENV TERM linux
 ---> Using cache
 ---> 4de134edb4c1
Step 19 : RUN unzip /root/apache-tomcat.zip -d /root/
```

```
 ---> Using cache
 ---> 82e383127268
Step 20 : RUN mkdir -p /var/log/httpd
 ---> Using cache
 ---> 7c6b2dd45a94
Step 21 : RUN mkdir -p /var/www/html
 ---> Using cache
 ---> dfc6c3b70067
Step 22 : RUN chmod +x /root/build_table.sh
 ---> Using cache
 ---> b05d37584562
Step 23 : RUN chmod +x /root/start.sh
 ---> Using cache
 ---> 7ce2a62cf47a
Step 24 : RUN /root/build_table.sh
 ---> Running in 5c8285c350e1
170527 08:20:48 mysqld_safe Logging to '/var/log/mariadb/mariadb.log'.
170527 08:20:48 mysqld_safe Starting mysqld daemon with databases from
/var/lib/mysql
 ---> e86fbbfe0637
Removing intermediate container 5c8285c350e1
Step 25 : RUN chmod u+x /root/apache-tomcat-6/bin/*
 ---> Running in bc880b02c041
 ---> 166bdaf87eac
Removing intermediate container bc880b02c041
Step 26 : EXPOSE 80
 ---> Running in d698c1462730
 ---> fa755aaec9c9
Removing intermediate container d698c1462730
Step 27 : EXPOSE 3306
 ---> Running in a495dd950295
 ---> 54691b6afbba
Removing intermediate container a495dd950295
Step 28 : EXPOSE 8080
 ---> Running in 6f22409ea253
 ---> 74031df17899
Removing intermediate container 6f22409ea253
Step 29 : ENTRYPOINT /root/start.sh
 ---> Running in 0862596331bf
 ---> 12967323872b
Removing intermediate container 0862596331bf
Step 30 : CMD ""
 ---> Running in 8769b7fb7108
 ---> d597843933a5
```

Removing intermediate container 8769b7fb7108 Successfully built d597843933a5 31fe3ca6a469636211be9faf65aee9102ab6a9e7e43259185b0b4b9622eafb6c [SSH] exit-status: 0 [SSH] executing post build script: Finished: SUCCESS # curl 10.0.6.81:8585/test.php <table border='1'> <tr> <th>NAME</th> <th>YEARS</th> /tr><tr><td>xiandian</td></tr><tr><td>xiandian</td></tr><tr><td>xiandian</td></tr><tr><td>xiandian_new</td></tr></table><table border='1'> <tr> <th>NAME</th> <th>YEARS</th> </tr><tr><td>2014</td></tr><tr><td>2015</td></tr><tr><td>2016</td></tr><tr><td>2017</td></tr></table>
构建环境命令正确得 0.5 分 控制台内容一致无错误得 0.5 分 获取的网页内容完全一致得 1 分

第三部分、大数据平台（15 分）

任务一、大数据平台搭建（5 分）

大数据平台的搭建采用分布式部署，部署在云平台的两台云主机上，云主机 1 部署大数据平台 master 节点，云主机 2 部署大数据平台 slaver 节点。

云主机 1：

（1）名称：master。

（2）镜像文件：hadoop_master_centos7_x86_xiandian_images-v0.4.qcow2。

（3）云主机类型：2CPU、6G 内存、100G 硬盘。

（4）网络：网络 1：int-net1，绑定浮动 IP。

云主机 2：

（1）名称：slaver。

（2）镜像文件：hadoop_slaver1_centos7_x86_xiandian_images-v0.4.qcow2。

（3）云主机类型：2CPU、6G 内存、100G 硬盘。

（4）网络：网络 1：int-net1，绑定浮动 IP。

云主机创建成功后，检查 master 节点的主机名 master，slaver 节点的主机名 slaver。修改 2 个节点的 hosts 文件，配置 IP 地址与主机名之间的映射关系。检查 master 节点安装的 ntp 时钟服务是否启动，并同步 master 节点时钟至 slaver 节点。

1. ambari 服务检查（2 分）

检查 master 节点 ambari-server 的运行状态，如未启动，则启动 ambari-server 服务。使用 curl 命令在 Linux Shell 中查询 http://master:8080，以文本形式提交查询结果到答题框。

```
[root@master ~]# curl http://master:8080
<!--
* Licensed to the Apache Software Foundation (ASF) under one
* or more contributor license agreements.    See the NOTICE file
* distributed with this work for additional information
* regarding copyright ownership.    The ASF licenses this file
* to you under the Apache License, Version 2.0 (the
* "License"); you may not use this file except in compliance
* with the License.    You may obtain a copy of the License at
*
*         http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
-->


<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="utf-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet" href="stylesheets/vendor.css">
   <link rel="stylesheet" href="stylesheets/app.css">
   <script src="javascripts/vendor.js"></script>
   <script src="javascripts/app.js"></script>
   <script>
         $(document).ready(function() {
               require('initialize');
               // make favicon work in firefox
               $('link[type*=icon]').detach().appendTo('head');
               $('#loading').remove();
         });
```

```
  </script>
  <title>先电大数据平台</title>
  <link rel="shortcut icon" href="/img/logo.png" type="image/x-icon">
</head>
<body>
    <div id="loading">...加载中...</div>
    <div id="wrapper">
    <!-- ApplicationView -->
    </div>
    <footer>
        <div class="container">
            <a href="http://edu.1daoyun.com/" target="_blank">© 2015 南京第五十五所技术开发有限公司 版权所有  版本号：V2.0</a><br>
            <a href="/licenses/NOTICE.txt" target="_blank">查看使用的第三方工具/资源，以及各自归属</a>
         </div>
    </footer>
</body>
</html>
```

curl 命令获取的内容完全一致得 2 分

2. ambari 环境检查（3 分）

分别在 master 节点和 slaver 节点的 Linux Shell 中通过 java 进程状态命令查看 Hadoop 集群服务进程信息，以文本形式提交查询结果到答题框。

```
[root@master ~]# source /etc/profile
[root@master ~]# jps
2770 NameNode
7186 HMaster
5203 NodeManager
1860 AmbariServer
4132 JobHistoryServer
4916 ApplicationHistoryServer
7401 HRegionServer
6633 QuorumPeerMain
21707 Jps
5709 RunJar

[root@slaver1 ~]# source /etc/profile
[root@slaver1 ~]# jps
2849 ResourceManager
3330 NodeManager
6034 HRegionServer
2132 DataNode
12356 Jps
```

4487 RunJar 2393 SecondaryNameNode 4889 RunJar
master 节点内容一致得 1.5 分 slaver1 节点内容一致得 1.5 分

任务二、大数据平台运维（10 分）

1. hive 数据仓库查询（2 分）

启动先电大数据平台的 hive 数据仓库，使用 hive 命令来查找出 phy_course_xd.txt 文件中 Software_1403 班级报名选修 volleyball 的成员所有信息，其中 phy_course_xd.txt 文件数据结构如表 3-1 所示，选修科目字段为 opt_cour，班级字段为 class，将以上操作命令（相关数据库命令语言全部使用小写格式）和输出结果以文本形式提交到答题框。

表 3-1 phy_course_xd.txt 数据结构

stname(string)	stID(int)	class(string)	opt_cour(string)

```
hive> create table xd_phy_course(stname string,stID int,class string,opt_cour string)
    > row format delimited
    > fields terminated by '\t'
    > lines terminated by '\n';
OK
Time taken: 3.02 seconds
hive> load data local inpath '/opt/bigdata/Hive/phy_course_xd.txt' into table xd_phy_course;
Loading data to table default.xd_phy_course
Table default.xd_phy_course stats: [numFiles=1, totalSize=91494]
OK
Time taken: 2.097 seconds
hive> select * from xd_phy_course where class='Software_1403' and opt_cour='volleyball';
OK
student409      10120408        Software_1403   volleyball
student411      10120410        Software_1403   volleyball
student413      10120412        Software_1403   volleyball
student419      10120418        Software_1403   volleyball
student421      10120420        Software_1403   volleyball
student422      10120421        Software_1403   volleyball
student424      10120423        Software_1403   volleyball
student432      10120431        Software_1403   volleyball
student438      10120437        Software_1403   volleyball
student447      10120446        Software_1403   volleyball
Time taken: 0.994 seconds, Fetched: 10 row(s)
```

创建表格式及内容正确得 1 分 查找命令一致且结果正确得 1 分

2. hive 数据仓库管理（3 分）

使用 hive 命令将网络日志 weblog_entries.txt 的查询结果作为数据项创建新的 Hive 表。通过创建一张名为 weblog_entries_url_length 的新表来定义新的网络日志数据库的三个字段：url、request_date 和 request_time。此外，在表中定义一个获取 url 字符串长度名为“url_length”的新字段，其中 weblog_entries.txt 的数据结构如表 3-2 所示。完成后查询 weblog_entries_url_length 表文件内容，将以上操作命令（相关数据库命令语言请全部使用小写格式）和后十行输出结果以文本形式提交到答题框。

表 3-2 weblog_entries.txt 数据结构

md5(string)	url(string)	request_date (string)	request_time (string)	ip(string)

```
hive> create table weblog_entries(md5 string,url string,request_date string,request_time
string,ip string)
    > row format delimited
    > fields terminated by '\t'
    > lines terminated by '\n';
OK
Time taken: 0.717 seconds
hive> load data local inpath '/opt/bigdata/Hive/weblog_entries.txt' into table weblog_entries;
Loading data to table default.weblog_entries
Table default.weblog_entries stats: [numFiles=1, totalSize=254130]
OK
Time taken: 0.899 seconds
hive> create table weblog_entries_url_length as
    > select url,request_date,request_time,length(url) as url_lenth
    > from weblog_entries;
Query ID = root_20170503052636_14ccb311-9df3-4fa1-aa6c-4b99f4c646aa
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.

Status: Running (Executing on YARN cluster with App id
application_1493703646578_0029)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING
PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0
0
--------------------------------------------------------------------------------
VERTICES: 01/01  [==========================>>] 100%  ELAPSED TIME:
```

```
7.47 s
----------------------------------------------------------------------
Moving data to: hdfs://master:8020/apps/hive/warehouse/weblog_entries_url_length
Table default.weblog_entries_url_length stats: [numFiles=1, numRows=3000,
totalSize=121379, rawDataSize=118379]
OK
Time taken: 21.038 seconds
hive> select * from weblog_entries_url_length;
/qnrxlxqacgiudbtfggcg.html          2012-05-10          21:29:01              26
/sbbiuot.html      2012-05-10          21:13:47              13
/ofxi.html         2012-05-10          21:12:37              10
/hjmdhaoogwqhp.html         2012-05-10          21:34:20              19
/angjbmea.html     2012-05-10          21:27:00              14
/mmdttqsnjfifkihcvqu.html           2012-05-10          21:33:53              25
/eorxuryjadhkiwsf.html     2012-05-10          21:10:19              22
/e.html 2012-05-10          21:12:05              7
/khvc.html         2012-05-10          21:25:58              10
/c.html 2012-05-10          21:34:28              7
Time taken: 0.129 seconds, Fetched: 3000 row(s)
```

创建表格式及内容正确得 1.5 分
查找命令一致且结果正确得 1.5 分

3. Spark 部署（2 分）

在先电大数据平台中部署 Spark 服务，打开 Linux Shell 启动 spark-shell 终端。启动后，使用 scala 语言加载 Key-Value 数据：
“("A",4),("A",2),("C",3),("A",4),("B",5),("C",3),("A",4) ”，以 Key 为基准进行去重操作，并通过 toDebugString 方法来查看 RDD 的谱系。以文本形式提交操作命令和结果到答题框。

```
scala> val kv1=sc.parallelize(List(("A",4),("A",2),("C",3),("A",4),("B",5),("C",3),("A",4)))
kv1: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[2] at parallelize at
<console>:21
scala> kv1.distinct.collect
17/05/19 23:22:18 INFO SparkContext: Starting job: collect at <console>:24
17/05/19 23:22:18 INFO DAGScheduler: Registering RDD 3 (distinct at <console>:24)
17/05/19 23:22:18 INFO DAGScheduler: Got job 1 (collect at <console>:24) with 2 output
partitions
17/05/19 23:22:18 INFO DAGScheduler: Final stage: ResultStage 3 (collect at <console>:24)
17/05/19 23:22:18 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 2)
17/05/19 23:22:18 INFO DAGScheduler: Missing parents: List(ShuffleMapStage 2)
17/05/19 23:22:18 INFO DAGScheduler: Submitting ShuffleMapStage 2
(MapPartitionsRDD[3] at distinct at <console>:24), which has no missing parents
17/05/19 23:22:18 INFO MemoryStore: Block broadcast_2 stored as values in memory
(estimated size 2.7 KB, free 10.2 KB)
17/05/19 23:22:18 INFO MemoryStore: Block broadcast_2_piece0 stored as bytes in memory
```

```
(estimated size 1621.0 B, free 11.7 KB)
17/05/19 23:22:18 INFO BlockManagerInfo: Added broadcast_2_piece0 in memory on
localhost:45869 (size: 1621.0 B, free: 511.1 MB)
17/05/19 23:22:18 INFO SparkContext: Created broadcast 2 from broadcast at
DAGScheduler.scala:1008
17/05/19 23:22:18 INFO DAGScheduler: Submitting 2 missing tasks from ShuffleMapStage
2 (MapPartitionsRDD[3] at distinct at <console>:24)
17/05/19 23:22:18 INFO TaskSchedulerImpl: Adding task set 2.0 with 2 tasks
17/05/19 23:22:18 INFO TaskSetManager: Starting task 0.0 in stage 2.0 (TID 4, localhost,
partition 0,PROCESS_LOCAL, 2209 bytes)
17/05/19 23:22:18 INFO TaskSetManager: Starting task 1.0 in stage 2.0 (TID 5, localhost,
partition 1,PROCESS_LOCAL, 2224 bytes)
17/05/19 23:22:18 INFO Executor: Running task 0.0 in stage 2.0 (TID 4)
17/05/19 23:22:18 INFO Executor: Finished task 0.0 in stage 2.0 (TID 4). 1159 bytes result
sent to driver
17/05/19 23:22:18 INFO TaskSetManager: Finished task 0.0 in stage 2.0 (TID 4) in 18 ms on
localhost (1/2)
17/05/19 23:22:18 INFO Executor: Running task 1.0 in stage 2.0 (TID 5)
17/05/19 23:22:18 INFO Executor: Finished task 1.0 in stage 2.0 (TID 5). 1159 bytes result
sent to driver
17/05/19 23:22:18 INFO TaskSetManager: Finished task 1.0 in stage 2.0 (TID 5) in 24 ms on
localhost (2/2)
17/05/19 23:22:18 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all
completed, from pool
17/05/19 23:22:18 INFO DAGScheduler: ShuffleMapStage 2 (distinct at <console>:24)
finished in 0.029 s
17/05/19 23:22:18 INFO DAGScheduler: looking for newly runnable stages
17/05/19 23:22:18 INFO DAGScheduler: running: Set()
17/05/19 23:22:18 INFO DAGScheduler: waiting: Set(ResultStage 3)
17/05/19 23:22:18 INFO DAGScheduler: failed: Set()
17/05/19 23:22:18 INFO DAGScheduler: Submitting ResultStage 3 (MapPartitionsRDD[5] at
distinct at <console>:24), which has no missing parents
17/05/19 23:22:18 INFO MemoryStore: Block broadcast_3 stored as values in memory
(estimated size 3.3 KB, free 15.0 KB)
17/05/19 23:22:18 INFO MemoryStore: Block broadcast_3_piece0 stored as bytes in memory
(estimated size 1933.0 B, free 16.9 KB)
17/05/19 23:22:18 INFO BlockManagerInfo: Added broadcast_3_piece0 in memory on
localhost:45869 (size: 1933.0 B, free: 511.1 MB)
17/05/19 23:22:18 INFO SparkContext: Created broadcast 3 from broadcast at
DAGScheduler.scala:1008
17/05/19 23:22:18 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 3
(MapPartitionsRDD[5] at distinct at <console>:24)
17/05/19 23:22:18 INFO TaskSchedulerImpl: Adding task set 3.0 with 2 tasks
17/05/19 23:22:18 INFO TaskSetManager: Starting task 0.0 in stage 3.0 (TID 6, localhost,
```

```
partition 0,NODE_LOCAL, 1894 bytes)
17/05/19 23:22:18 INFO TaskSetManager: Starting task 1.0 in stage 3.0 (TID 7, localhost,
partition 1,NODE_LOCAL, 1894 bytes)
17/05/19 23:22:18 INFO Executor: Running task 0.0 in stage 3.0 (TID 6)
17/05/19 23:22:18 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 2
blocks
17/05/19 23:22:18 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
17/05/19 23:22:18 INFO BlockManagerInfo: Removed broadcast_1_piece0 on
localhost:45869 in memory (size: 1618.0 B, free: 511.1 MB)
17/05/19 23:22:18 INFO Executor: Running task 1.0 in stage 3.0 (TID 7)
17/05/19 23:22:18 INFO Executor: Finished task 0.0 in stage 3.0 (TID 6). 1307 bytes result
sent to driver
17/05/19 23:22:18 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 2
blocks
17/05/19 23:22:18 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
17/05/19 23:22:18 INFO Executor: Finished task 1.0 in stage 3.0 (TID 7). 1347 bytes result
sent to driver
17/05/19 23:22:18 INFO TaskSetManager: Finished task 0.0 in stage 3.0 (TID 6) in 27 ms on
localhost (1/2)
17/05/19 23:22:18 INFO TaskSetManager: Finished task 1.0 in stage 3.0 (TID 7) in 28 ms on
localhost (2/2)
17/05/19 23:22:18 INFO TaskSchedulerImpl: Removed TaskSet 3.0, whose tasks have all
completed, from pool
17/05/19 23:22:18 INFO DAGScheduler: ResultStage 3 (collect at <console>:24) finished in
0.029 s
17/05/19 23:22:18 INFO DAGScheduler: Job 1 finished: collect at <console>:24, took
0.147233 s
res2: Array[(String, Int)] = Array((A,4), (B,5), (C,3), (A,2))
```

使用 scala 语言加载数据得 0.5 分
去重操作得 1 分
通过 toDebugString 方法查看 RDD 谱系得 0.5 分

4. Spark 过滤（3 分）

在 Spark-Shell 中使用 scala 语言加载 search.txt 文件数据，其数据结构如表 3-3 所示。加载完成后按顺序过滤掉不足 6 列的行数据、第四列排名为 2 的数据和第五列点击顺序号为 1 的数据，并进行计数。将以上操作命令和结果信息以文本形式提交到答题框。

表 3-3 search.txt 数据结构

访问时间	用户 ID	查询词	排名	点击顺序号	用户点击的 URL

```
[root@master spark]# hadoop fs -mkdir /data/spark/
[root@master spark]# hadoop fs -put search.txt /data/spark/
scala> val rdd1 = sc.textFile("hdfs://master/data/spark/search.txt")
17/06/06 01:12:51 INFO MemoryStore: Block broadcast_0 stored as values in memory
```

```
(estimated size 316.8 KB, free 316.8 KB)
17/06/06 01:12:51 INFO MemoryStore: Block broadcast_0_piece0 stored as bytes in memory
(estimated size 27.2 KB, free 344.0 KB)
17/06/06 01:12:51 INFO BlockManagerInfo: Added broadcast_0_piece0 in memory on
localhost:57169 (size: 27.2 KB, free: 517.4 MB)
17/06/06 01:12:51 INFO SparkContext: Created broadcast 0 from textFile at <console>:27
rdd1: org.apache.spark.rdd.RDD[String] = hdfs://master/data/spark/search.txt
MapPartitionsRDD[1] at textFile at <console>:27

scala> val rdd2=rdd1.map(_.split("\t")).filter(_.length==6)
rdd2: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[3] at filter at
<console>:29

scala> val rdd3=rdd2.filter(_(3).toInt==2).filter(_(4).toInt==1)
rdd3: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[46] at filter at
<console>:31

scala> rdd3.count()
17/06/06 02:57:53 INFO SparkContext: Starting job: count at <console>:34
17/06/06 02:57:53 INFO DAGScheduler: Got job 18 (count at <console>:34) with 1 output
partitions
17/06/06 02:57:53 INFO DAGScheduler: Final stage: ResultStage 29 (count at <console>:34)
17/06/06 02:57:53 INFO DAGScheduler: Parents of final stage: List()
17/06/06 02:57:53 INFO DAGScheduler: Missing parents: List()
17/06/06 02:57:53 INFO DAGScheduler: Submitting ResultStage 29 (MapPartitionsRDD[46]
at filter at <console>:31), which has no missing parents
17/06/06 02:57:53 INFO MemoryStore: Block broadcast_28 stored as values in memory
(estimated size 3.7 KB, free 2.0 MB)
17/06/06 02:57:53 INFO MemoryStore: Block broadcast_28_piece0 stored as bytes in
memory (estimated size 2039.0 B, free 2.0 MB)
17/06/06 02:57:53 INFO BlockManagerInfo: Added broadcast_28_piece0 in memory on
localhost:57169 (size: 2039.0 B, free: 517.2 MB)
17/06/06 02:57:53 INFO SparkContext: Created broadcast 28 from broadcast at
DAGScheduler.scala:1008
17/06/06 02:57:53 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 29
(MapPartitionsRDD[46] at filter at <console>:31)
17/06/06 02:57:53 INFO TaskSchedulerImpl: Adding task set 29.0 with 1 tasks
17/06/06 02:57:53 INFO TaskSetManager: Starting task 0.0 in stage 29.0 (TID 24, localhost,
partition 0,ANY, 2137 bytes)
17/06/06 02:57:53 INFO Executor: Running task 0.0 in stage 29.0 (TID 24)
17/06/06 02:57:53 INFO HadoopRDD: Input split: hdfs://master/data/spark/search.txt:0+2032
17/06/06 02:57:53 INFO LineRecordReader: Found UTF-8 BOM and skipped it
17/06/06 02:57:53 INFO Executor: Finished task 0.0 in stage 29.0 (TID 24). 2137 bytes result
sent to driver
```

17/06/06 02:57:53 INFO TaskSetManager: Finished task 0.0 in stage 29.0 (TID 24) in 49 ms on localhost (1/1) 17/06/06 02:57:53 INFO TaskSchedulerImpl: Removed TaskSet 29.0, whose tasks have all completed, from pool 17/06/06 02:57:53 INFO DAGScheduler: ResultStage 29 (count at <console>:34) finished in 0.051 s 17/06/06 02:57:53 INFO DAGScheduler: Job 18 finished: count at <console>:34, took 0.081193 s res24: Long = 4
加载文件数据得 1 分 按顺序过滤数据得 1 分 进行计数且结果为 4 得 1 分

第四部分：SaaS 云应用开发（20 分）

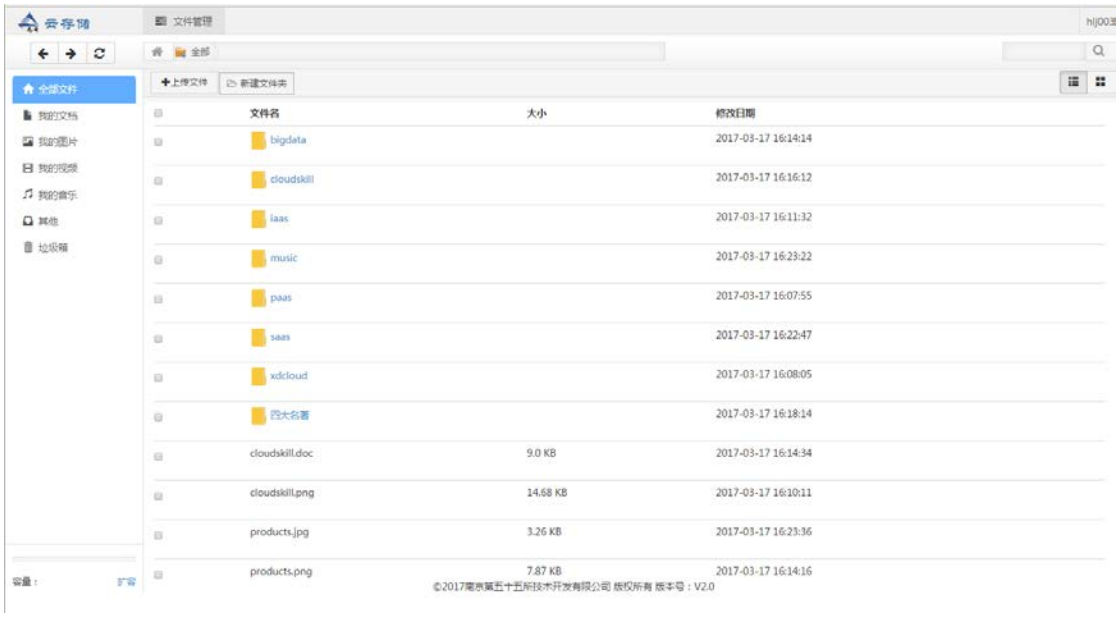
任务一、云存储 WEB 应用开发（6 分）

云存储网盘 Web 应用的开发，选用 javaEE 技术平台，使用集中部署的云存储服务。

开发环境：SDK(xd-cloudstorage-web-v2.0) + Eclipse + JDK + Tomcat + Mysql + swift.sql + 案例工程 cloudstorage_web。

1. 搭建开发环境和导入项目（1 分）

根据指定的账户名、密码等信息修改连接云平台的配置，并运行。按顺序提交浏览器登录后“全部文件”页面截图、修改的配置及修改的配置代码到答题框。

效果截图    2、代码 1）.修改数据库配置文件：jdbc.properties jdbc.driverClassName=com.mysql.jdbc.Driver

jdbc.url=jdbc:mysql://localhost:3306/swift?useUnicode=true&characterEncoding=UTF-8
jdbc.username=root
jdbc.password=123456
2）修改云存储配置文件
#1 Config 1 （填写时选手自己的考号和密码，案例为 gw001，000000）
USERNAME=gw001
PASSWORD=000000
AUTHURL=http://swift:35357/v2.0/tokens
TENANTNAME=gw001

看到“全部文件”下的文件列表，数据和参考截图一致（0.5 分）：
代码（0.5 分）

2. “我的图片”功能（2 分）

实现云存储网盘“我的图片”功能，通过“我的图片”导航展示当前网盘所有图片的文件列表。按顺序提交运行的网页截图和增改的 java 代码到答题框。

Jsp 页面
href="category.action?type=1"
Control 层

```
/**
 * 取得特定类型的文件
 *
 * @param request
 * @param response
 * @param type
 *                  :1 图片格式 2 文档格式 3 视频格式 4 音乐格式 5 其他
 * @return
 */
@RequestMapping("/category")
public ModelAndView category(HttpServletRequest request,
        HttpServletResponse response, int type) {

    ModelAndView view = new ModelAndView();
    User user = getSessionUser(request);
    String upath = request.getSession().getServletContext()
            .getRealPath("/");
    List list = storage.getCategoryStoredList(user.getUsername(), type, upath);
    view.addObject("list", list);
    view.addObject("search", "true");
    view.addObject("type", type);
    view.setViewName("/main");
    return view;
}
```

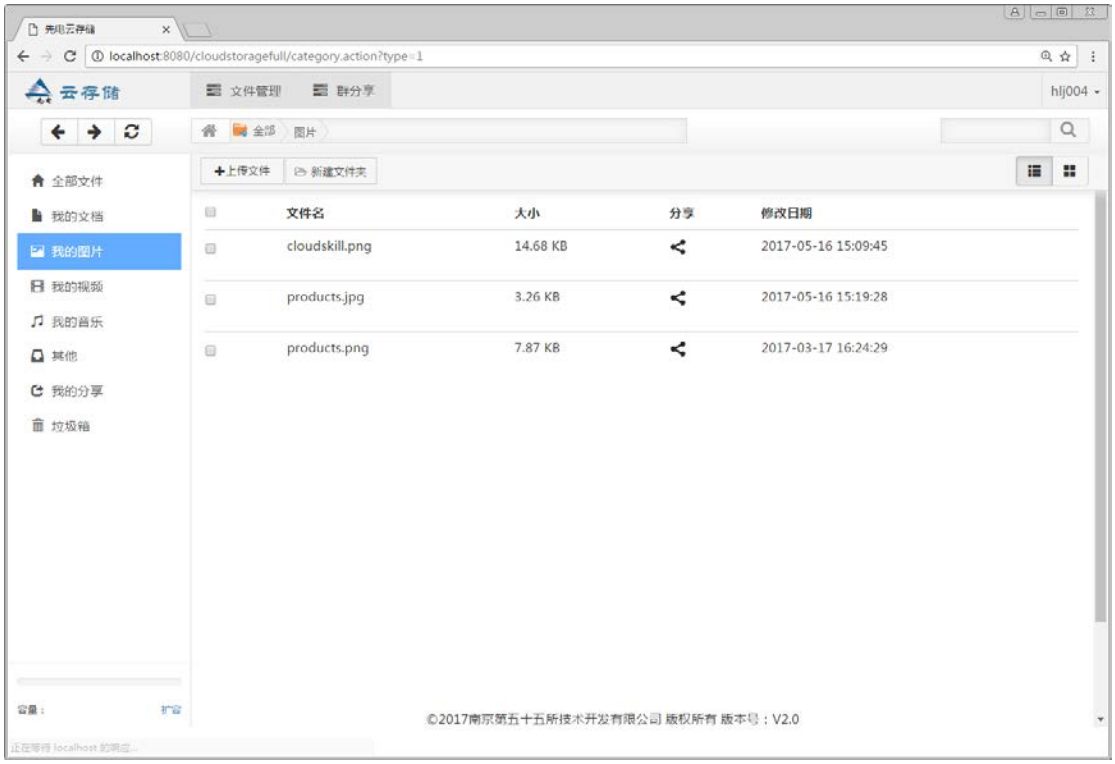
Service 层

```
/**
```

```
     * 取得路径下对应类型的所有文件
     *
     * @param email
     * @return
     */
    public List getCategoryStoredList(String username, int type, String upath) {
        SwiftDFS swiftdfs = new SwiftDFS();
        List categoryStoredList = swiftdfs
                .getCategoryStoredList(username, type);
        return categoryStoredList;
    }
```

考点：（以上是参考答案不是唯一答案）

SwiftDFS 内的 getCategoryStoredList 其中的 type 传的值是 1
SwiftDFS swiftdfs = new SwiftDFS();
List categoryStoredList = swiftdfs.getCategoryStoredList(username, type);



我的“图片选中”(0.5 分)，cloudskill.png、products.png、products.jpg 三个图片(0.5 分)、展示成如下列表方式（0.5 分）
代码(0.5 分)

3. 下载多个文件（3 分）

实现云网盘的多个文件下载的功能，以 zip 格式打包下载网盘中“四大名著\txt”目录下的“三国演义.txt”和“水浒传.txt”，按顺序提交运行的网页截图和增改的 java 代码到答题框。

Jsp 页面

```
//下载文件
        function downloadfile() {
            var objTable = document.getElementById("tab");
            var data = '';
            for ( var y = 0; y < objTable.rows.length; y++) {
                var checkbox =
objTable.rows[y].childNodes[1].childNodes[0].childNodes[5];
                if (checkbox.checked == true) {
                    var td3 =
checkbox.parentNode.parentNode.parentNode.childNodes[3];
                    var imgpic = td3.getElementsByTagName("img");
                    if (imgpic.length > 0) {
                        alert("文件夹中内容暂时无法提供下载，请进入文件夹选择相
应文件进行下载");
                        return;
                    } else {
                        var path =
checkbox.parentNode.parentNode.parentNode.childNodes[5].innerHTML;
                        path = decodeURIComponent(path);
                        var names = td3.childNodes[0].childNodes[0]
                        var name = names.innerHTML
                    }
                    data += path + ",";
                }
            }
            if (imgpic.length > 0) {
                window.location.reload();
            } else {
                data = data.substring(0, data.length - 1);
                $("#download").attr("href", "download.action?paths=" + data);
            }

            var filepath = '';
            var filename = '';
            var isdir_share = '';
            var filelength = '';
            function share(name, path, isdir, length) {
                filename = name;
                filepath = path;
                isdir_share = isdir;
                filelength = length;
                $("#modal3").css("display", "block");
            }
```

```
        }
Control 层
    @RequestMapping("/download")
    public void download(HttpServletRequest request,
            HttpServletResponse response, String paths) {
        try {
            String[] strs = paths.split(",");
            int len = strs.length;
            if (len < 1) {
                return;
            }
            User user = getSessionUser(request);
            response.setContentType("text/html;charset=UTF-8");
            request.setCharacterEncoding("UTF-8");
            BufferedInputStream bis = null;
            BufferedOutputStream bos = null;
            String path = request.getSession().getServletContext()
                    .getRealPath("/");
            String ctxPath = path + "export";

            if (len == 1) {
                String filepath = strs[0];// (String) map.get("path");
                filepath = UtilTools.converStr(filepath);
                byte[] b = new byte[] {};

                if (storage instanceof SwiftStorageService) {
                    b = ((SwiftStorageService) storage).download(
                            user.getUsername(), filepath);
                }

                if (storage instanceof HadoopStorageService) {
                    String[] strsss = filepath.split("/");
                    String filename = strsss[strsss.length - 1];
                    String downLoadPath = ctxPath + File.separator + filename;
                    ((HadoopStorageService) storage).download(filepath,
                            downLoadPath);

                }

                String[] strsss = filepath.split("/");
                String filename = strsss[strsss.length - 1];

                String downLoadPath = ctxPath + File.separator + filename;
                File downLoadFile = new File(downLoadPath);
```

```
            downLoadFile.createNewFile();
            BufferedOutputStream buffer = new BufferedOutputStream(
                    new FileOutputStream(downLoadFile));
            buffer.write(b);
            buffer.flush();
            buffer.close();

            long fileLength = downLoadFile.length();
            response.setContentType("application/octet-stream");
            response.setHeader(
                    "Content-disposition",
                    "attachment; filename="
                            + new String(filename.getBytes("utf-8"),
                                    "ISO8859-1"));
            response.setHeader("Content-Length", String.valueOf(fileLength));
            bis = new BufferedInputStream(new FileInputStream(downLoadPath));
            bos = new BufferedOutputStream(response.getOutputStream());
            byte[] buff = new byte[2048];
            int bytesRead;
            while (-1 != (bytesRead = bis.read(buff, 0, buff.length))) {
                bos.write(buff, 0, bytesRead);
            }
            bis.close();
            bos.close();

            downLoadFile.delete();

        } else {
            String[] strss = strs[0].split("/");
            String temps = strss[strss.length - 1];
            String[] strr = temps.split("\\.");
            strr[0] = UtilTools.converStr(strr[0]);
            String rootname = ctxPath + File.separator + strr[0];
            File rfile = new File(rootname);
            rfile.mkdir();
            for (int i = 0; i < len; i++) {
                String filepath = strs[i];// (String) map.get("path");
                filepath = UtilTools.converStr(filepath);
                byte[] b = new byte[] {};
                if (storage instanceof SwiftStorageService) {
                    b = ((SwiftStorageService) storage).download(
                            user.getUsername(), filepath);
                }
                if (storage instanceof HadoopStorageService) {
```

```
                String[] strsss = filepath.split("/");
                String filename = strsss[strsss.length - 1];
                String dpath = rootname + File.separator + filename;
                ((HadoopStorageService) storage).download(filepath,
                        dpath);
                continue;
            }

            String[] strsss = filepath.split("/");
            String filename = strsss[strsss.length - 1];

            String dpath = rootname + File.separator + filename;
            File dfile = new File(dpath);
            dfile.createNewFile();
            BufferedOutputStream buff = new BufferedOutputStream(
                    new FileOutputStream(dfile));
            buff.write(b);
            buff.flush();
            buff.close();
        }
        // 打包生成 tar.gz 文件
        File tarfile = new File(rootname + ".zip");
        tarfile.createNewFile();
        UtilTools.WriteToTarGzip(ctxPath + File.separator, strr[0],
                strr[0] + ".zip");

        // 下载 tar.gz 文件
        String downLoadPath = rootname + ".zip";
        String fp = "[批量下载]" + strr[0] + ".zip";
        //
        File downLoadFile = new File(downLoadPath);
        long fileLength = downLoadFile.length();
        //
        response.setContentType("application/octet-stream");
        response.setHeader(
                "Content-disposition",
                "attachment; filename="
                        + new String(fp.getBytes("utf-8"), "ISO8859-1"));
        response.setHeader("Content-Length", String.valueOf(fileLength));
        bis = new BufferedInputStream(new FileInputStream(downLoadPath));
        bos = new BufferedOutputStream(response.getOutputStream());
        byte[] buff = new byte[2048];
        int bytesRead;
        while (-1 != (bytesRead = bis.read(buff, 0, buff.length))) {
```

```
                    bos.write(buff, 0, bytesRead);
                }
                bis.close();
                bos.close();

                downLoadFile.delete();
                UtilTools.deletefile(rootname);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
```

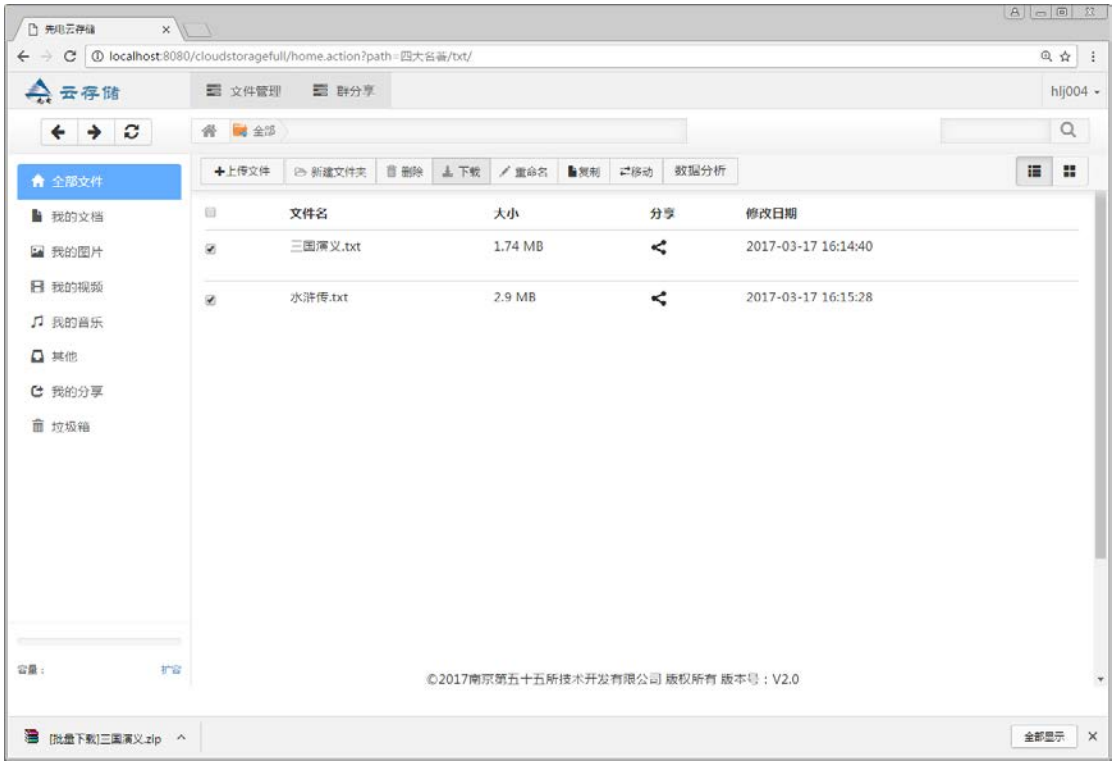
Service 层

```
    /**
     * 下载文件
     *
     * @param email
     * @param path
     */
    public byte[] download(String username, String path) {
        SwiftDFS swiftdfs = new SwiftDFS();
        return swiftdfs.downloadFile(username + "/" + path);
    }
```

考点：（以上是参考答案不是唯一答案）

考核 swiftDFS 内的 download 的方法

```
SwiftDFS swiftdfs = new SwiftDFS();
return swiftdfs.downloadFile(username + "/" + path);
```



选择“三国演义.txt”和“水浒传.txt”2 个文件(0.5 分)，有下载按钮(0.5 分)，有下载结果（0.5 分），下载格式显示 zip 文件(1.0 分)。
代码：（0.5 分）

任务二、云存储网盘客户端（6 分）

云存储网盘客户端 APP 的开发，选用 Android 开源技术平台，使用集中部署的云存储服务。

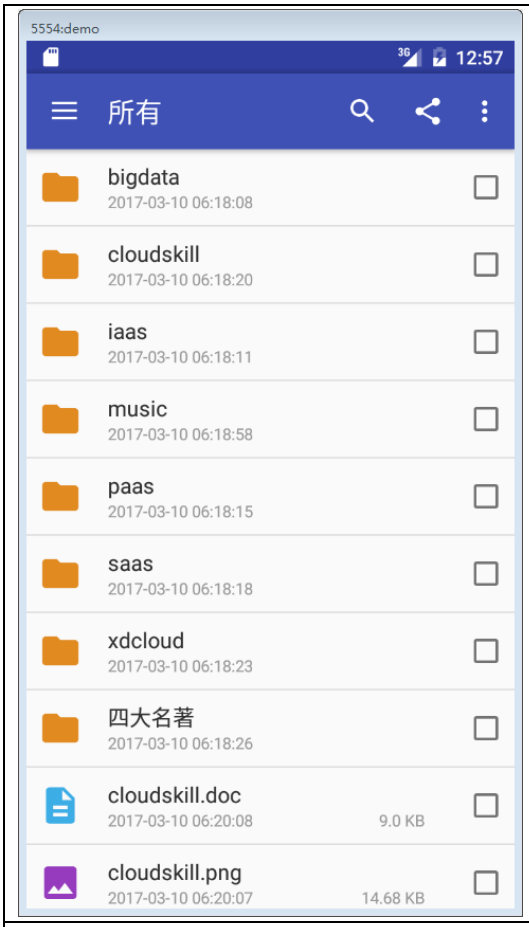
开发环境：SwiftSDK(openstack-java-sdk） + Android 开发环境（Android Studio） + JDK + 案例工程 swiftstorage，程序的运行采用 Android Studio 默认模拟器。

1. 搭建开发环境（1 分）

根据指定的账户名、密码等信息修改连接云平台的配置并运行。按顺序提交 APP 登录后“所有”窗口的模拟器截图及增改的 java 代码到答题框。

```
private    String openstack_ip = "58.214.31.6";
```

效果图：



“所有”界面：文件目录列表和以下截图一致。(0.5 分)
增改的代码(0.5 分)

2. 拍照上传功能（2 分）

实现网盘 APP“所有”根目录下拍照上传功能，照片文件命名规则为：image + 拍照时间 + 临时文件随机编号，文件格式为 jpg 格式，如“image_20170101080808_123.jpg”。进入“iaas”目录，点击拍照并上传照片文件。按顺序提交“iaas”目录下文件列表的模拟器截图和增改的 java 代码到答题框。

代码：
MainFragment:

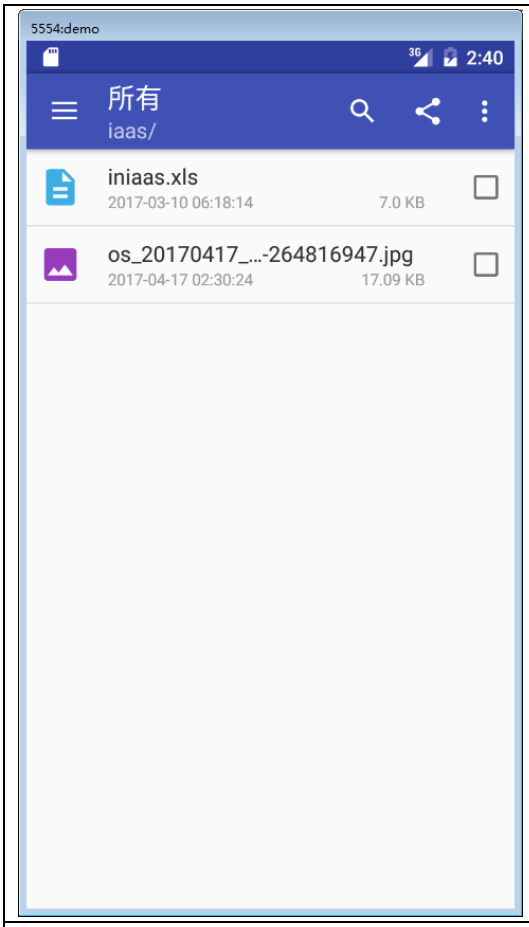
```
@Override
public void takePhoto() {
    runtakePhoto();
}


private static final int ACTION_SELECT_CONTENT_FROM_CAMERA = 3;
//拍照临时存储路径
private Uri mImageUri;

/**
 * 拍照
```

```
 */
private void runtakePhoto() {
    try {
        //创建 Intent
        Intent cameraIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        //临时文件传递参数
        mImageUri = Uri.fromFile(createPicTempFiles());
        cameraIntent.putExtra(MediaStore.EXTRA_OUTPUT, mImageUri);
        //启动，系统自己选择
        startActivityForResult(cameraIntent,
ACTION_SELECT_CONTENT_FROM_CAMERA);
    } catch (IOException e) {
        e.printStackTrace();
    }
}

/**
 * 创建临时文件，保持文件。
 *
 * @return
 * @throws IOException
 */
public File createPicTempFiles() throws IOException {
    String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss")
            .format(new Date());
    String imageFileName = "os_" + timeStamp + "_";
    File storageDir = Environment
            .getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES);
    File image = File.createTempFile(imageFileName,
            ".jpg",
            storageDir
    );
    return image;
}


@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {

    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode) {
        case ACTION_SELECT_CONTENT_FROM_CAMERA:
            //成功返回
            if (resultCode == Activity.RESULT_OK) {
```

```
                    Uri uri = mImageUri;
                    //上传图片
                    UploadCameraObjectTask uploadObjectTask = new
UploadCameraObjectTask(uri);
                    uploadObjectTask.execute();
                    break;
                }
            default:
                break;
        }

}

/**
 * 上传拍照图片
 */
private class UploadCameraObjectTask extends
            AsyncTask<String, Object, TaskResult<ObjectForUpload>> {
        private Uri fileUri;

        /**
         * 上传拍照图片。
         *
         * @param fileUri
         */
        private UploadCameraObjectTask(Uri fileUri) {
            this.fileUri = fileUri;
        }

        protected TaskResult<ObjectForUpload> doInBackground(String... params) {
            try {
                //文件存储位置，如果拍照存本地在临时目录
                String filePath = DisplayUtils.getOriginalFilePath(
                        getActivity(), fileUri);
                //上传当前目录
                String directory = getAppState().getSelectedDirectory().getName();
                //本地临时目录只取文件名称
                String[] parts = filePath.split("/");
                String path = directory + parts[parts.length - 1];
                String contentType = "image/jpeg";
                String containerName = getAppState().getSelectedContainer().getName();
                InputStream fileInputStream = new FileInputStream(filePath);
                //调用上传服务
                ObjectForUpload objectForUpload = getService().upload(containerName,
```

```
fileInputStream, contentType, path);

                return new TaskResult<ObjectForUpload>(objectForUpload);
            } catch (Exception e) {
                return new TaskResult<ObjectForUpload>(e);
            }
        }

        protected void onPostExecute(TaskResult<ObjectForUpload> result) {
            super.onPostExecute(result);
            if (result.isValid()) {
                //刷新当前目录
                GetOSSObjectsTask getObjectsTask = new GetOSSObjectsTask();
                getObjectsTask.execute();
            } else {
                //上传失败
                PromptDialogUtil.showErrorDialog(
                        getActivity(),
                        R.string.alert_take_photo_fail,
                        result.getException(),
                        new Intent(getActivity(), LoginActivity.class));
            }
        }
}
```

效果图：



看到拍照的图片（0.5 分）、文件名称合理“image_201706070909_56789000.jpg”（ 0.5 分），图片在 IaaS/目录下面（0.5 分）。
代码 （0.5 分）

3. 复制功能（3 分）

完善 openstack-java-sdk 底层 copy 的代码，并实现网盘 APP“所有”根目录下复制功能，将“music”文件夹中的“降央卓玛-草原夜色美.mp3”文件复制至“iaas”文件夹中，展示文件复制后在“iaas”文件夹中的列表截图。按顺序提交运行的模拟器截图和增改的 java 代码到答题框。

MainFragment.java

```
/**
 * 复制
 *
 * @param fromPath：初始目录
 * @param toPath：最终目录
 */
@Override
public void copy(String fromPath, String toPath) {
    if (getFirstSelected() != null) {
        this.iscopy = true;
        this.copyFileName = getFirstSelected().getName();
        this.copyFileType = getFirstSelected().getContentType();
```

```
            fileActionBar.setVisibility(View.VISIBLE);
        }

}


if (iscopy) {
        //是复制
        copyToFileName = getAppState().getSelectedDirectory().getName() +
cleanName(copyFileName);
        System.out.println("copyToFileName:" + copyToFileName);
        CopyObjectTask copyObjectTask = new CopyObjectTask(copyFileName,
copyToFileName, copyFileType);
        copyObjectTask.execute();
}


/**
 * 复制功能线程
 */
private class CopyObjectTask extends AsyncTask<String, Object, TaskResult<Object>> {
        private String _path, _pathTo, _type;

        private CopyObjectTask(String path, String pathTo, String type) {
                this._path = path;
                this._pathTo = pathTo;
                this._type = type;
        }

        @Override
        protected TaskResult<Object> doInBackground(String... params) {
                try {
                        String
containName=getAppState().getSelectedContainer().getName().toString();
                        getService().copy(containName,_path,_pathTo,_type);
                        return new TaskResult<Object>((Object) getAppState().getSelectedObject());
                } catch (Exception e) {
                        return new TaskResult<Object>(e);
                }
        }

        /**
         * 复制完成后刷新
         *
```

```
     * @param result
     */
    @Override
    protected void onPostExecute(TaskResult<Object> result) {
        // TODO Auto-generated method stub
        if (result.isValid()) {
            iscopy = false;
            fileActionBar.setVisibility(View.GONE);
            GetOSSObjectsTask getObjectsTask = new GetOSSObjectsTask();
            getObjectsTask.execute();
        } else {
            PromptDialogUtil.showErrorDialog(
                    getActivity(),
                    R.string.error_dlg,
                    result.getException(),
                    new Intent(getActivity(), LoginActivity.class));
        }
    }


}
```

效果图：

在“所有”根目录（0.5 分），“”在 IaaS 目录下（1.0 分），看到“降央卓玛-草原夜色美.mp3”文件(1.0 分)。
代码（0.5 分）。

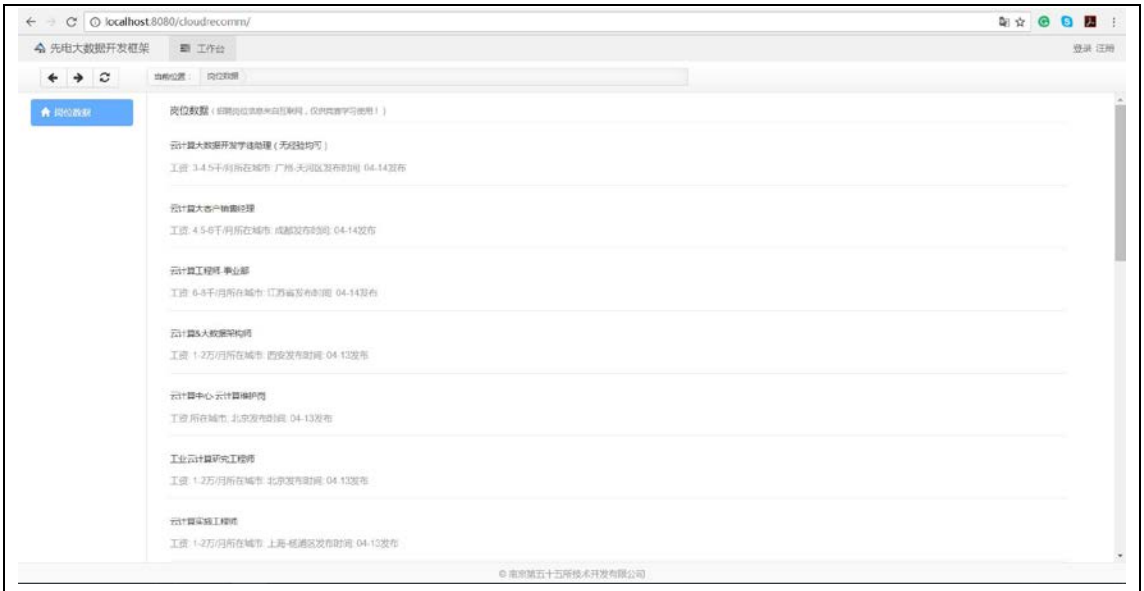
任务三、大数据案例开发（8 分）

大数据案例的开发，选用 javaEE 技术平台，使用镜像提供的大数据平台服务。

开发环境：Eclipse + JDK + Tomcat + Mysql + recomm.sql + 案例工程 cloudrecomm。

1. 搭建开发环境和导入项目（1 分）

修改连接大数据平台的配置并运行。运行后按顺序提交岗位数据的网页截图、修改的配置到答题框。



修改 jdbc.properties
database.url=jdbc:mysql://localhost:3306/recomm?autoReconnect=true&useUnicode=true&characterEncoding=UTF-8&zeroDateTimeBehavior=convertToNull

修改 xd.properties
#hadoop hdfs 配置文件
hdfs=hdfs://192.168.56.102:8020

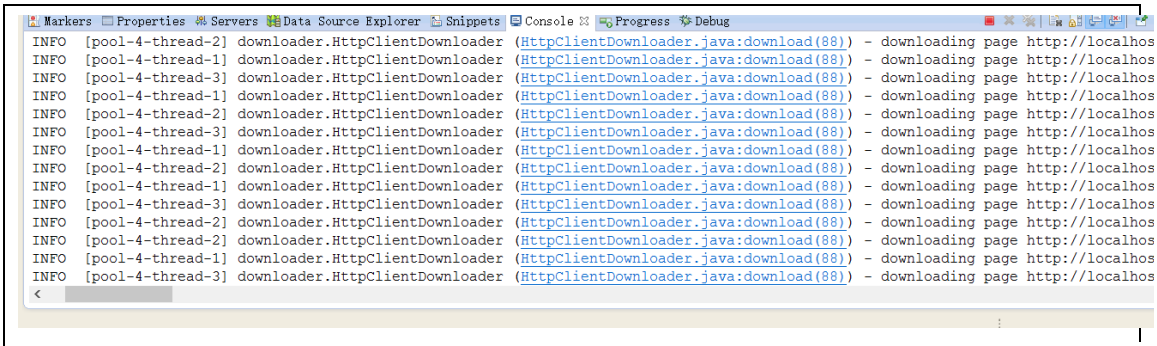
#hadoop mapreduce 配置文件
mapred=192.168.56.102:9001

展示出岗位数据列表(0.5 分)
代码 （0.5 分）

2. 爬取岗位信息（3 分）

通过解析网站页面源代码，爬取所有岗位页面信息中的岗位名称、发布日期、招聘人数、岗位描述等信息，按顺序以制表符分隔，每一个岗位信息一行，并按照发布日期排序（发布日期早的排在前面），存到本地文件目录下，并在网页上增加一个新的表格，含四列信息：第一列为“序号”，列宽占 10%；第二列为“岗位名称”，列宽占 20%；第三列为“发布日期”，列宽占 10%；第四列为“岗位描述”，列宽占 60%。

按顺序提交 Eclipse 控制台运行爬虫的日志截图、实现的网页截图及增改的 java 与 html 代码到答题框。



岗位id	岗位名称	发布日期	岗位描述
1	软件架构师	02-19发布	职位描述：负责数据平台技术架构设计，参与制定技术发展路线与关键技术选 等；结合业务发展与技术发展趋势，提出技术架构设计并不断完善数据平台架 求：1、5年及以上大型系统软件技术架构设计经验2、技术敏感性强，熟悉物 数据以及云计算等技术发展趋势，并有丰富的技术研发经验；3.熟练掌握架构 论知识、方法与工具运用，具备大型系统或平台产品的技术架构设计能力;4、 的抽象思维能力，善于提炼共性技术与框架模块,有智慧城市项目技术研发经验 5、良好的团队协作精神，责任心强，能承受较大的工作压力。职能类别：软 报分享
2	大数据开发工程师（JAVA）华网	02-21发布	职位描述：以下为杭州华网信息技术有限公司大数据研发工程师（JAVA）的招 岗位职责:1、负责大数据应用相关解决方案的设计，进行技术方案材料的撰写 大数据应用相关产品的整体架构设计，进行大数据平台上数据挖掘产品的规划 3、完成各种面向业务目标的数据分析模型的定义和应用开发；4、开发具有数 数据挖掘能力的创新型产品；任职要求：1、计算机相关专业，本科及以上学 上Spark,Hadoop相关开发经验；2、熟悉主流的云计算、大数据开源（hadoop spark、flume等）和数据分析技术（机器学习)并具有相关项目经验；3、精通 数据结构，精通JAVA或Scala语言编程；4、熟悉Linux/Unix平台上的开发环境 路敏捷清晰，良好的表达和理解能力，良好的学习能力，强烈的创新意识；月 20000联系方式：邮箱keketoo@qq.com，电话15325819229职能类别：高级 师关键字：大数据开发举报分享
3	架构师	02-22发布	职位描述：1、主持软件项目/产品的总体架构设计及重要技术决策；2、核心的 任务；3、指导研发团队工程师的产品开发和技术研究工作；4、重点项目的设 持与评审，难点攻关；5、负责推动产品性能优化；6、促进团队技术进步与创 参与公司技术研发体系的流程制度建设和管理。任职资格：1、5年以上软件设 作经验，其中2年以上软件架构经验，大数据、互联网从业经验优先；2、精通 言，精通Spring、SpringMVC、Struts2、Hibernate/MyBatis等开源框架；3、 系统、网络和安全、应用系统架构等有全面的认识；对应用性能监控调优，测 压力测试有相关经验；4、熟悉SOA系统架构设计和实现；熟悉负载均衡技术。 Nginx、Apache；5、精通常见开源系统/框架，能掌握内部工作机制并熟练使

4	系统集成部经理 (职位编号：1)	02-22发布	职位描述：从事IT行业5年经验以上，部门管理和项目管理经验2年以上；了解构、硬件、网络、存储及数据中心SDDC创建；对新技术有热情，熟悉虚拟化、义的数据中心SDDC、IaaS、SaaS，且能不断学习。对微软、VMware、Citrix RedHat了解，有大项目经验。职能类别：技术总监/经理关键字：立即到岗 MCSEPMPSDDC云计算虚拟化VmwareVCPCCNP举报分享
5	云计算讲师	02-22发布	职位描述：云计算方向讲师？岗位职责：1.完成云计算专业的授课任务。按照教纲，教学进度高质量的完成日常授课任务；2.学生答疑：负责帮助学员解决学习的问题，并负责组织和实施学员内部测试；3.配合市场部门的招生宣传工作，讲课或宣讲；4.教学研发工作，主要负责教学大纲、教学PPT、教学案例、教学用发工作；5.负责云计算在线实验的设计与实验指导书的撰写；6.负责云计算案例频、教程的测试和验证；7.教学过程中的其他临时工作。？任职要求:1.计算机专业，三年以上工作经验，至少一年以上培训行业经验2.积极向上，具有较强的自3.较好的英文技术文档阅读能力与经验；4.熟练掌握C/C++、Java或Python中的种语言并熟悉Linux系统编程；5.熟悉Linux内核网络协议栈代码，扎实的网络知协议/二层网络/路由等）；6.掌握网络基础知识；网络组成、ISO/OSI七层模型模型、IP协议、子网划分等；7.精通MySQL数据库管理；8.熟悉网络虚拟化，对OpenvSwitch和OpenFlow等有深入了解，并熟悉相关的网络硬件设备；9.熟悉OpenStack或CloudStack等开源云平台并有实际项目经验者优先；10.熟悉主流台和云平台，深刻理解云平台应用部署与传统应用系统部署的区别，有Hadoop维经验者;11.强烈责任心、良好的沟通和学习能力、团队精神和服务意识。？工待遇1.签订正式劳动合同；2.带薪年休+周末双休+餐补，人性化的工作环境；3行各种培训（公司提供相关学习资料，书籍等）；4.广阔的发展空间和晋升平台供丰富多彩的员工活动，拓展训练，定期旅游；6.节日发放福利，生日福利，年职能类别：其他举报分享
6	云数据中心方案架构师	02-22发布	职位描述：岗位描述：1、负责云数据中心整体方案及交付的规划与精进，确保性、先进性和竞争力2、为重点行业提供有针对性的行业云数据中心整体方案，PoC（概念验证）3、为行业关键用户提供云数据中心解决方案的咨询支持和实选择先进的云计算技术以及云计算架构设计满足客户需求岗位要求：1、本科以至少5年的IT或CT技术类岗位经验，3年以上数据中心方向工作经验2、熟悉服务储、网络、安全、云平台、虚拟化、运维等领域技术3、熟悉openstack与vmw
7	测试开发工程师（云数据中心）	02-22发布	职位描述：角色介绍：云数据中心相关产品质量、性能检测方案设计与开发岗位参与云数据中心检测相关工作。岗位要求：1.本科以上学历，1年以上云计算相关试设计经验；2.针对客户需求参与测试方案的编制工作，负责跟客户沟通测试内熟悉OpenStack架构，包括计算、存储、网络等组件，有社区开发经验者优先测试方案的讨论，整改工作，对测试成本、工期进行管控；5.制定测试计划，安内容；按计划安排进行测试实施；6.具有较强的学习能力，及分析/解决问题的良好的文档书写和口头表达能力；8.良好的工作压力承受能力、负责任；9.计算信、电子及相关专业。职能类别：软件测试系统测试举报分享
8	云平台架构师	02-22发布	职位描述：岗位职责：1.负责云平台系统架构的分析设计及优化2.负责技术研究关，跟进社区及主要商业版本的技术路线，把握技术研究的大方向3.分析客户需供私有云产品的整体解决方案，并形成具体的目标及开发计划岗位要求：1.本科历，3年以上云计算相关领域开发设计经验2.精通linux系统3.熟悉OpenStack、CloudStack等云计算平台，精通OpenStack架构，包括计算、存储、网络等组区开发经验者优先4.有Kubernates、OpenDC/OS相关项目开发经验者优先5.熟系统，有相关项目开发经验者优先6.熟悉服务器虚拟化技术7.精通Java、Scala Python、Ruby、Go中的至少一种语言职能类别：系统架构设计师举报分享
9	桌面云研发工程师	02-23发布	职位描述：岗位职责：1、桌面云产品的功能设计与编码；2、参与文档和代码3、从事与桌面云有关的技术调研工作；4、组织或参与定期的技术培训。5、编开发设计文档。岗位要求：1、熟练使用Python编程语言。2、熟练使用IDEA/Eclipse/Pycharm等至少一种集成环境。3、熟悉Linux操作系统和网络。web开发流程，熟悉restful服务开发。5、具有独立学习、调研、思考的能力，设计文档并实现编码。6、对云计算行业的研发工作有浓厚的兴趣。7、有桌面发经验者优先录用。职能类别：软件测试系统测试举报分享
10	云平台研发工程师	02-23发布	职位描述：岗位职责：1.云平台的功能设计与编码；2.参与文档和代码的评审；参与SCRUM相关工作；4.从事与云平台有关的技术调研工作；5.组织或参与定培训。岗位要求：1.熟练使用Python编程语言。2.熟练使用IDEA/Eclipse/Pycha少一种集成环境。3.熟悉Linux操作系统和网络。4.掌握web开发流程，熟悉res发。5.具有独立学习、调研、思考的能力，能够编写设计文档并实现编码。6.对行业的研发工作有浓厚的兴趣。7.了解或使用过Openstack/Vmware/CloudStac录用。职能类别：软件测试举报分享

岗位id	岗位名称	发布日期	岗位描述
11	物联网平台软件架构师	02-23发布	职位描述：岗位职责：1．负责物联网应用的需求分析与方案设计；2．负责 的需求分析和软件架构设计工作；3．负责物联网平台的基础软件开发工作； 织物联网平台的软件开发工作；5．负责物联网平台相关产品容灾、集群、性 及相关产品规划工作；岗位要求：1．本科及以上学历，计算机或通信相关专 年以上软件开发经验，三年以上软件架构设计经验，具有互联网运营平台架构 佳；3．熟悉通用开源软件平台，Hadoop，redis，MySQL，Linux，MQ等； 计算技术，OpenStack，虚拟化，Docker，IaaS/PaaS等技术；5．精通C++ 熟悉Windows、Linux下服务器软件开发；6．精通TCP/IP网络技术，熟悉We 相关技术；7．良好的团队协作能力。注：因组织架构调整，正式入职单位为 对控股子公司，请注意！职能类别：其他举报分享
12	云安全研发工程师--第四事业部	02-23发布	职位描述：岗位职责：1、跟踪云计算主流最新开源框架；2、负责云安全产品 作；3、负责与其它产品的兼容性测试及整合工作；4、负责技术难点攻关；5 关技术文档及专利申请工作；6、承担国家部委云安全项目课题研究和项目管 品技术支持工作。任职要求：1、硕士以上学历，信息安全、计算机、软件等 业；2、熟悉当前主流云计算架构，了解一种云计算开源框架，参与过OpenS CloudStack或Eucalyptus等项目优先；3、有身份认证、网络安全、数据安全 化、虚拟化安全等领域产品开发者优先；4、精通至少一门编程语言，如JAV Python；5、熟悉JAVAweb开发，及struts2（或springMVC）、spring、hibe MyBatis）框架；6、对云计算底层技术架构和虚拟化解决方案有一定的研究 业、责任心强，良好的沟通能力及团队合作精神，能承受工作压力。职能类别 程师举报分享
13	物联网平台软件架构师	02-23发布	职位描述：岗位职责：1．负责物联网应用的需求分析与方案设计；2．负责 的需求分析和软件架构设计工作；3．负责物联网平台的基础软件开发工作； 织物联网平台的软件开发工作；5．负责物联网平台相关产品容灾、集群、性 及相关产品规划工作；岗位要求：1．本科及以上学历，计算机或通信相关专 年以上软件开发经验，三年以上软件架构设计经验，具有互联网运营平台架构 佳；3．熟悉通用开源软件平台，Hadoop，redis，MySQL，Linux，MQ等； 计算技术，OpenStack，虚拟化，Docker，IaaS/PaaS等技术；5．精通C++ 熟悉Windows、Linux下服务器软件开发；6．精通TCP/IP网络技术，熟悉We
14	云计算安全工程师	02-23发布	职位描述：岗位职责：1、负责公司云平台产品的安全设计和评估；2、负责公 件分析与追溯，高危漏洞的应急响应；3、负责自动化代码安全扫描、测试系 发;4、负责业务安全漏洞、技术安全漏洞的扫描规则开发实现；岗位要求：1、 WEB安全，熟悉各种WEB攻防技术以及安全漏洞原理，掌握多种安全行为的原 现方法，有过独立分析或挖掘漏洞的经验；2、熟悉WEB软件开发流程，至少 java，js、python、php、c++其中两种开发语言；3、熟悉漏洞扫描、渗透测试 通常见漏洞/木马的原理、危害、利用方式、检测、和修复方案；4、时刻关注 WEB、系统漏洞，能独立分析漏洞原理，实现PoC5、有云计算相关领域安全 先；6、具有冲击对抗、流量攻击、伪装通讯等黑客攻击且有规模防御成功经 7、具有在安全公司的安全相关从业经历，有过大型互联网漏洞挖掘经验者优先 别：网络信息安全工程师其他举报分享
15	云计算产品经理	02-23发布	职位描述：岗位职责：1.负责售前咨询工作，配合和帮助销售人员完成既定销 2.负责与客户进行沟通和交流，挖掘和引导用户需求，为客户提供合理的解决 术建议书、安全规划等；3.参与项目的招投标，负责撰写招投标相关的文档、 的讲标和答疑工作。任职要求：1.大专及以上学历，计算机、电子或通信相关 云计算产品有简单了解。2.1年以上的系统集成售前工作经验或3年以上IT行业 验；3.有良好网络和安全基础，熟悉网络主流产品线：CISCO、H3C、华为、 等；4.责任心强、良好的沟通、表达能力，文档组织能力，思维逻辑清晰；5. 师或产品经理工作经验者优先;6.有云计算售前经验优先（一经录取，我们将提 产品系统培训）.职能类别：售前/售后技术支持工程师举报分享
16	客户经理(高性能、云计算方向)	02-24发布	职位描述：岗位职责:1）负责相关产品在市场上的推广与销售；2）制定销售计 销售任务；3）积极开拓电信、电力、交通、军工、政府等目标市场，负责客户 立与维护；4）掌握产品相关知识，熟悉产品应用方案，有很好的沟通能力。 求:1）大学本科以上学历，计算机软硬件专业；2）有2年以上销售计算机软硬 方案的经验；3）热爱本职工作，能够承受工作压力；4）有在电信、电力、交 工、政府等行业的客户资源，熟悉行业特点；5）具备计算机系统的软、硬件知 学习新知识的能力。职能类别：大客户销售销售工程师关键字：客户经理市场 销售云计算销售销售高性能云计算举报分享

17	虚拟网络开发工程师(数据面方向)(001990) (职位编号：suning001990)	02-24发布	职位描述：岗位职责:1、负载维护公司云计算平台转发面组件OpenVswitch、LinuxBridge等，解决软件网络组件在实际使用中的技术问题；2、了解业界数的新技术和新发展，积极引入先进技术；3、理解openstackneutron的vlan/vx网模型，能够做相关定制开发；4、能够帮助运维，针对出现的网络有关问题析定位和解决。任职资格:1、本科以上学历，计算机相关专业，至少有3年以经验；2、熟悉Linux系统和内核原理，具备内核网络子系统相关开发经验；3、linuxbridge，veth，tun/tap设备原理；4、熟悉数据面加速原理和技术，如DP VMDq，SRIOV等技术，有实际开发和部署经验；5、精通C语言，熟悉Bash/python/Lua等至少一种脚本语言；6、有虚拟网络优化经验优先；7、良作精神，易于沟通，能够部门间协同工作。职能类别：软件工程师举报分享
18	系统资深工程师	02-24发布	职位描述：工作职责：1、对数据中心服务器、存储等设备进行日常管理，保及安全可用；2、负责服务器网络安全和数据安全的保障工作；3、研究数据中施架构,实施架构优化和性能改进工作；4、撰写技术文档，参与运维流程优化需求完成服务器/存储设备软硬件平台搭建/维护/故障排除；6、负责运算资源划、设计、部署、管理与维护；7、负责存储系统的架构规划、优化、设计及护工作；8、负责运维公司服务器资源、存储、数据备份等设备，确保系统的满足业务高速增长的需求；9、负责公司私有云优化、管理与维护，了解和研的技术发展趋势和最新的解决方案；10、协助部署、监控应用及数据库的日常职要求：1.大学本科或以上学历，8年以上工作经验，其中系统领域至少5年以验。2.精通WindowsServer，Linux等操作系统，对Windows、Linux具备丰富维护经验，能熟练编写shell脚本；3.精通VMware和hyper—v虚拟化的部署，护；4.熟悉主流的存储系统和备份软件系统，如emc、hp、ibm、veritas等。5数据库技术（Oracle、SQLServer等）；6.熟悉云计算概念、架构，了解云计状及趋势；7.熟悉网络相关知识，了解主流CISCO网络设备；8.具备较强的学良好的沟通、规划、管理能力和强烈的责任心，良好的团队合作意识；职能类支持/维护经理关键字：系统工程师，系统领域、服务器举报分享
19	安全开发工程师（云安全）(002351) (职位编号：suning002351)	02-24发布	职位描述：岗位职责:1.负责安全产品的研发工作，推动技术架构不断优化；2.负威胁风险分析、未知Web漏洞分析等安全产品的开发，对网络安全攻击进行高效及拦截。任职资格:1.精通Java或C开发语言；2.熟悉Spark、Hive或Pig原理与实有大数据平台Hadoop/Storm/Spark开发经验；3.熟悉常用设计模式，具有大型高并发、高负载、高可用性系统设计开发经验；4.有Linux内核开发经验和网络经验者优先；5.对互联网、云计算、云安全有热情，学习能力强；5.具有良好的计能力，思路清晰，能独立分析和解决问题,责任心强，具备良好的团队合作精类别：互联网软件开发工程师举报分享
20	大数据项目助理/产品助理	02-24发布	职位描述：岗位职责：1.协助大数据项目经理进行业务理解及需求分析；2.编写档、需求文档、用户手册等项目材料；3.项目相关的规章制度、流程文档等收集归档维护及总结分析；4.项目实施过程中的质量、进度、问题跟踪汇报；岗位条计算机相关全日制本科学历；2.2年或以上IT项目相关经验；3.熟悉常用办公软件的文字组织表达能力；----------公司简介----------★【广业公司】（全省广业资产经营有限公司”）是广东省国资委下第二大的国有多元化产业经营集亿级资产，中国500强，下设15家产业集团，两万人以上规模，旗下多家上市公★【广业开元科技】是在省政府和国资委重点关注支持，由广业公司投资千万控的大数据科技公司，不仅拥有雄厚的资本支持，业务更广泛涉及政府、公共服务国企事业单位、科技创新园区等多个高端领域。★核心团队来自于日立信息、阿巴、华为、SAP、金山、YY等国际国内一流IT科技公司，以及云计算和大数据内著名专家、博士后流动工作站组成，90%以上员工为技术工程师。职能类别：程师技术文员/助理举报分享

```
package com.xiandian.cloud.web;
class RecommController.java
// 爬取岗位
    @RequestMapping("/crawljob")
    @ResponseBody
    public Object crawljob(HttpServletRequest request) {
        try {
```

```
        // 应用部署路径
        String path = request.getSession().getServletContext()
                .getRealPath("/");
        HdfsClient hdfsClient = HdfsClient.getInstance();
        hdfsClient.delete("/user/data_in/");
        // 爬取路径
        String crawlpath = path + "webmagic";
        String sortpath = path + "sortingresult.txt";
        File file = new File(crawlpath);
        if (file.exists()) {
            File[] files = file.listFiles();
            for (File file2 : files) {
                file2.delete();
            }
        }
        File sortfile= new File(sortpath);
        if (sortfile.exists()){
            sortfile.delete();
        }
        // webmagic 爬取岗位
        Spider.create(new CrawlJob())
                .addUrl("http://localhost:8080/cloudrecomm/course/job/1")
                .addPipeline(new PipelineJob(crawlpath)).thread(3).run();
        // 上传到 hdfs 指定目录
        // 循环爬取下来的目录：放到 hadoop 的/user/data_in/目录
        if (file.exists()) {
            File[] files = file.listFiles();
            for (File file2 : files) {
                if (!file2.isDirectory()) {
                    PipelineJob.sort(file2.getAbsolutePath(),sortpath);
}
                }
        }
        //读取爬取文件并写入数据库
        jobCrawlService.removeAll();
        File textfile = new File(sortpath);
        BufferedReader b = new BufferedReader(new InputStreamReader(
                new FileInputStream(textfile), "utf-8"));
        String line;
        while((line=b.readLine())!=null ){
            String[] strs = line.split("\t");
            String jobname=strs[0];
            String jobds=strs[3];
            //插入数据库
```

```
                    JobCrawl js = new JobCrawl();
                    js.setName(jobname);
                            js.setCreatetime(jobtime);
                    js.setDescription(jobds);
                    jobCrawlService.save(js);

                }
                return new MessageBean(true);
            } catch (Exception e) {
                return new MessageBean(false);
            }
        }

        @RequestMapping("/showcrawljob")
        @ResponseBody
        public Object crawljob(HttpServletRequest request,int pageNo) {
            Page page = jobCrawlService.getAll(pageNo, Constants.PAGE_SIZE_10);
            JobCrawl jobCrawl = new JobCrawl();
            List<JobCrawl> result = page.getResult();
            List<JobCrawl> resList = new ArrayList<>();
            for (JobCrawl cj : result) {
                resList.add(cj);
            }

            return resList;

        }
package com.xiandian.cloud.bigdata.crawl;
class CrawlJob.java
public void process(Page page) {

            Selectable select=null;
            List<String> urls=null;
            // 判断是否为列表页，如是，添加岗位信息页以及后续分页的 url 至抓取链接
队列中
            if (page.getUrl().toString().contains("/cloudrecomm/course/job/")) {
                select = page.getHtml().xpath("//div[@class='yq-new-item item-two']/h3");
                urls = select.links().all();
                page.addTargetRequests(urls);
                // 获取总共页数
                String totalpage = page.getHtml().xpath("//div[@class='pull-left
totalpages']/text()").toString();
                if (totalpage != null)
                {
```

```
                    totalpage = totalpage.substring(1,totalpage.length()-1);
                    int totalnum = Integer.parseInt(totalpage);
                    int i = 2;
                    while(i<totalnum+1){
                        urls.add("/cloudrecomm/course/job/"+i);
                        i++;
                    }
                }

                if (urls != null && urls.size() > 0)
                {
                    page.addTargetRequests(urls);
                }
            }
            // 判断是否为岗位信息页，如是，则抓取其中的各项信息
            else if (page.getUrl().toString().contains("/cloudrecomm/51job/")) {
                page.putField("url", page.getUrl().toString());
                // 岗位名称
                page.putField("job",
page.getHtml().xpath("//div[@class='cn']/h1/text()").toString());
                // 发布日期
                page.putField("createtime", page.getHtml().regex("<em
class=\"i4\"></em>([^<]+)</span>").toString());
                // 岗位描述
                page.putField("content", page.getHtml().xpath("//div[@class='bmsg job_msg
inbox']/html()").toString());
                 // 招聘人数
                page.putField("number", page.getHtml().regex("<em
class=\"i3\"></em>([^<]+)</span>").toString());

            }
        }
package com.xiandian.cloud.bigdata.crawl;
class PipelineJob.java
public PipelineJob() {
            setPath("/data/webmagic");
        }

        public PipelineJob(String path) {
            setPath(path);
        }

        @Override
        public void process(ResultItems resultItems, Task task) {
```

```
        try {
            //定义存储路径，以每天为一个文件存储
            String path = this.path + PATH_SEPERATOR +
UtilTools.timeTostrYMD(new Date()) + ".txt";
            File file = getFile(path);
            String str =resultItems.get("job")
                    +"\t"
                    + resultItems.get("number")
                    + "\t"
                    + resultItems.get("createtime")
                    + "\t"
                    +
UtilTools.replaceBlank(UtilTools.delHTMLTag(resultItems.get("content")==null?"":resultIte
ms.get("content").toString()))
                    + "\r\n";
             if(resultItems.get("url")!=null){
            FileUtils.writeStringToFile(file, str, "utf-8", true);
             }
        } catch (IOException e) {
            logger.warn("write file error", e);
        }
public static void sort(String inputpath, String outputpath) throws IOException {
        List<String> lines = FileUtils.readLines(new File(inputpath), "utf-8");
        Collections.sort(lines, new Comparator<String>() {
            @Override
            public int compare(String o1, String o2) {
                String[] split1 = o1.split("\t");
                String[] split2 = o2.split("\t");
                return split1[2].compareTo(split2[2]);
            }
        });
        FileUtils.writeLines(new File(outputpath), "utf-8", lines);
    }

    }

gwjn.html
//爬取数据
        function crawljob()
        {
        $(".bgWait").removeClass('hide');
            $.ajax({
                url : "${request.contextPath}/user/crawljob",
```

```
                type : "post",
                success : function(s) {
                    if (s.success) {
                        $(".bgWait").addClass('hide');
                        $("#textHtml").removeClass('hide');
                        showresult(1)
                    }
                }
            });
        }




    function showresult(num) {
        var data={"pageNo":num}
        $.ajax({
                    url : "${request.contextPath}/user/showcrawljob",
                    type : "post",
                    data : data,
                    success : function(s) {
                        var datas=s;
                        var tmp = "";
                        for(i=0;i<datas.length;i++){
                            var name=datas[i].name;
                            var description=datas[i].description;
                            var createtime=datas[i].createtime;
                            var curnum=(num-1)*10+i+1;
                            tmp +="<tr style='width:100%'><td
style='width:10%'>"+curnum+"</td>"+"<td style='width:20%'>"+name+"</td>"+"<td
style='width:10%'>"+createtime+"</td>"+"<td
style='width:60%'>"+description+"</td>"+"</tr>";

                        }

                        $("#textHtml").html(tmp);
                        var currentpage = num;
                        initpage(currentpage);
                    }
                })
    }
    function initpage(currentpage) {
            var totalpage=20
        $.jqPaginator('#pagination',
```

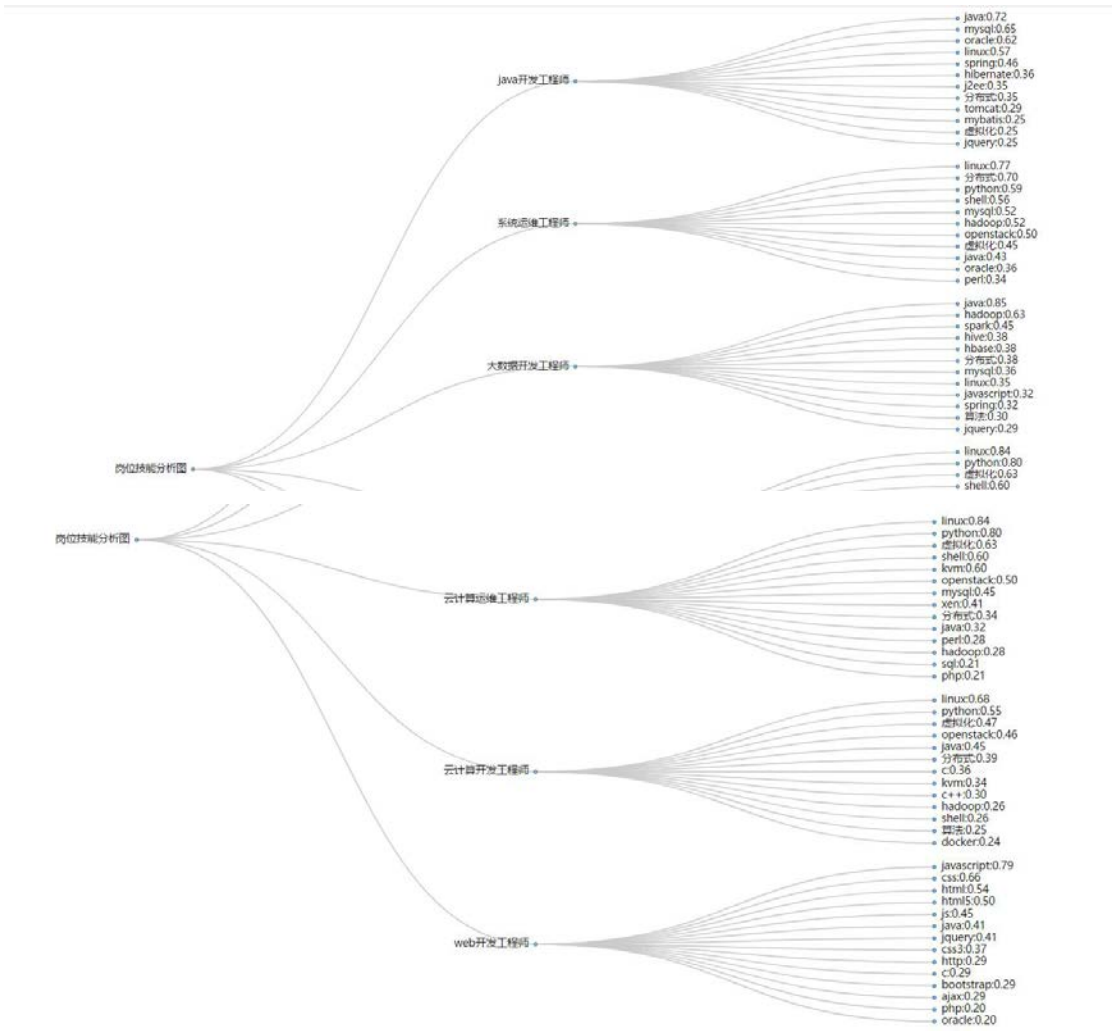
```
                                {
                                    totalPages : totalpage,
                                    visiblePages : 5,
                                    currentPage : currentpage,

                                    wrapper : '<ul class="pagination lastspan"></ul>',
                                    first : '<li class="first"><a href="javascript:void(0);">首页
</a></li>',
                                    prev : '<li class="prev"><a
href="javascript:void(0);">&laquo;</a></li>',
                                    next : '<li class="next"><a
href="javascript:void(0);">&raquo;</a></li>',
                                    last : '<li class="last"><a href="javascript:void(0);">尾页
</a></li>',
                                    page : '<li class="page"><a
href="javascript:void(0);">{{page}}</a></li>',
                                    onPageChange : function(num) {
                                        if (currentpage != num) {

                                            showresult(num);

                                        }
                                    }
                                });
    }
```

Eclipse 控制台截图（0.5 分）。
爬虫框架展示的爬取过程：“2017-06-07 11:47:23,427 INFO [pool-25-thread-3] downloader.HttpClientDownloader (HttpClientDownloader.java:download(88)) - downloading page http://localhost:8080/cloudrecomm/51job/85115163.html”
岗位列表表头名称一致（0.5 分），宽度符合要求(0.5 分)，序号从 1 开始计数日期顺序展示（发布日期早的排在前面）（0.5 分），岗位名称和岗位描述前 2 项要一致（0.5 分）。
代码 （0.5 分）

3. 聚类岗位信息（2 分）

将岗位信息序列文件/WebRoot/data/GS_jobskill.seq 上传至 HDFS，使用 mahout 的 canopy 算法生成聚类中心，再使用 kmeans 算法对岗位进行聚类。Canopy 调用参数为：余弦距离、T1 值 0.8、T2 值 0.6； kmeans 调用参数为：收敛阈值 0.5、最大迭代次数 100。使用 Mahout 的 ClusterDumper 的 API 将聚类结果输出到 txt 文件中，其中每个聚类的 TopTerms 最多输出 15 个词。使用 d3.js 的树状图可视化展示岗位技能（带权重，保留两位小数，不考虑四舍五入）、使用雷达图可视化展示聚类数量最多的岗位技能点权重信息（权重值取到小数点后两位，不考虑四舍五入）。

按顺序提交可视化呈现的岗位技能点截图、聚类数量最多的岗位技能雷达图及增改的 java 与 html 代码到答题框。




```
package com.xiandian.cloud.web;
class RecommController.java
// 针对岗位、技能点的数据聚类
    @RequestMapping("/jobskills")
```

```
    @ResponseBody
    public Object jobskills(HttpServletRequest request) {
        try {
            Configuration conf = new Configuration();
            conf.set("fs.defaultFS", UtilTools.getConfig().getProperty("hdfs"));
            HdfsClient hdfsClient = HdfsClient.getInstance();
hdfsClient.delete("/user/jobskill_in");
                // 上传到 hdfs /user/jobskill_in/目录，作为 kmeans 的入参
                File file = new File("D:/GS_jobskill.seq");
                if (file.exists()) {
                    File[] files = file.listFiles();
                    for (File file2 : files) {
                        if (!file2.isDirectory()) {
                                hdfsClient.upload(file2.getAbsolutePath(),
                                        "/user/jobskill_in/" + file2.getName());
                            }
                        }
                    }
                // kmeans 运行后产生的结果文件，下载的路径
                String dumppath = request.getSession().getServletContext()
                        .getRealPath("/dump/");
                String indumppath = dumppath + "/clusterdump.txt";
                File indumppathf = new File(indumppath);
                if (indumppathf.exists()) {
                    indumppathf.delete();
                }
                // 对接 kmeans
                ClusteringJob mahoutClient = new ClusteringJob();
                // 调用 marhoutkemans 算法，生成
                int result = mahoutClient.runKemans("/user/jobskill_in",
                        indumppath);
                // 下载结果 /user/txt-kmeans/clusters-1-final/part-r-00000
                if (result != -1) {
                    String outdumppath = dumppath + "/outclusterdump.txt";
                    File outdumppathf = new File(outdumppath);
                    if (outdumppathf.exists()) {
                        outdumppathf.delete();
                    }
                    ReadClusters rc = new ReadClusters();
                    rc.readDumpTxt(indumppath, outdumppath);
                    // 插入岗位技能点
                    jobSkillService.removeAll();// 每次插入前都清除
                    File dumpfile = new File(outdumppath);
                    InputStreamReader read = new InputStreamReader(
```

```
                        new FileInputStream(dumpfile), "utf-8");
                BufferedReader reader = new BufferedReader(read);
                String line;
                while ((line = reader.readLine()) != null) {
                    String[] strs = line.split("\t");
                    int len = strs.length;
                    String jobskills = "[";
                    for (int i = 3; i < len; i++) {
                        int index = strs[i].indexOf(":");
                        index += 5;
                        if (index<strs[i].length()){
                        jobskills += "{name:'"
                                + strs[i].substring(0, index) + "'},";
                        }else{
                            jobskills += "{name:'"
                                    + strs[i].substring(0, index-1) + "0'},";
                        }
                    }
                    jobskills = jobskills.substring(0,
                            jobskills.length() - 1);
                    jobskills += "]";
                    // 将岗位、技能点、插入数据库
                    JobSkill js = new JobSkill();
                    js.setName(strs[2]);
                    js.setJobskill(jobskills);
                    jobSkillService.save(js);
                }
                return new MessageBean(true);
            } else {
                return new MessageBean(false);
            }
        }  catch (Exception e) {
            return new MessageBean(false);
        }

    }

package com.xiandian.cloud.bigdata.analysis;
 class ClusteringJob.java
private void kemansClustering(Configuration conf, Path vectorsFolder, Path
canopyCentroids, Path clusterOutput
            ) throws Exception {

        // 运行 canopy 算法生成初始聚类中心 调用 mahout 算法库中封装好的
```

```
CanopyDriver 方法 参数给定
        CanopyDriver.run(conf, vectorsFolder, canopyCentroids,
                new CosineDistanceMeasure(), 0.8, 0.6, false, 0.0, false);

        // 运行 kmeans 算法得到聚类结果 调用 mahout 算法库中封装好的
KmeansDriver 方法 参数给定
        KMeansDriver.run(conf, vectorsFolder, new Path(canopyCentroids,
                Constants.CLUSTERS_0_FINAL), clusterOutput, 0.5, 100, true, 0.0,
                false);

public int runKemans(String inputpath, String dumppath) {
        try {
            String outputDir = Constants.KMEANS_OUTPUTDIR;
            Configuration conf = new Configuration();
            conf.set("fs.defaultFS", UtilTools.getConfig().getProperty("hdfs"));
            FileSystem fs = FileSystem.get(conf);
            HadoopUtil.delete(conf, new Path(outputDir));

            // 对序列文件进行文本---词向量形式的转换 不需要定制参数
            sequence2TFVectors(conf, inputpath, outputDir);

             //获取相关路径
            Path vectorsFolder = new Path(outputDir, Constants.TF_VECTORS);
            Path canopyCentroids = new Path(outputDir,
Constants.CANOPY_CENTROIDS);
            Path clusterOutput = new Path(outputDir, Constants.CLUSTERS);

            // 调用封装好的 Canopy+Kmeans 聚类算法
            kemansClustering(conf, vectorsFolder, canopyCentroids,clusterOutput);

            // 将聚类结果 dump 至本地 调用 mahout 算法库中封装好的 ClusterDumper
类 参数给定
            ClusterDumper clusterDumper = new ClusterDumper(new Path(

    Constants.KMEANS_OUTPUTDIR+"/"+Constants.CLUSTERS+"/"+Constants.CLUST
ERS_1_FINAL), new Path(

    Constants.KMEANS_OUTPUTDIR+"/"+Constants.CLUSTERS+"/"+Constants.CLUST
EREDPOINTS));
            String[] args = { "--input",

    Constants.KMEANS_OUTPUTDIR+"/"+Constants.CLUSTERS+"/"+Constants.CLUST
ERS_1_FINAL, "--output",
                    dumppath, "-d",
```

```
Constants.KMEANS_OUTPUTDIR+"/"+Constants.DICTIONARY, "-dt",
                        Constants.SEQUENCEFILE, "-n", Constants.DUMP_NUM};
                int result = clusterDumper.run(args);
                return result;
            } catch (Exception e) {
                e.printStackTrace();
                return -1;
            }
        }

package com.xiandian.cloud.bigdata.analysis;
 class ReadClusters.java
public class ReadClusters {
    /**
     * 解析 Kmeans 的聚类主调用程序
     *
     * @param inpath
     *                  : Kmeans 聚类后 dump 下来的文件目录
     * @param outpath
     *                  ： 输出岗位-技能点信息的文件目录
     */
    public void readDumpTxt(String inpath, String outpath) {
        String file = new String();
        file = T2S(inpath);
        String[] data = getData(file);
        writeout(data, outpath);
    }

    /**
     * 将 txt 文件读取到一个 String 中
     *
     * @param fileName
     *                  ： txt 文件名
     * @return
     */
    public String T2S(String fileName) {
        BufferedReader br = null;
        StringBuffer sb = null;
        try {
            br = new BufferedReader(new InputStreamReader(new FileInputStream(
                    fileName), "UTF-8"));
            sb = new StringBuffer();
            String line = null;
            while ((line = br.readLine()) != null) {
```

```
                sb.append(line);

            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                br.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        String data = new String(sb);
        return data;

    }

    /**
     * 解析聚类结果中的有用信息
     *
     * @param file
     *                ：保存聚类结果的 String
     * @return
     */
    public String[] getData(String file) {
        Pattern pattern1 = Pattern
                .compile("\"n\":([1-9]+[0-9]*)[^\\}]+\\}[\\s\\t]*Top
Terms:[\\s\\t]*([^\\s\\t0-9=>]+)[\\s\\t]*=>[\\s\\t]*[\\.0-9]+[\\s\\t]+([^:]*):");
        Matcher matcher1 = pattern1.matcher(file);
        int count = 0;
        while (matcher1.find()) {
            count++;
        }
        Matcher matcher11 = pattern1.matcher(file);
        String[] temp = new String[count];
        String[] num = new String[count];
        String[] jobname = new String[count];
        int c = 0;
        while (matcher11.find()) {
            temp[c] = matcher11.group(3);
            num[c] = matcher11.group(1);
            jobname[c] = matcher11.group(2);
            c++;
        }
```

```
        String[] data = new String[count];

        for (int i = 0; i < count; i++) {
            data[i] = "岗位" + (i + 1) + "\t" + num[i] + "个\t" + jobname[i];
            Pattern pattern2 = Pattern
                    .compile("([^\\s\\t=>]+)[\\s\\t]*=>[\\s\\t]*([\\.0-9]+)");
            Matcher matcher2 = pattern2.matcher(temp[i]);
            while (matcher2.find()) {
                if (!matcher2.group(1).contains("工程师")){
                data[i] = data[i] + "\t" + matcher2.group(1) + ":"
                        + matcher2.group(2);
                }
            }
        }

        return data;

    }

    /**
     * 将岗位-技能点信息按行写入 txt 文件
     *
     * @param data
     *                ：保存岗位-技能点信息的 String 数组
     * @param fileName_2
     *                ：输出 txt 文件路径
     */
    public void writeout(String[] data, String fileName_2) {

        try {
            // 创建一个文本文件
            File f = new File(fileName_2);
            // 用 FileOutputSteam 包装文件，并设置文件可追加
            OutputStreamWriter out = new OutputStreamWriter(
                    new FileOutputStream(f, true), "UTF-8");
            // 字符数组
            for (int i = 0; i < data.length; i++) {
                out.write(data[i].toCharArray());
                out.write('\r');
                out.write('\n');
            }
            out.close(); // 关闭输出流
        } catch (Exception e) {
```

```
            }
        }

}
RadarChart.html
<!DOCTYPE html>
<!--[if IE 7]><html class="ie7 lte9 lte8 lte7" lang="zh-cn"><![endif]-->
<!--[if IE 8]><html class="ie8 lte9 lte8" lang="zh-cn"><![endif]-->
<!--[if IE 9]><html class="ie9 lte9" lang="zh-cn"><![endif]-->
<!--[if gt IE 9]><!-->
<html lang="zh-cn">
<!--<![endif]-->
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta name="description" content="">
<meta name="author" content="">
<title>先电大数据开发框架</title>
<#include '../common/css.html'>
<link href="${request.contextPath}/assets/stylesheets/aliindex.css" rel="stylesheet" />
<style>
.node circle {
  fill: #fff;
  stroke: steelblue;
  stroke-width: 1.5px;
}

.node {
  font: 12px sans-serif;
}

.link {
  fill: none;
  stroke: #ccc;
  stroke-width: 1.5px;
}

.chartname{
            text-align: left;
      margin: 19px 16px 0;
      color: #333;
      font-size: 15px;
      border-bottom: 1px solid #ccc;
```

```
        padding: 0 0 7px;
}

.title{font-family:Arial,微软雅黑;font-size:18px;text-anchor:middle;}
.subTitle{font-family:Arial,宋体;font-size:12px;text-anchor:middle;fill:#666}
.axis path,
.axis line {
fill: none;
stroke: black;
shape-rendering: crispEdges;
}
.axis text {
font-family: sans-serif;
font-size: 11px;
fill:#999;
}
.inner_line path,
.inner_line line {
fill: none;
stroke:#E7E7E7;
shape-rendering: crispEdges;
}
.legend{font-size: 12px; font-family:Arial, Helvetica, sans-serif}

.hide{display:none}

a{
text-decoration:none
}
</style>
</head>
<body>
        <#include '../common/header.html'>
        <div class="main">
        <#include '../common/mainleft.html'>
        <div class="main-right">
        <div class="chartname">岗位技能图</div>
        <div class="container">
                <div class="chartresult">
                        <div class="col-xs-12">
                                <div class="box bordered-box orange-border">
                                        <div class="box-content box-no-padding">
                                                <div class="responsive-table">
                                                        <div class="dataTables_wrapper form-inline">
```

```
                                        <div class="row datatables-top mtl">
                                            <div class="col-sm-12">
                                            <div class="radarChart"
style="text-align:center"></div>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
        </div>
</div>
<div class="disk-foot"

    style="position:fixed;bottom:0;left:0 ;right:0 ;height:32px;background:#f9f9f9;border-to
p: 1px solid #e8e8e8;color: #999;text-align:center;padding:7px">
            <p>© 南京第五十五所技术开发有限公司</p>
        </div>
    <!--  JavaScript-->
        <script
src="${request.contextPath}/assets/javascripts/jquery/jquery.min.js"></script>
        <script
src="${request.contextPath}/assets/javascripts/bootstrap/bootstrap.js"></script>
        <script
src="${request.contextPath}/assets/javascripts/Validform_v5.3.2.js"></script>
        <script src="${request.contextPath}/assets/javascripts/common.js"></script>
        <script src="${request.contextPath}/assets/javascripts/d3.min.js"></script>
        <script src="${request.contextPath}/assets/javascripts/radarChart.js"></script>
        <script type="text/javascript">
        $(function() {
              $("#jobsanalyse").addClass("active");
                  $(".title_name").html("岗位分析");
        var datas=${JobSkills};
        var tmp="";
        var tmpselect="";
        for (i=0;i<datas.length;i++){
            /* var id="name"+i;
            tmpselect += "<option value="+id+">"+datas[i].name+"</option>"; */
            var name=datas[i].name;
            if(name=="云计算开发工程师"){
                var children=datas[i].children;
```

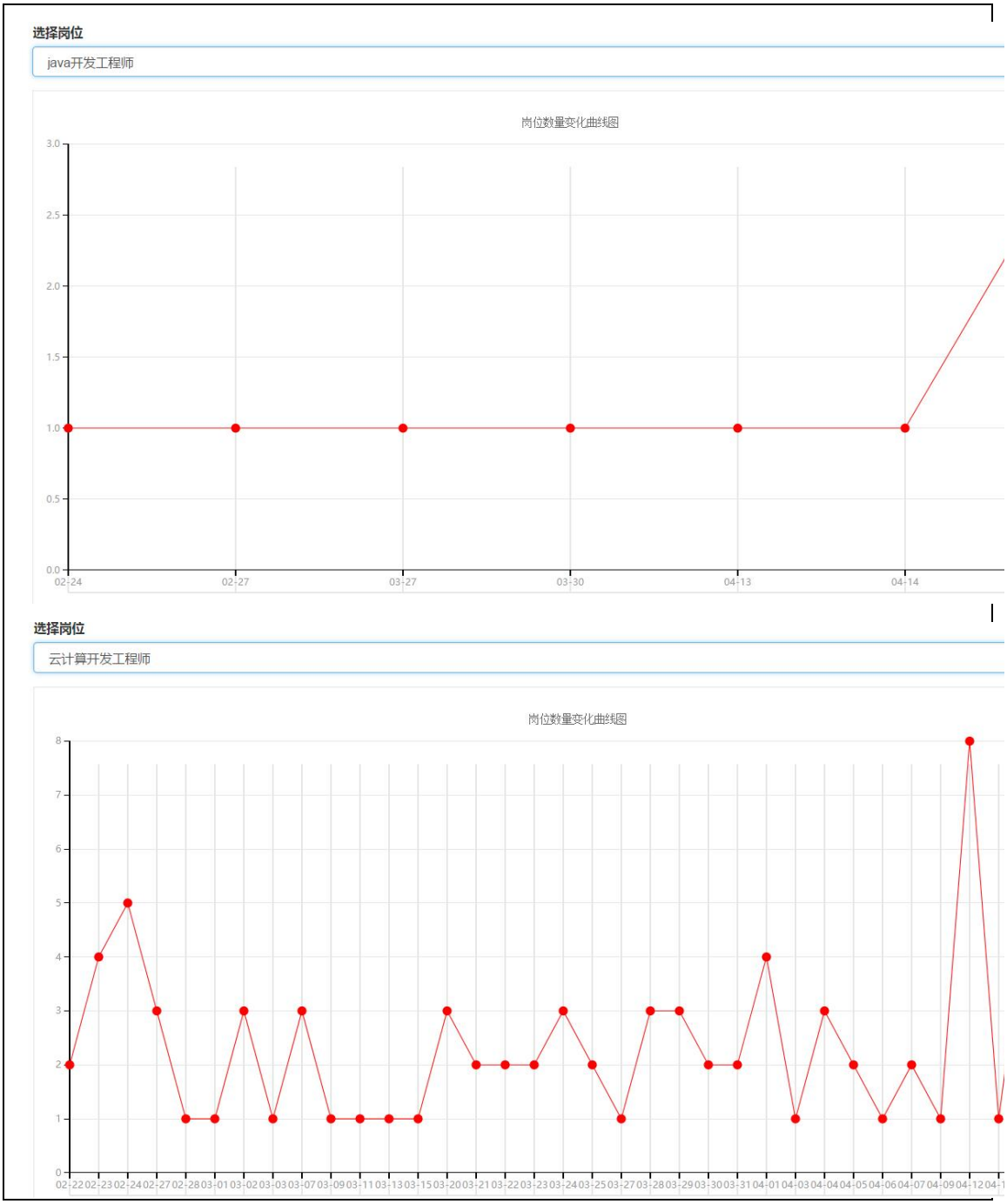
```
                var datasskill="[";
                for(j=0;j<children.length;j++){
                    var dataname=children[j].name;
                    var datanames=dataname.split(":");
                    var skillname=datanames[0];
                    var weight=datanames[1];
                    datasskill += "{axis:"'+skillname+'",value:"+weight+"},"
                }
                datasskill+="]";
                 var  data="["+datasskill+"]";
                data=eval(data);
               var color = d3.scale.ordinal()
               .range(["#EDC951","#CC333F","#00A0B0"]);

        var radarChartOptions = {
        labelFactor: 0.95,
           maxValue: 1,
           levels: 5,
           roundStrokes: true,
           color: color
        };
        RadarChart(".radarChart", data, radarChartOptions);
         }

    }

    })
    </script>
    </body>
    </html>
```

聚类 6 个岗位，tree 显示技能点（0.5 分），显示权重取 2 位小数点（0.5 分）。 “云计算开发工程师岗位”技能雷达图，数据和前面 tree 一致，百分百显示，显示数据条目同图示一致（0.5 分） 代码 （0.5 分）

4. 岗位变化曲线（2 分）

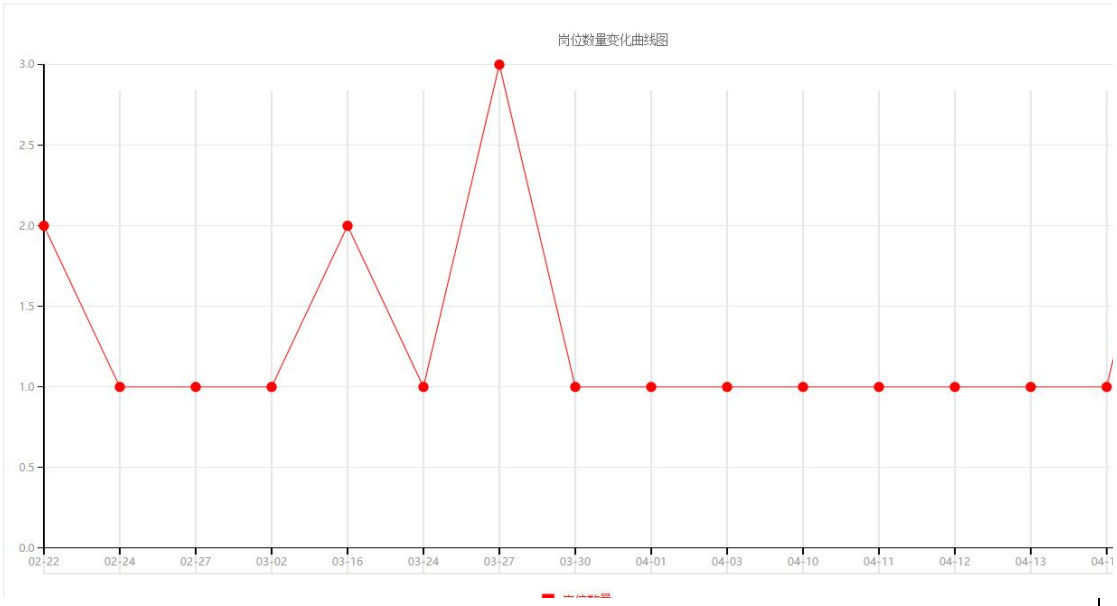
将爬取的文件/WebRoot/data/GS_jobchange.txt 上传至 HDFS，使用 CleanJobMR 程序限定的清洗规则，在 Map 过程对岗位名称进行规范，并调出其中的发布时间信息，在 Reduce 过程按照不同岗位名称与日期进行计数统计，一条岗位记录算作一个，不考虑招聘人数。使用 d3.js 折线图展示岗位需求的时间变化。

按顺序提交岗位变化曲线的截图及增改的 java 代码到答题框。


选择岗位
java开发工程师
岗位数量变化曲线图
3.0
2.5
2.0
1.5
1.0
0.5
0.0
02-24
02-27
03-27
03-30
04-13
04-14
选择岗位
云计算开发工程师
岗位数量变化曲线图
8
7
6
5
4
3
2
1
0

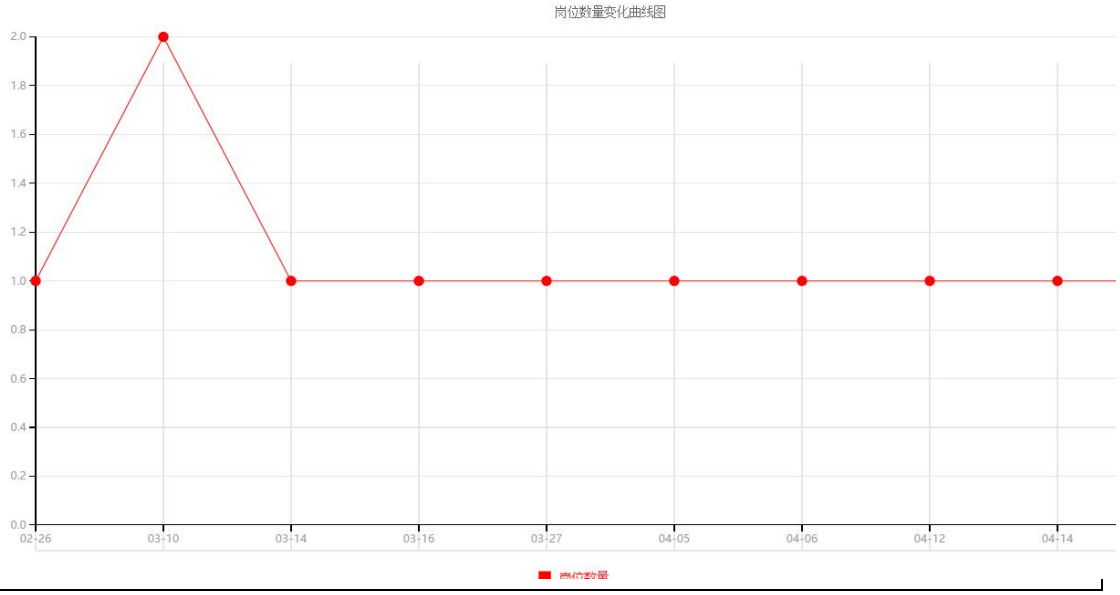
选择岗位

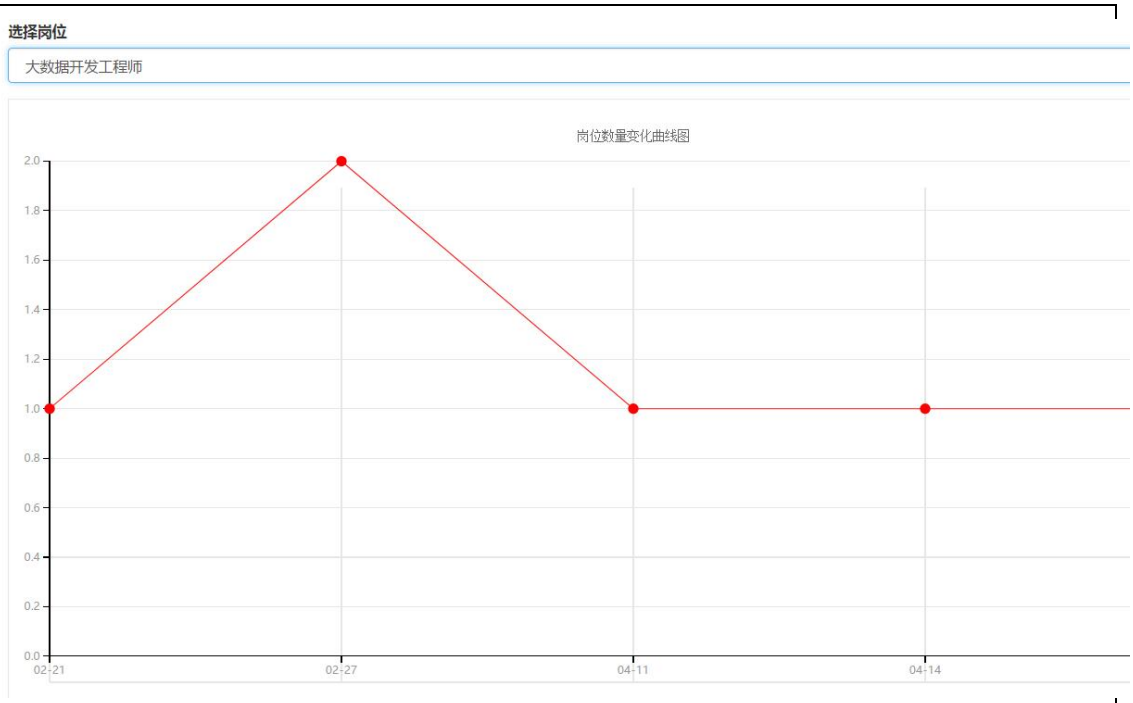
云计算架构工程师



选择岗位

云计算运维工程师






```
package com.xiandian.cloud.web;
class RecommController.java
//针对岗位变化的数据清洗
    @RequestMapping("/jobchange")
    @ResponseBody
    public Object jobchange(HttpServletRequest request) {
        try {
            Configuration conf = new Configuration();
            conf.set("fs.defaultFS", UtilTools.getConfig().getProperty("hdfs"));

            HdfsClient hdfsClient = HdfsClient.getInstance();
            hdfsClient.delete("/user/jobchange_in");
            hdfsClient.delete("/user/jobchange_out");
            File file = new File("D:/GS_jobchange.txt");

            hdfsClient.upload(file.getAbsolutePath(),"/user/jobchange_in/" +
file.getName());

            String[] ioArgs = new String[] { "/user/jobchange_in",
                    "/user/jobchange_out" };
            String[] otherArgs = new GenericOptionsParser(conf, ioArgs)
                    .getRemainingArgs();

            if (otherArgs.length != 2) {
                System.err.println("Usage: Data Deduplication <in> <out>");
                System.exit(2);
            }
```

```
        Job job = new Job(conf, "Data jobchange");
        job.setJarByClass(JobNumMR.class);
        // 设置 Map、Combine 和 Reduce 处理类
        job.setMapperClass(JobNumMapper.class);
        job.setCombinerClass(JobNumReducer.class);
        job.setReducerClass(JobNumReducer.class);
        // 设置输出类型
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(LongWritable.class);
        // 设置输入和输出目录
        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
        // 任务结束是否结束
        boolean flag = job.waitForCompletion(true);
        // 任务完成后，将输出文件下载下来，然后解析
        if (flag) {

            // 下载路径
            String path = request.getSession().getServletContext()
                    .getRealPath("/change");
            hdfsClient.downLoad("/user/jobchange_out/part-r-00000", path
                    + "/part-r-00000");

            jobChangeService.removeAll();// 每次插入前都清除

            File dumpfile = new File(path + "/part-r-00000");
            InputStreamReader read = new InputStreamReader(
                    new FileInputStream(dumpfile), "utf-8");
            BufferedReader reader = new BufferedReader(read);
            String line;
            while ((line = reader.readLine()) != null) {
                String[] strs = line.split("\t");
                JobChange jc = new JobChange();
                jc.setName(strs[0]);
                jc.setJobdate(strs[1]);
                jc.setJobcounts(strs[2]);
                jobChangeService.save(jc);
            }
            return new MessageBean(true);
        } else {
            return new MessageBean(false);
        }
    } catch (Exception e) {
```

```
                return new MessageBean(false);
            }

        }
package com.xiandian.cloud.clean;
class JobNumMR.java
public class JobNumMR {

    public static class JobNumMapper extends
            Mapper<LongWritable, Text, Text, LongWritable> {
        private Text word = new Text();

        public void map(LongWritable key, Text value, Context context) {
            try {
                String str = value.toString();
                // 简单 split，原因是前面抓取的规则就是\t 来间隔
                String[] strs = str.split("\t");
                // 对岗位名称进行规范化处理
                // 去除“产品”“经理”概念的岗位
                // 根据优先级规范化岗位名称，优先级为云计算》大数据》java》其
他小类别
                // 判断岗位是属于开发，架构还是运维。
                int count_job1 = 0;
                int count_job2 = 0;
                String temp_job = "";
                Boolean deletejob = false;
                if (strs[1].contains(Constants.JOB_1) ||
strs[1].contains(Constants.JOB_2) ){
                    deletejob=true;
                }

                if
(strs[1].contains(Constants.JOB_3)||strs[1].toLowerCase().contains(Constants.JOB_4)||strs[1].
toLowerCase().contains(Constants.JOB_5)){
                    temp_job+=Constants.JOB_6;
                    count_job1++;
                }else if (strs[1].contains(Constants.JOB_7) ||
strs[1].toLowerCase().contains(Constants.JOB_8)){
                    temp_job+=Constants.JOB_7;
                    count_job1++;
                }else if (strs[1].toLowerCase().contains(Constants.JOB_10)){
                    temp_job+=Constants.JOB_10;
                    count_job1++;
                }else if (strs[1].toLowerCase().contains(Constants.JOB_11)){
```

```
                    temp_job+=Constants.JOB_11;
                    count_job1++;
                }else if (strs[1].toLowerCase().contains(Constants.JOB_12)){
                    temp_job+=Constants.JOB_12;
                    count_job1++;
                }

                for (String i:Dictionary.dictionary_job2){
                    if (strs[1].toLowerCase().contains(i)){

                        temp_job += i;
                        count_job2++;
                    }

                }
                if (count_job2 == 0){
                    temp_job+=Constants.JOB_13;
                }else{temp_job+=Constants.JOB_14;}
                // 保留岗位名称和招聘时间
                if (!deletejob && count_job1 == 1 && count_job2 <= 1) {
                    word.set(temp_job + "\t" + strs[8].substring(0, strs[8].length() - 2));
                    context.write(word, new LongWritable(1));
                }
            } catch (Exception e) {
                e.printStackTrace();
            }

        }
    }

    /**
     *
     * 将 Map 数据进行 Reduce 聚合
     *
     */
    public static class JobNumReducer extends
            Reducer<Text, LongWritable, Text, LongWritable> {

        public void reduce(Text key, Iterable<LongWritable> values,
                Context context) {
            try {
                Long sum = 0L;
                for (LongWritable times : values) {
                    sum += times.get();
```

```
                    }
                    context.write(key, new LongWritable(sum));
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        }
}
```

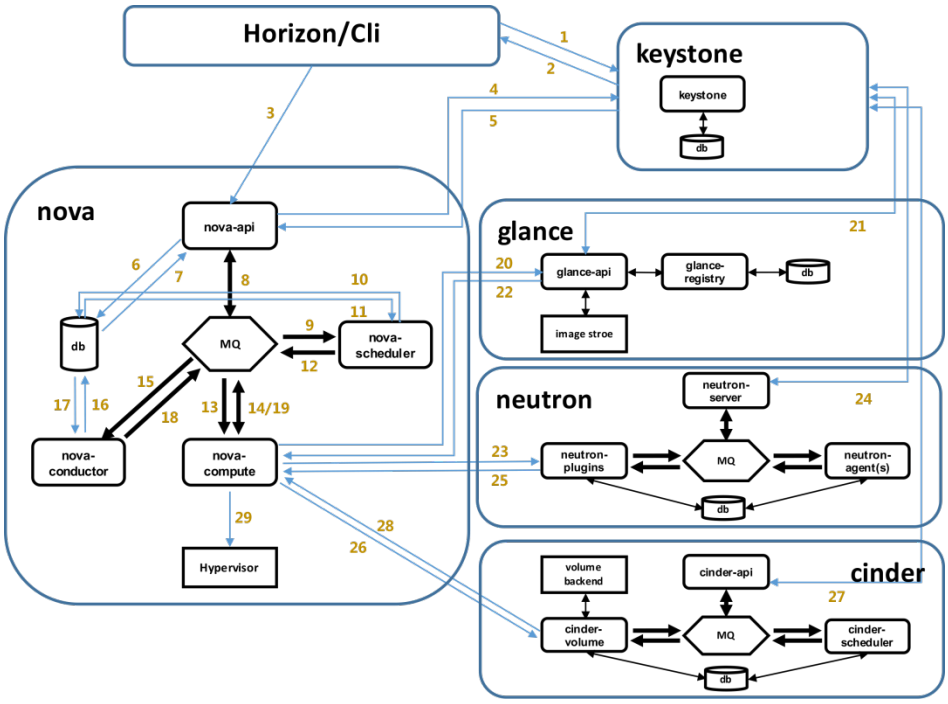
云计算开发工程师的时间变化曲线(0.5 分)，折现变化趋势和参考图一致。(1 分)
实现代码（0.5 分）。

第五部分：工作总结报告（5 分）

1. 云平台架构设计和说明（2 分）

绘制 IaaS OpenStack 云平台新建云主机的流程图，并简述云主机创建的过程。要求流程从 dashboard 中进行创建，使用 keystone 作为所有组件的权限认证，nova 作为计算服务，neutron 提供网络，由 cinder 提供持久化存储。

- Openstack 新建云主机流程图



虚拟机创建过程：

1. 界面或命令行通过 RESTful API 向 keystone 获取认证信息。
2. keystone 通过用户请求认证信息，并生成 auth-token 返回给对应的认证请求。
3. 界面或命令行通过 RESTful API 向 nova-api 发送一个 boot instance 的请求（携带 auth-token）。

4. nova-api 接受请求后向 keystone 发送认证请求，查看 token 是否为有效用户和 token。 5. keystone 验证 token 是否有效，如有效则返回有效的认证和对应的角色（注：有些操作需要有角色权限才能操作）。 6. 通过认证后 nova-api 和数据库通讯。 7. 初始化新建虚拟机的数据库记录。 8. nova-api 通过 rpc.call 向 nova-scheduler 请求是否有创建虚拟机的资源(Host ID)。 9. nova-scheduler 进程侦听消息队列，获取 nova-api 的请求。 10. nova-scheduler 通过查询 nova 数据库中计算资源的情况，并通过调度算法计算符合虚拟机创建需要的主机。 11. 对于有符合虚拟机创建的主机，nova-scheduler 更新数据库中虚拟机对应的物理主机信息。 12. nova-scheduler 通过 rpc.cast 向 nova-compute 发送对应的创建虚拟机请求的消息。 13. nova-compute 会从对应的消息队列中获取创建虚拟机请求的消息。 14. nova-compute 通过 rpc.call 向 nova-conductor 请求获取虚拟机消息。（Flavor） 15. nova-conductor 从消息队队列中拿到 nova-compute 请求消息。 16. nova-conductor 根据消息查询虚拟机对应的信息。 17. nova-conductor 从数据库中获得虚拟机对应信息。 18. nova-conductor 把虚拟机信息通过消息的方式发送到消息队列中。 19. nova-compute 从对应的消息队列中获取虚拟机信息消息。 20. nova-compute 通过 keystone 的 RESTfull API 拿到认证的 token，并通过 HTTP 请求 glance-api 获取创建虚拟机所需要镜像。 21. glance-api 向 keystone 认证 token 是否有效，并返回验证结果。 22. token 验证通过，nova-compute 获得虚拟机镜像信息(URL)。 23. nova-compute 通过 keystone 的 RESTfull API 拿到认证 k 的 token，并通过 HTTP 请求 neutron-server 获取创建虚拟机所需要的网络信息。 24. neutron-server 向 keystone 认证 token 是否有效，并返回验证结果。 25. token 验证通过，nova-compute 获得虚拟机网络信息。 26. nova-compute 通过 keystone 的 RESTfull API 拿到认证的 token，并通过 HTTP 请求 cinder-api 获取创建虚拟机所需要的持久化存储信息。 27. cinder-api 向 keystone 认证 token 是否有效，并返回验证结果。 28. token 验证通过，nova-compute 获得虚拟机持久化存储信息。 29. nova-compute 根据 instance 的信息调用配置的虚拟化驱动来创建虚拟机。
1、有流程图得 0.5 分 2、流程图中包含 keystone、nova、neutron、cinder 组件得 0.5 分 3、流程图中各组件之间有对应连线得 0.5 分 4、对各对应连线进行说明，得 0.5 分

2. 运维脚本编写（3 分）

编写 shell 脚本文件 swift-muli.sh，实现 swift 云存储账号的批量创建和容器的批量删除，要求如下：

（1）使用命令“/bin/bash swift-muli.sh create keystoneuser.txt”批量创建用户账号，账号名称及密码读取自附件文件 keystoneuser.txt，文件中第一列为 keystone 云存储账号，第二列为对应的密码。云存储账号的项目名与账号名称相同，账号在项目中拥有“user”权限，并使用云存储账号名创建容器。

（2）使用命令“/bin/bash swift-muli.sh deletecontainer keystoneuser.txt”批量删除用户容器，删除 keystoneuser.txt 中云存储账号名下的所有容器。

提交 shell 脚本 swift-muli.sh 的全部文本内容及执行批量创建用户后查询用户列表的截图到答题框。

```
[root@controller ~]# cat swift-muli.sh

#!/bin/bash

source /etc/keystone/admin-openrc.sh

if [[ $1 = "create" ]];

then

cat $2 | while read line

do

user=$(echo $line | awk '{print $1}')

pass=$(echo $line | awk '{print $2}')

openstack user create    --password $pass --domain demo $user

openstack project create --domain demo $user

openstack role add    --user $user --project $user user

swift --os-auth-url http://controller:5000/v3 --auth-version 3 --os-project-name $user
--os-project-domain-name demo --os-username $user --os-user-domain-name demo
--os-password    $pass post $user

done

elif [[ $1 = "deletecontainer" ]];

then

cat $2 | while read line

do
```

```
user=$(echo $line | awk '{print $1}')

pass=$(echo $line | awk '{print $2}')

swift --os-auth-url http://controller:5000/v3 --auth-version 3 --os-project-name $user
--os-project-domain-name demo --os-username $user --os-user-domain-name demo
--os-password    $pass delete --all

done

else

echo "Error:Please use create/deletecontainer to create/delete user's container !"

fi
```

```
+----------------------------------+-------------------+
| ID                               | Name              |
+----------------------------------+-------------------+
| 020cce9b69a44aef984fa8aba5135f3f | heat_domain_admin |
| 1cc5496471a842aabec2588e520ce7bc | user001           |
| 2c8b1e282f804fc58bc8cdfe7518d80f | neutron           |
| 350701e038d9428aa8f2197e78a90a66 | adminuser         |
| 372c371d6c4a4d5aa099376499fa21ef | aodh              |
| 413c222a0c0d476fa77e3a096f9da216 | heat              |
| 46faed53c3534611a83f586d55951246 | glance            |
| 53a1cf0ad2924532aa4b7b0750dec282 | admin             |
| 73191a8367e4410da4bf764263ade04c | demo              |
| 95f197fef75a490cbe926350d2688512 | cinder            |
| 97dcfcb9a3b84dd5a47d40c01425e8f5 | swift             |
| a0024f4b405745b999aaa1276844dec6 | nova              |
| b61702b3ffe247cd86908cb009bb7629 | testuser          |
| d46602d109214aea92ec4f9f0e4223e2 | ceilometer        |
| eba70457c243485f9924f064b3f85f19 | user002           |
+----------------------------------+-------------------+
```

1、脚本存在 create 和 deletecontainer 的判断语句得 0.5 分
2、脚本存在对外部文件的读取方面的语句得 0.5 分
3、脚本存在创建账号和容器的命令得 0.5 分
4、脚本存在删除容器的命令得 0.5 分
5、截图中包含 testuser1、user001、user002、adminuser 且表格格式相同得 1 分

第六部分：职业素养（5 分）

依工作作风、安全意识、团队协作和遵守考场纪律情况由裁判现场判分。