

## ORACLE MANAGEMENT CLOUD: DATABASE AGENT DEPLOYMENT / MONITORING:

### Details:

In this lab, you will create a database, connect to it, modify it, and prepare it to be monitored using *Oracle Management Cloud*. To do this, we will configure a *Cloud Agent* to the host machine supporting the instance of *Oracle Database*, and then relate the agent back to an instance of *Oracle Management Cloud* to do monitoring. It is critical to carefully follow the steps outlined in this guide to accomplish this. With that, let's get started!

### Prerequisites:

There are some key components you'll need to have throughout the course of this lab. I'm going to go through this lab assuming you're using a Windows machine, but I will explain the simple changes to make if you're using Linux. Since we're going to be moving files to remote hosts, we're going to need an SCP client (such as WinSCP or PSCP). We're also going to an SSH client (my client of choice is PuTTY). And of course, you'll need an Oracle Cloud Account.

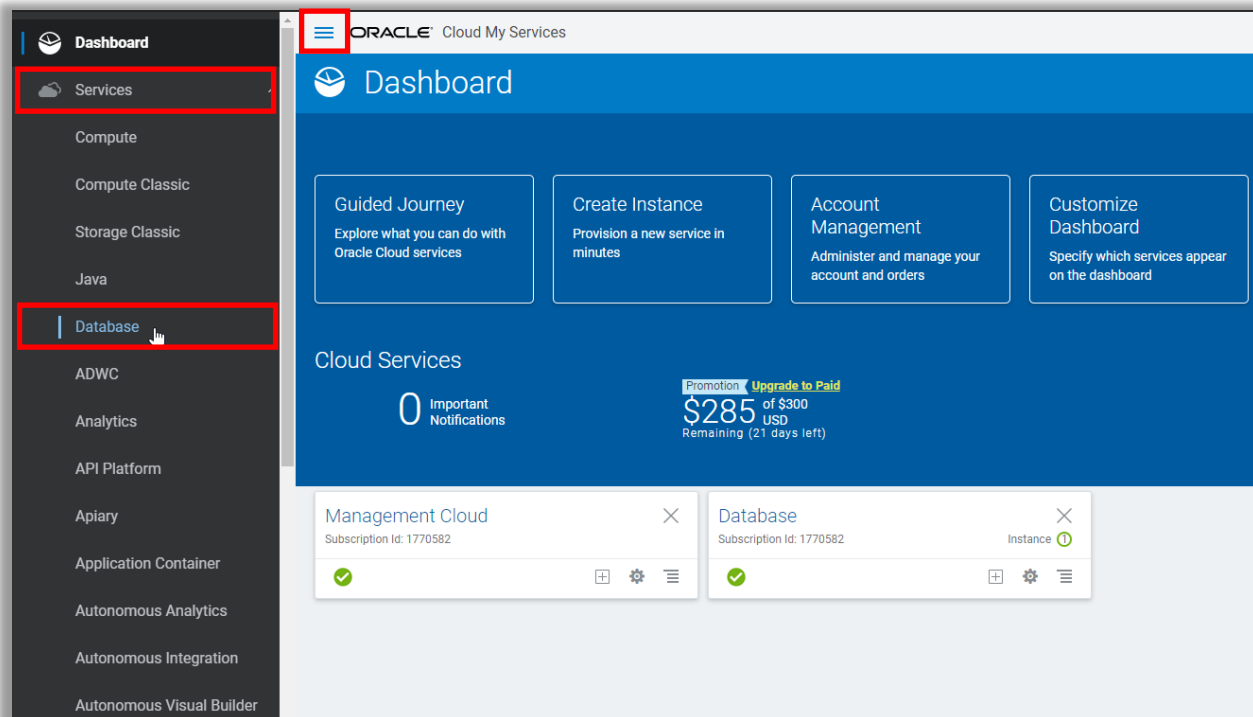
### Pre-Lab Exercise:

Download: PuTTY, PuTTY Gen, PSCP. (You can download these [here](#))

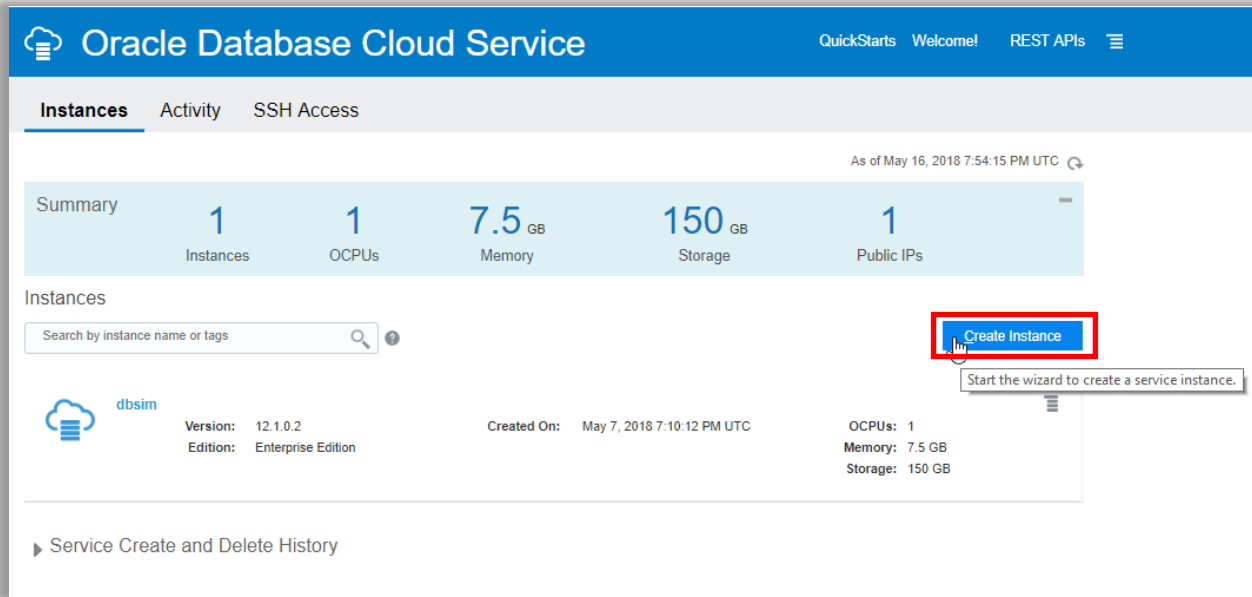
Make SSH keys with PuTTY Gen. (You can find a tutorial [here](#)). If you have any trouble with creating the SSH Keys, reach out to [zach.hamilton@oracle.com](mailto:zach.hamilton@oracle.com) for help.

### Creating an instance of Oracle Database Cloud Service:

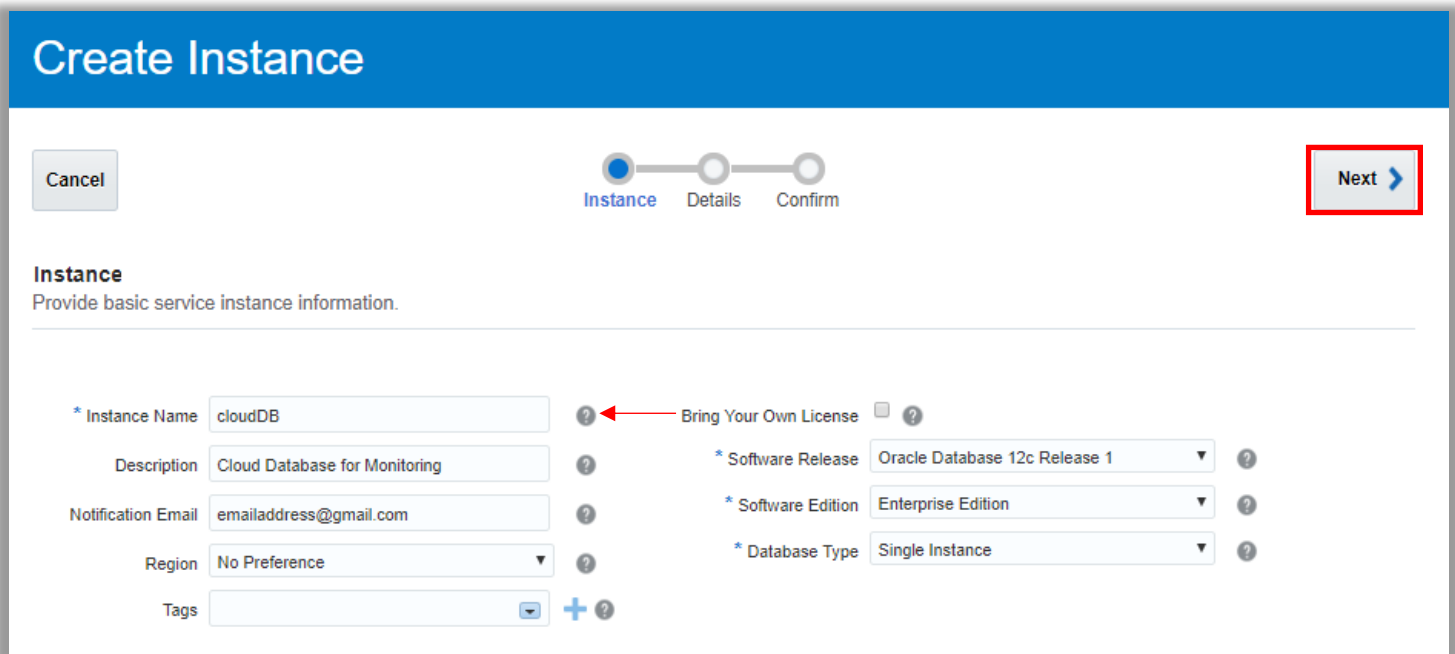
Step 1: From the My Services Dashboard, select the "Action Menu" at the top-left of the page to pop-out the window. Click on the "Services" dropdown and find and click "Database".



Step 2: On the Database Cloud Service console, select “Create Instance” to create a new instance of Oracle Database Cloud Service.



Step 3: Configure the correct options for the database instance you’re going to create. This page only *requires* you to provide a database name. For simplicity and consistency throughout this lab, use “cloudDB”. **Leave the other options unchanged.** When the options are added, click “Next”. (Note: provide a valid email address to get notifications about service maintenance, availability, and more by email about your instance).



Step 4: On this page you’ll need to configure just a few options. The first is the administrator password. **Take note of this password because you will need to use it later.** After you’ve entered an administrator password, configure the *Public Key* of the SSH keypair you made in the *Pre-Lab Setup*. To do this, click “Edit”, then the radio button “Key Value”, then paste the contents of your *Public Key* into the text box, then click “Enter”. Next, select the “Backup Destination” dropdown and select “None”. When you’ve done that, click “Next” to proceed.

# Create Instance

[< Previous](#) [Cancel](#)

Instance Details Confirm

[Next >](#)

## Instance Details

Provide details for this Oracle Database Cloud Service instance.

[Selection Summary](#)

### Database Configuration

\* DB Name  ?

\* PDB Name  ?

\* Administration Password  ?

\* Confirm Password  ?

\* Usable Database Storage (GB)  ?

Total Data File Storage (GB)  ?

\* Compute Shape  ?

\* SSH Public Key  [Edit](#) ?

Use High Performance Storage ☐ ?

[Advanced Settings](#)

### Backup and Recovery Configuration

\* Backup Destination  ?

### Initialize Data From Backup

\* Create Instance from Existing Backup  ?

Step 5: On the next page, you will have a summary of all the information from the database you're about to create. It should look like such:

# Create Instance

[< Previous](#)[Cancel](#)[Create >](#)

## Confirmation

Confirm your responses and create service instance.

### Instance

Instance Name:	cloudDB
Description:	
Bring Your Own License:	No
Service Level:	Oracle Database Cloud Service
Metering Frequency:	Hourly
Software Release:	Oracle Database 12c Release 1
Software Edition:	Enterprise Edition
Compute Shape:	OC3 - 1.0 OCPU, 7.5 GB RAM
SSH Public Key:	ssh-rsa AAAAB3NzaC1yc2EAA...
Use High Performance Storage:	No
Assign Public IP:	Yes

### Backup and Recovery Configuration

Backup Destination:	None
---------------------	------

### Notification

Notification Email:	zdamilt.spam@gmail.com
---------------------	------------------------

### Database Configuration

DB Name:	ORCL
PDB Name:	PDB1
Usable Database Storage (GB):	25
Total Data File Storage (GB):	88.5
Listener Port:	1521
Timezone:	(UTC) Coordinated Univers...
Character Set:	AL32UTF8 - Unicode Univer...
National Character Set:	AL16UTF16 - Unicode UTF-1...
Include "Demos" PDB:	No
Include GoldenGate:	No

### Standby Database Configuration

Standby Database with Data Guard:	No
-----------------------------------	----

Once you see this page, you're ready to create the instance. Click "Create" to proceed. You should be brought back to the Database Cloud Service summary page and see the new instance being created.

Oracle Database Cloud Service

QuickStartsWelcome!REST APIs

InstancesActivitySSH Access

As of May 16, 2018 8:21:44 PM UTC

Summary

2Instances

2OCPUs

15GBMemory

150GBStorage

1Public IPs

Instances

Search by instance name or tags

Create Instance

cloudDB

Status:Creating instance ...

Version:12.1.0.2

Edition:Enterprise Edition

Submitted On:May 16, 2018 8:21:33 PM UTC

OCPUs:1

Memory:7.5 GB

Storage:

dbsim

Version:12.1.0.2

Edition:Enterprise Edition

Created On:May 7, 2018 7:10:12 PM UTC

OCPUs:1

Memory:7.5 GB

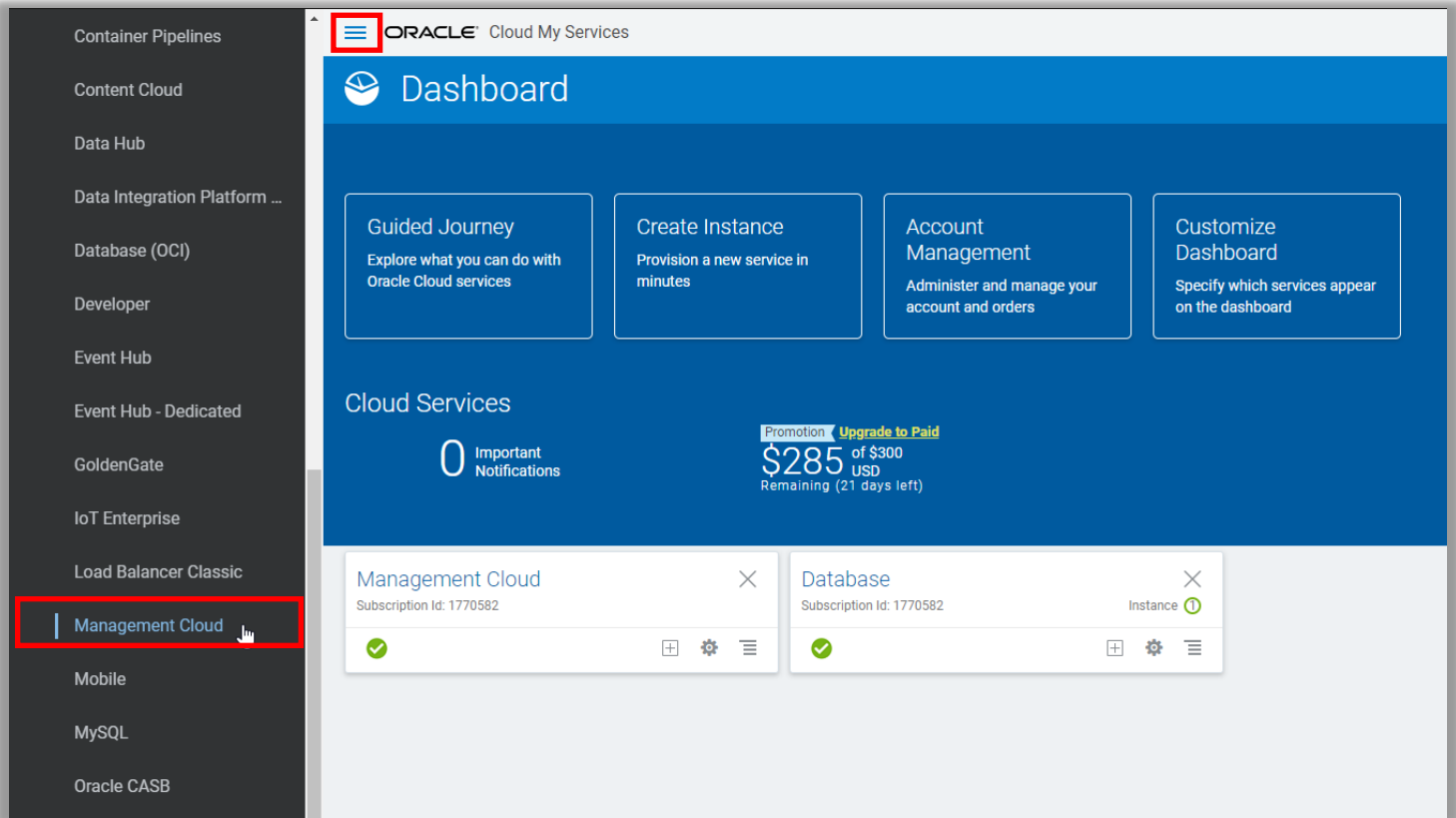
Storage:150 GB

Service Create and Delete History

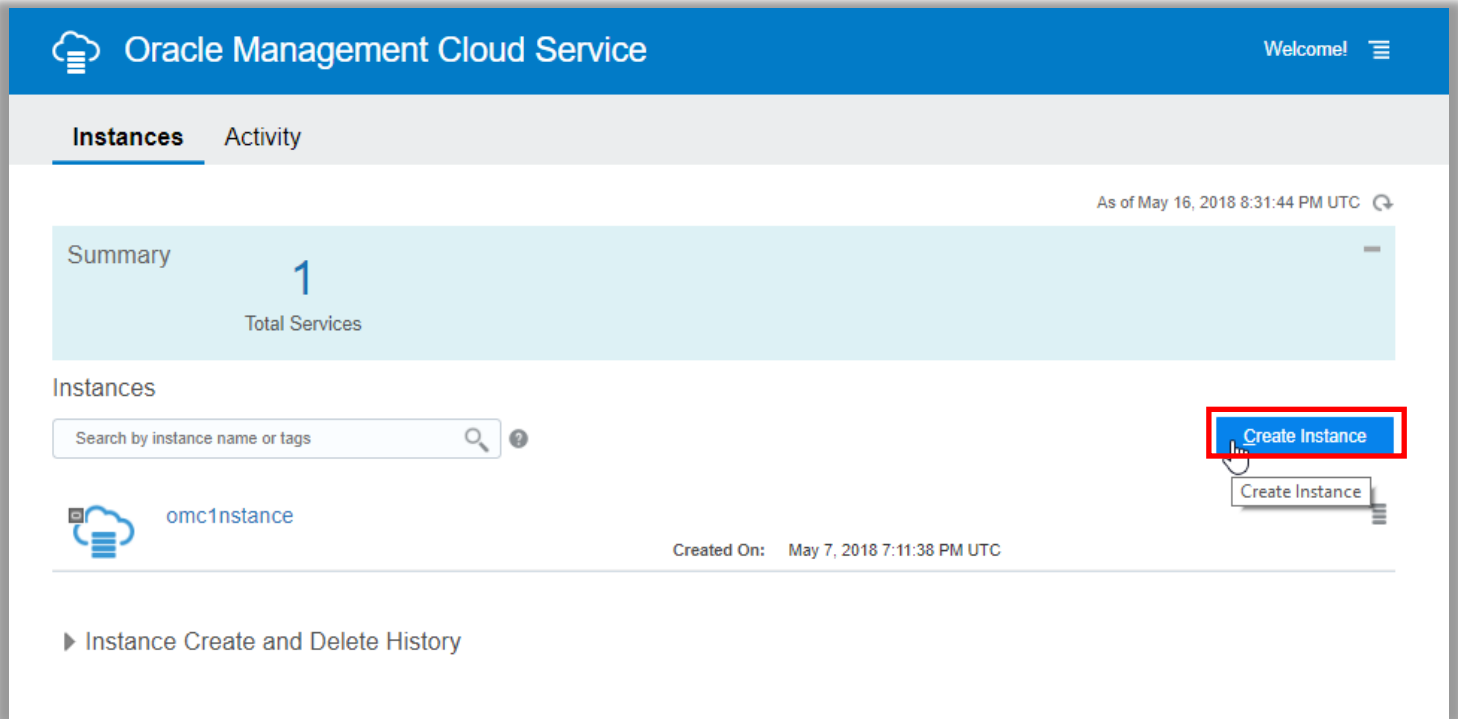
Once you see this page, you're done with Database Cloud Service for a while. Remember how you got to this console because we will be coming back later in the lab. While the database is provisioning, we'll create an instance of Oracle Management Cloud.

### Creating an instance of Oracle Management Cloud:

Step 1: From the My Services Dashboard, once again, click the "Action Menu" at the top-left of the screen to pop-out the window. After the window is popped out, click the "Services" dropdown. When the services dropdown is open, scroll until you find "Management Cloud" and select it.



Step 2: Once on the Management Cloud Service console, find and select the “Create Instance” button and select it.



Step 3: On the Create New Instance page, name the instance you are creating. I named my instance “omc2nstance”. You can name this instance anything you would like. Once you have named your instance, click “Next”.

## Create New Instance

Instance

Confirm

### Instance

Provide basic service instance information

---

#### Details

\* Instance Name

omc2nstance

?

Description

Management Cloud Instance

?

Notification Email

emailaddress@gmail.com

?

\* Region

No Preference

?

Tags

+

?

Step 4: Next you should see the summary page for the instance you are about to create. Review the details and then click “Create”.

## Create New Instance

< Previous

Cancel

Instance

Confirm

### Confirmation

Confirm your responses and create this Oracle Management Cloud Service Instance

---

#### Service

Instance Name:

omc2nstance

Description:

Management Cloud Instance

Notification Email:

emailaddress@gmail.com

Software Release:

OMCS Release 3.0

Metering Frequency:

HOURLY

Region:

No Preference

You should be brought back to the Management Cloud Service console page and see the new instance you just created being provisioned.

Oracle Management Cloud Service

Welcome!

**Instances** Activity

As of May 16, 2018 8:39:03 PM UTC

Summary

2  
Total Services

Instances

Search by instance name or tags

Create Instance

	omc2nstance	Status: Creating service ...	Submitted On: May 16, 2018 8:38:57 PM UTC
	omc1nstance		Created On: May 7, 2018 7:11:38 PM UTC

► Instance Create and Delete History

We're going to need the instance to be live before we can use it. You're ready to move to the next step.

### Connecting to an instance of Database Cloud Service:

Step 1: Navigate to Database Cloud Service console (from the My Services Dashboard, click the pop-out menu on the left and then select Database under Services). Once you are at the console, click on the database instance you created ("cloudDB"). You want to get the instance details page, which looks like this:



The screenshot displays the Oracle Cloud DB console interface. The top navigation bar is blue with the 'cloudDB' logo and a hamburger menu icon. Below the navigation bar, the breadcrumb 'Oracle Database Cloud Service / cloudDB' is visible. The main content area is titled 'Instance Overview' and includes a timestamp 'As of May 21, 2018 7:05:41 PM UTC'. A summary card shows '1 Node', '1 OCPUs', '7.5 GB Memory', and '150 GB Storage'. Below this, the 'Status' is 'Ready' and the 'Connect String' is 'cloudDB:1521/PDB1.601701664...'. The 'Edition' is 'Enterprise Edition'. Other details include 'Backup Destination: None', 'PDB Name: PDB1', 'Character Set: AL32UTF8 - Unicode Univer...', 'SQL \*Net Port: 1521', 'Container Name: ORCL', 'National Character Set: AL16UTF16 - Unicode UTF-1...', and 'Timezone: Coordinated Universal Time'. A 'Resources' section at the bottom shows 'Host Name: cloudDB', 'Public IP: 129.150.64.50', and 'SID: ORCL'. Red boxes highlight the 'Connect String' and the 'Resources' information.

Overview

1 Node

Administration

1 Patches available

0 Snapshots available

Instance Overview

As of May 21, 2018 7:05:41 PM UTC

1 Nodes

1 OCPUs

7.5 GB Memory

150 GB Storage

Status: Ready

Connect String: cloudDB:1521/PDB1.601701664... cloudDB:1521/PDB1.601701664.oraclecloud.internal

Edition: Enterprise Edition

Backup Destination: None

PDB Name: PDB1

Character Set: AL32UTF8 - Unicode Univer...

SQL \*Net Port: 1521

Container Name: ORCL

National Character Set: AL16UTF16 - Unicode UTF-1...

Timezone: Coordinated Universal Time

Show less...

Resources

Host Name: cloudDB

Public IP: 129.150.64.50

SID: ORCL

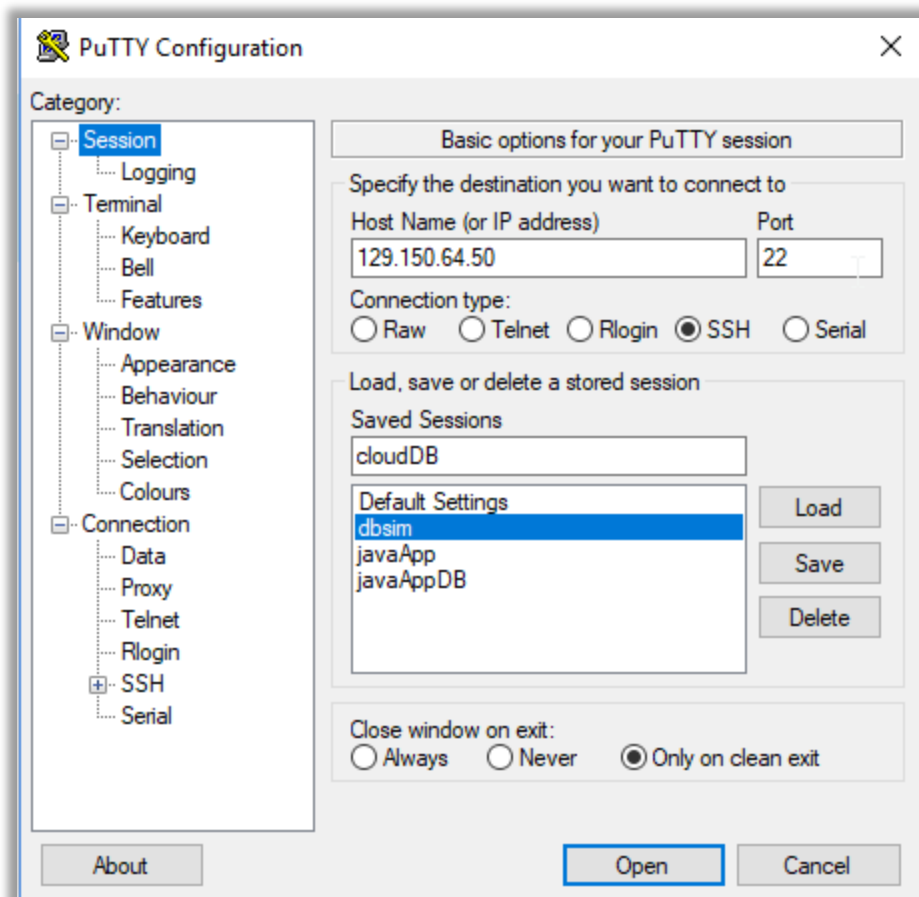
OCPUs: 1

Memory: 7.5 GB

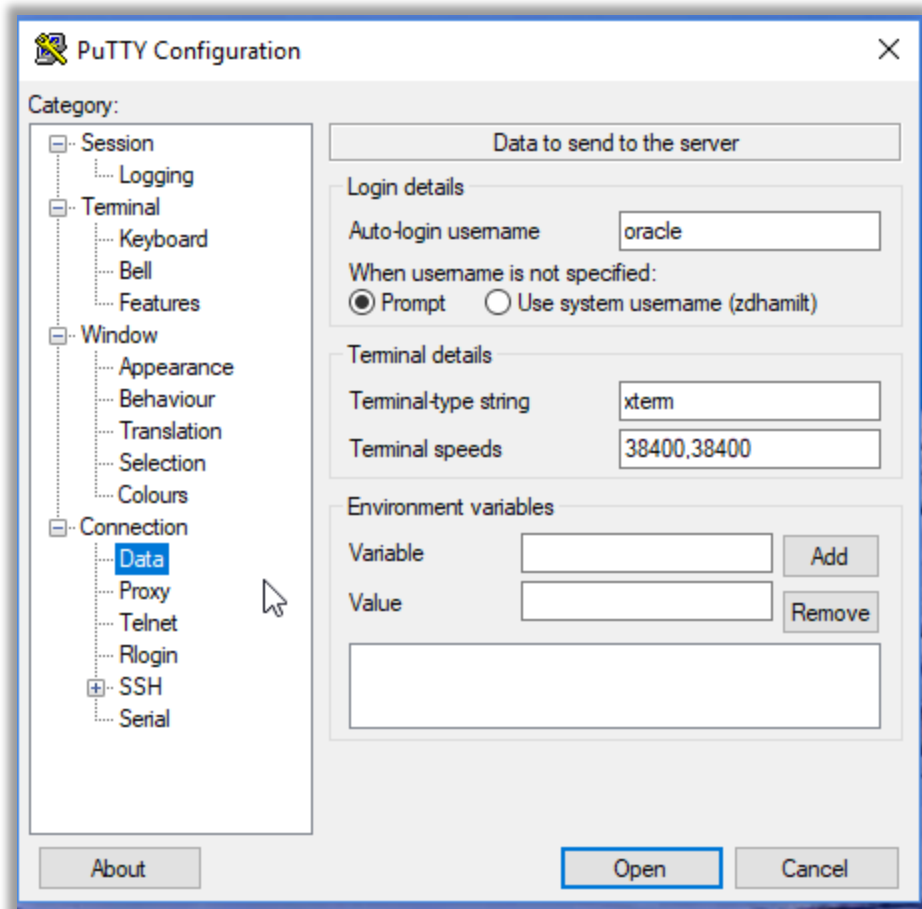
Storage: 150 GB

Step 2: Take some very important information down from this page. We're going to want the Public IP, SID, Host Name, and Connect String (see above). I usually keep open a window with Notepad and store the information there.

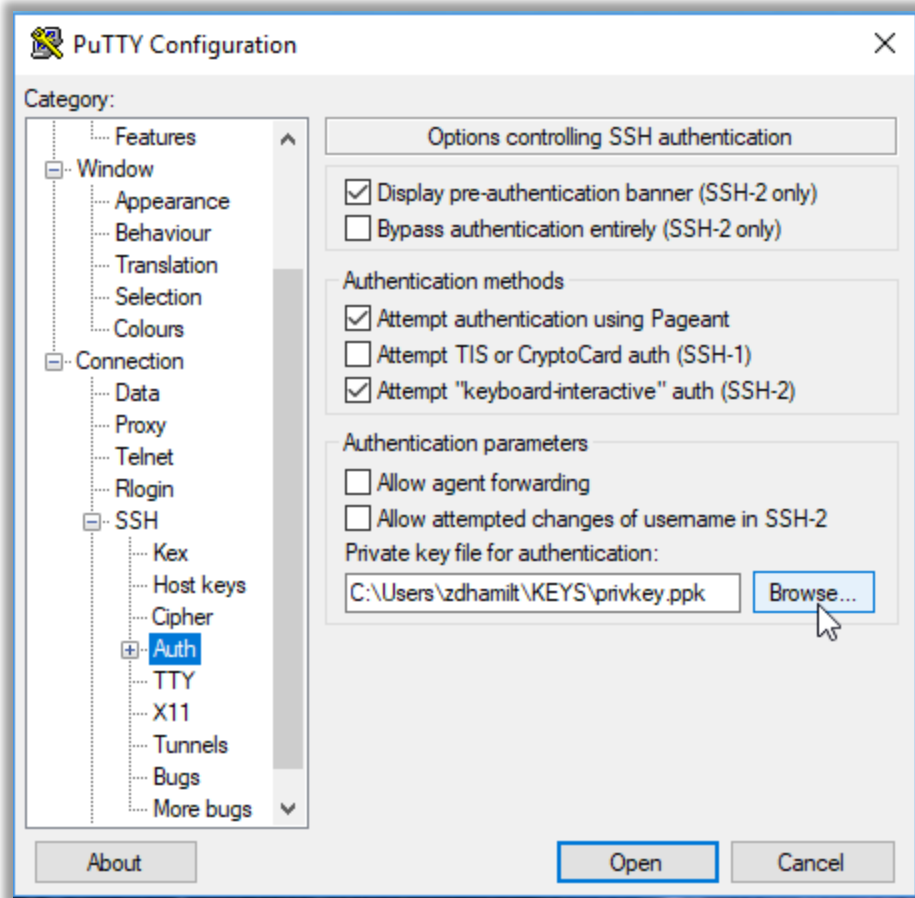
Step 3: Open a new PuTTY session. We're going to add three pieces of important information (Public IP, Auto-login username, and the private encryption key). The first thing we will do is name the settings we're configuring in the "Session > Saved Sessions" field. As you can see, I'm naming mine "cloudDB". Next, add the public IP address in the "Session > Host Name (or IP address)" field. Both can be seen below:



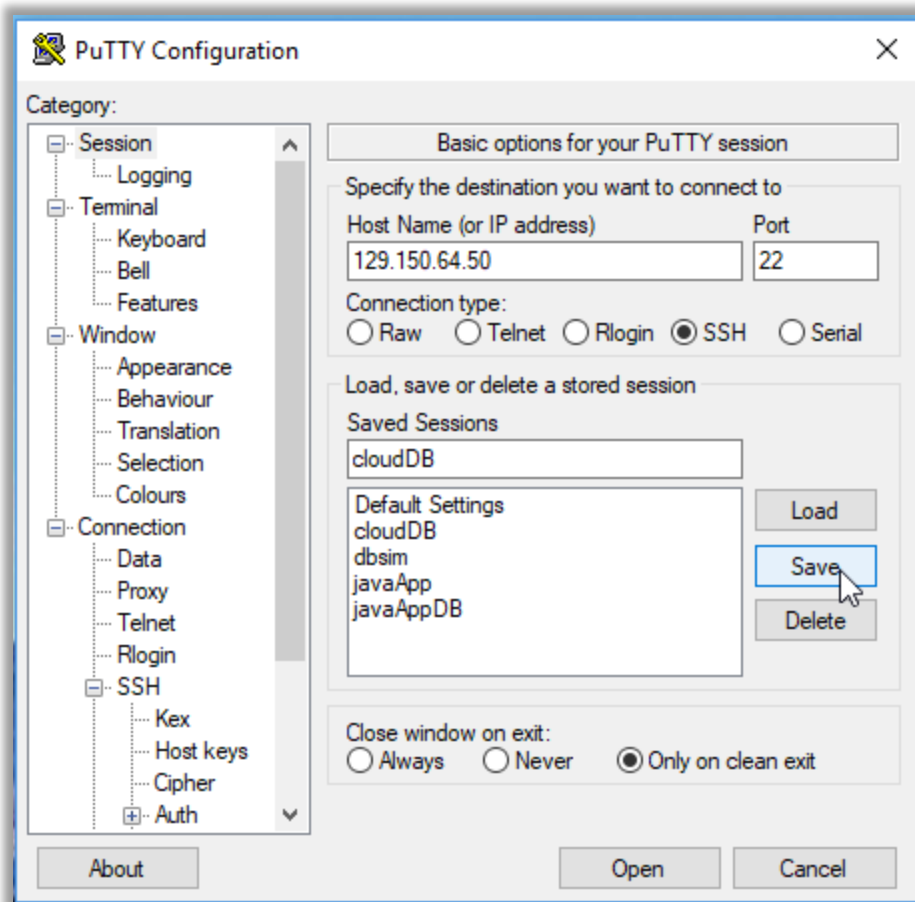
Next, we're going to go to the "Connection > Data > Auto-login username" field and enter "oracle". See below:



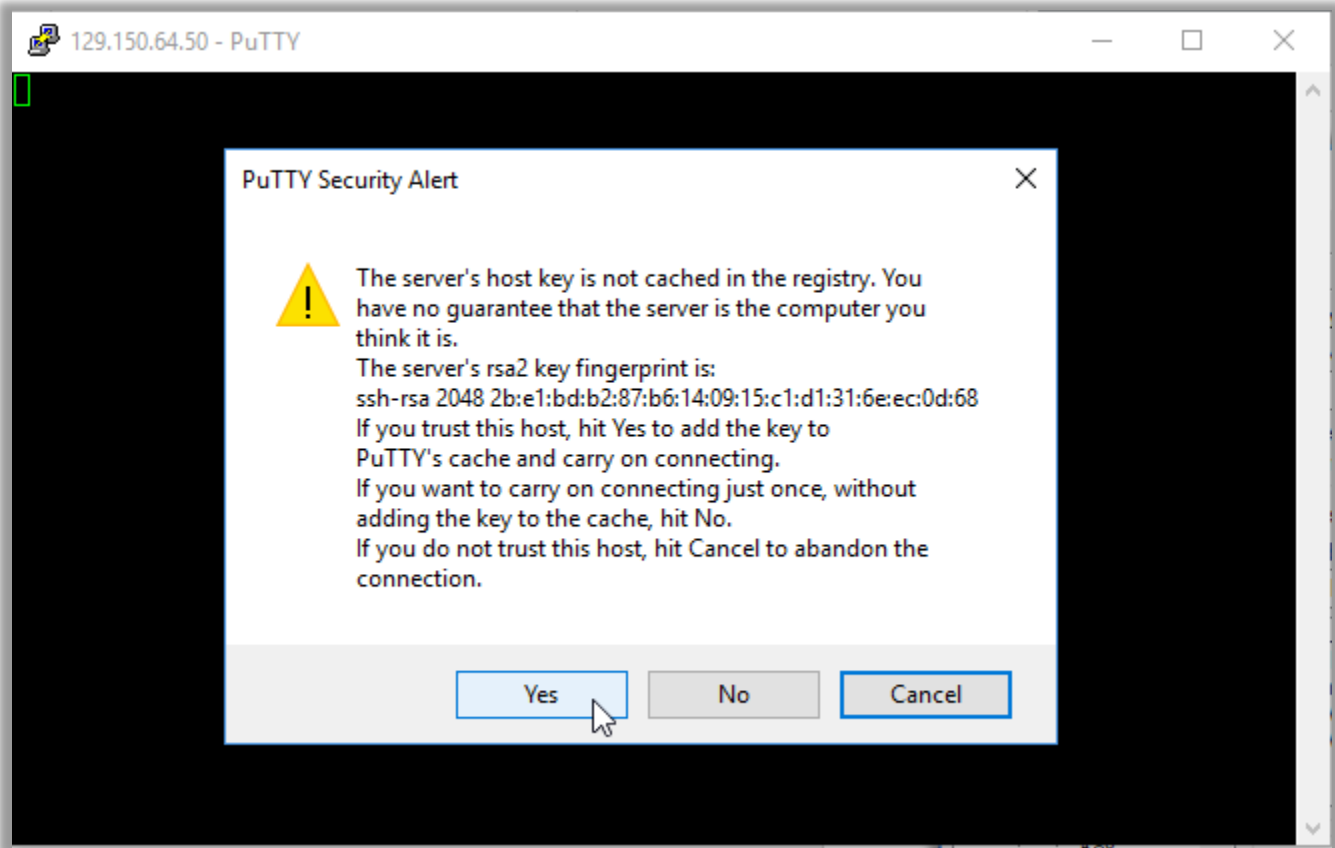
Once you've done that, navigate to "Connection > Expand 'SSH' > Auth > Private key file for authentication". Select the "Browse" button next to the field and find and select your private key file. See below:



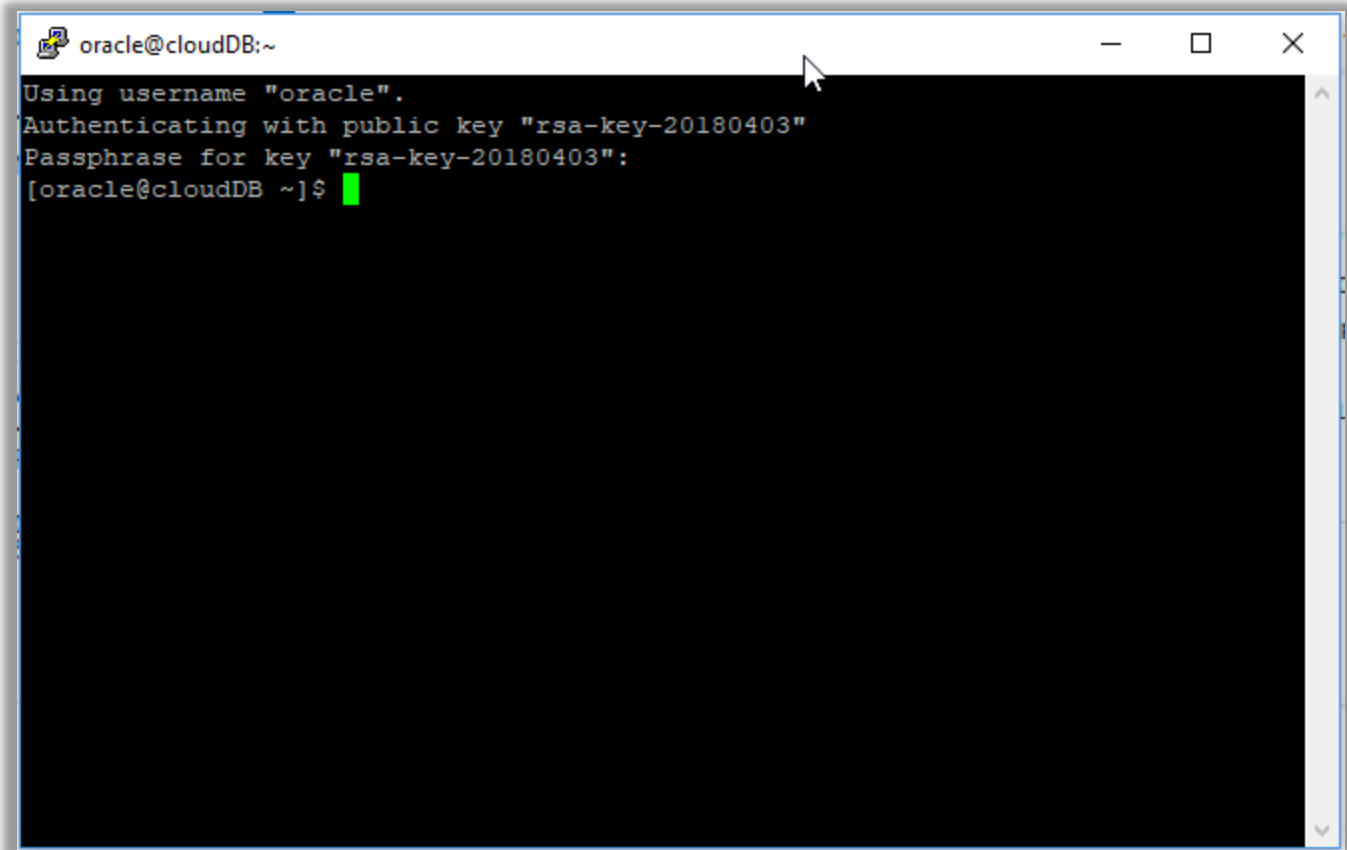
Now that you have all the settings for the connection configured, go back to the “Session” tab. We want to save all these settings so we can load them anytime we come back and want to create this same session. Click on “Save” as seen below.



Once you have saved the settings, select “Open”. You should get a warning (see below) that the RSA key fingerprint is not cached. Select “Yes” to continue. You can expect to not see this warning again after accepting.



Once you select “Yes” and proceed, you should see something like the following:

A terminal window titled 'oracle@cloudDB:~' with standard window controls. The terminal output shows an SSH connection process: 'Using username "oracle".', 'Authenticating with public key "rsa-key-20180403"', 'Passphrase for key "rsa-key-20180403":', and finally '[oracle@cloudDB ~]\$' with a green cursor.

```
oracle@cloudDB:~  
Using username "oracle".  
Authenticating with public key "rsa-key-20180403"  
Passphrase for key "rsa-key-20180403":  
[oracle@cloudDB ~]$
```

You're now connected to the host machine where your database instance is being hosted.

Step 4: Now we're going to create a database user (with the name of your choice) both to have and to demonstrate using the SQL\*NET utilities that have been pre-configured to this machine. In the command line of the PuTTY session you have open to the host machine, execute the statement:

```
$ sqlplus / as sysdba
```

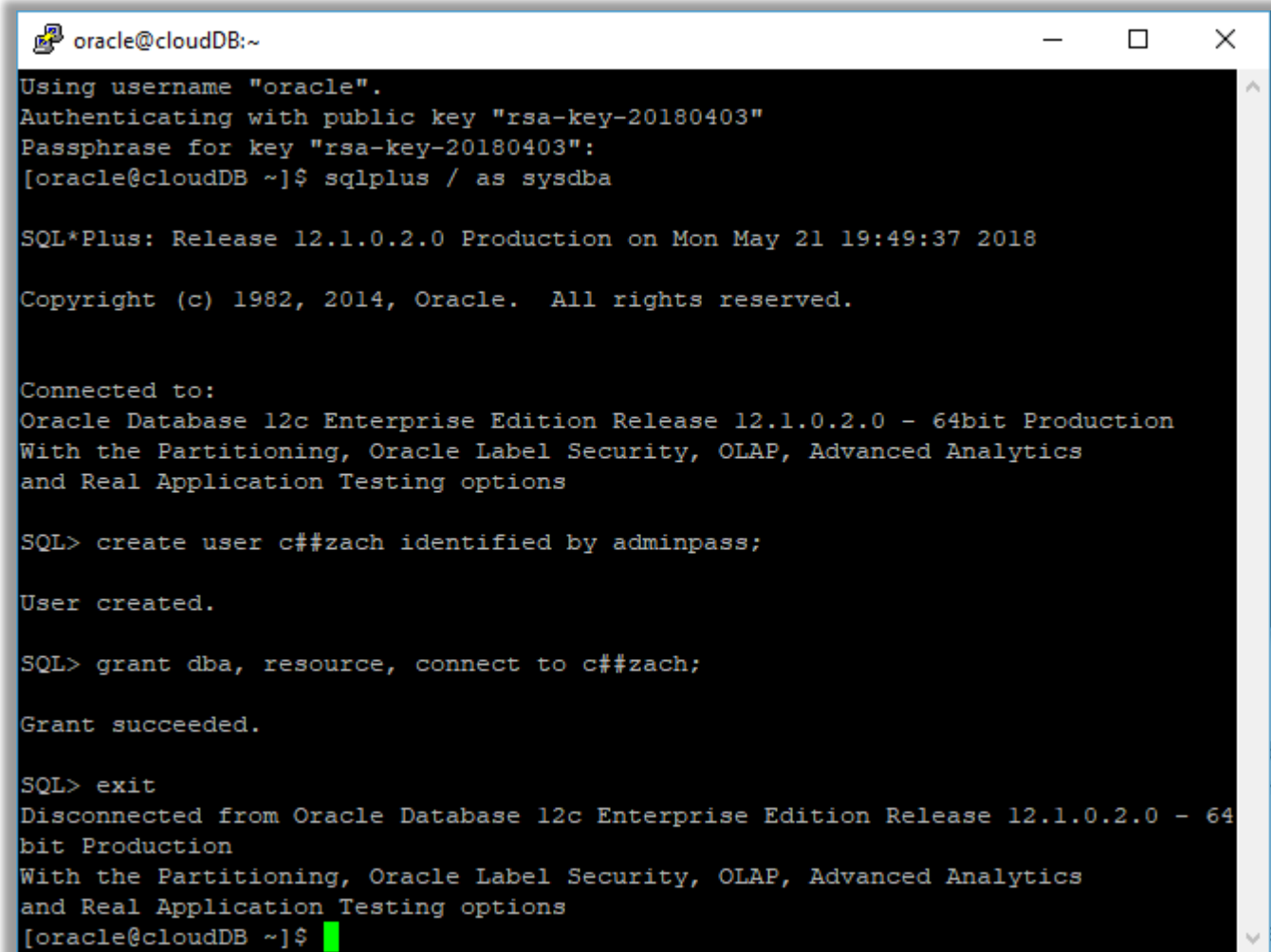
This will open-up the SQL\*Plus utility that we can use to create a new user in our database instance. By default, the DBCS instance we created is a multitenant database (if you want more information, click [here](#)). So, we're going to make a privileged user that exists in both the container database, and the pluggable database within our instance. We will do this by typing the following command from the SQL\*Plus command line:

```
SQL> create user c##zach identified by adminpass;
```

This will create a new user "c##zach" (Zach is my first name, you can concatenate anything you desire after the prefix "c##") with the password "adminpass" (similarly, you can define the password to be whatever you chose). Next, we're going to create our new user "c##zach" some privileges (DBA, resource, connect). To grant the user privileges, execute the following statement from the SQL\*Plus command line:

```
SQL> grant dba, resource, connect to c##zach;
```

You now have yourself a new privileged user in your database. We won't need this SQL\*Plus or PuTTY session for a while, so go ahead and exit both by simply typing and executing "exit" at the command lines. (See below for a whole recap).



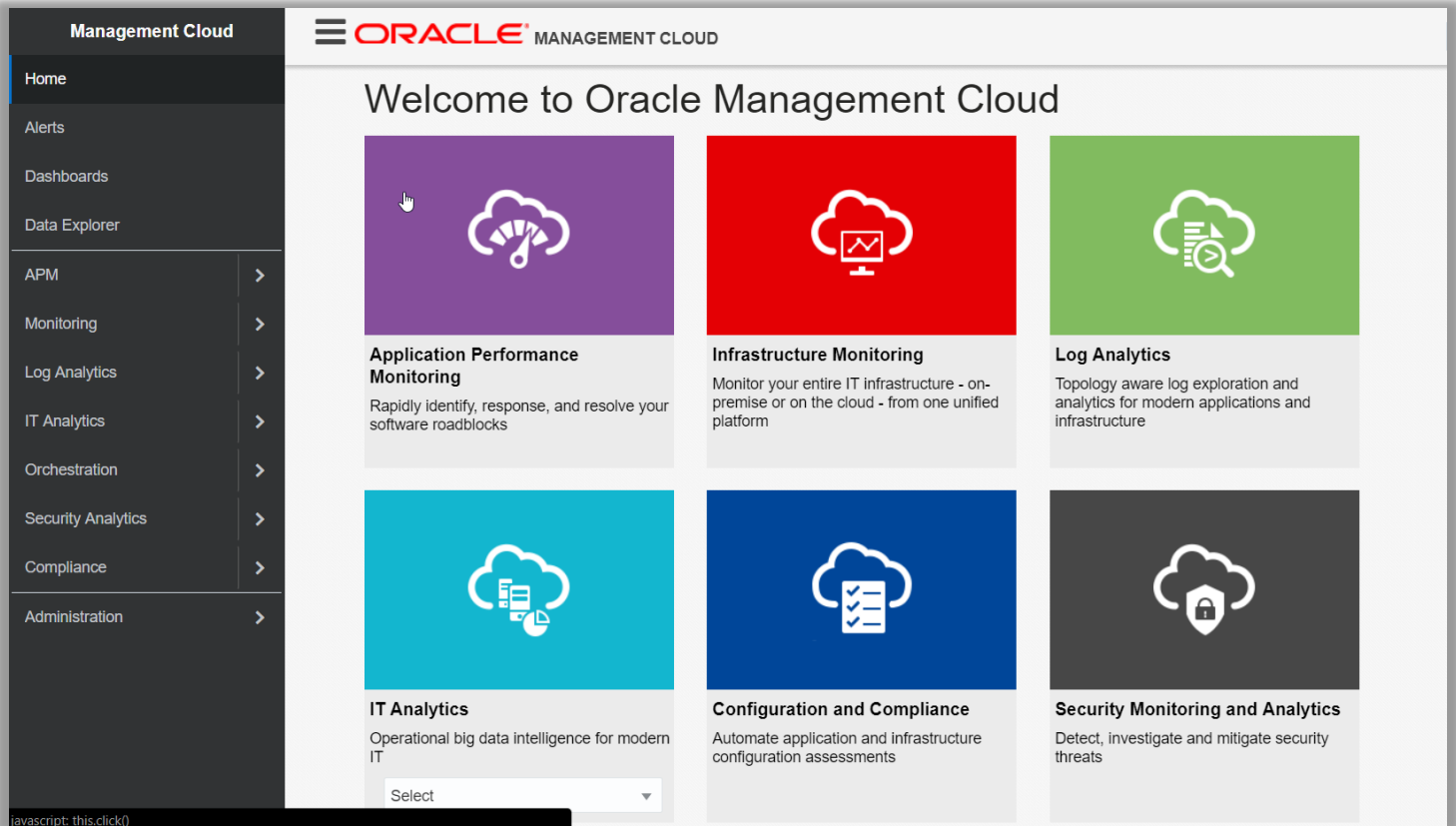
```
oracle@cloudDB:~  
Using username "oracle".  
Authenticating with public key "rsa-key-20180403"  
Passphrase for key "rsa-key-20180403":  
[oracle@cloudDB ~]$ sqlplus / as sysdba  
  
SQL*Plus: Release 12.1.0.2.0 Production on Mon May 21 19:49:37 2018  
  
Copyright (c) 1982, 2014, Oracle. All rights reserved.  
  
Connected to:  
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production  
With the Partitioning, Oracle Label Security, OLAP, Advanced Analytics  
and Real Application Testing options  
  
SQL> create user c##zach identified by adminpass;  
  
User created.  
  
SQL> grant dba, resource, connect to c##zach;  
  
Grant succeeded.  
  
SQL> exit  
Disconnected from Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64  
bit Production  
With the Partitioning, Oracle Label Security, OLAP, Advanced Analytics  
and Real Application Testing options  
[oracle@cloudDB ~]$
```

Keep PuTTY close by, we will need to use in again soon to validate some file transfers we're going to do.

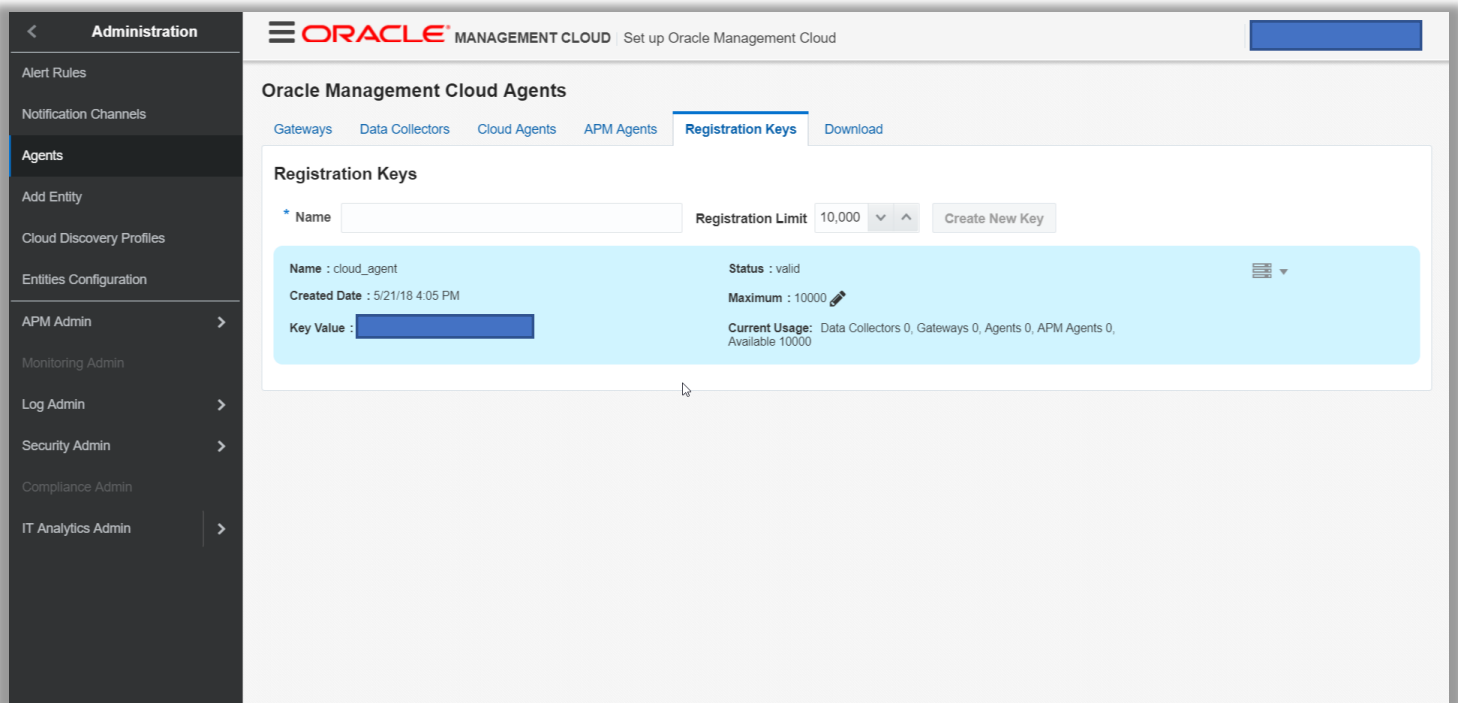
#### Downloading Agent software from OMC instance:

Step 1: Navigate to your OMC instance from the "My Services" Dashboard. You can do this by clicking the action menu at the top left of the dashboard, and selecting "Management" from the "Services" dropdown. Once you are in the OMC service console, click the action menu to the left of your instance name and select "OMC URL" to navigate to the OMC dashboard. Once there, you should see something like this:





Step 2: Navigate to the Agent software download page by selecting “Administration > Agents” from the pop-out menu on the left. Once you’ve selected the “Agents” page, select the “Registration Keys” tab. Now, create a new registration key by naming the key, and selecting “Create New Key”. This should result in something looking like this:



Once you’ve create the registration key, save the key value somewhere (I always save it to the save Notepad document as my Pubic IP, SID, Host Name, and Connect String).

Step 3: Now we're going to download the cloud agent software. To do this, navigate to the next tab on the "Agents" page titled "Downloads". Select "Cloud Agent" from the "Agent Type" drop down menu. You can leave the Operating system drop down as "All". The page should look like the following:

**Oracle Management Cloud Agents**

Gateways Data Collectors Cloud Agents APM Agents Registration Keys **Download**

**Agent Software Download**

Select the agent type to download and the operating system that the agent will be installed on.

[Registration Keys](#) are required to install agent.

\* Agent Type: Cloud Agent ⓘ

Operating System: All ⓘ

Download	Version	Size	SHA1 Checksum
<a href="#">Cloud Agent - AIX Power Systems (64-bit)</a>	1.29.0	577.08 MB	bbf88534c4f942df32dc5ec0b64dc29868f450fa
<a href="#">Cloud Agent - Windows (64-bit)</a>	1.29.0	605.50 MB	2074fdf610797cdd845f905837fe5dd1ee254821
<a href="#">Cloud Agent - Solaris SPARC (64-bit)</a>	1.29.0	434.44 MB	ae8c2617849d2a0548861c656d1781b97c8dda06
<a href="#">Cloud Agent - Linux (64-bit)</a>	1.29.0	482.36 MB	481c5c20ac0b741c97b5d5681f0fd5868d0ed4e0

[Instructions to install the agent](#)

Specify following mandatory parameter values during agent installation.

TENANT\_ID:

UPLOAD\_ROOT:

Before you download the agent software, you should take note of both the TENANT\_ID and the UPLOAD\_ROOT that you see at the bottom of the page (once again, I would save this with the other information you've been saving about your instances). Now, you are going to download the "Cloud Agent – Linux (64-bit)" package. To simplify the process of doing the file transfers later, make a dedicated folder for this lab to save the cloud agent software there. If not, just keep note of the file paths of your encryption keys, the cloud agent software, etc.

### Transferring the Agent Software to the Cloud Host:

Since we're assuming you're using a Windows machine, you will need an SCP client to do the file transfer to the cloud instance. In the pre-requisites section of this lab, you should have already downloaded PSCP along with PuTTY and PuTTYGen. Unlike PuTTY and PuTTYGen, PSCP doesn't have a user interface. We will be using it directly from the Command Prompt of your machine. You are going to either need to configure the executable (PSCP.exe) to your system PATH or move PSCP.exe to the same directory as the downloaded cloud agent .zip file. I have configured PSCP to my system PATH, so my statements will look slightly different than if you're doing it from the same directory as the cloud agent software.

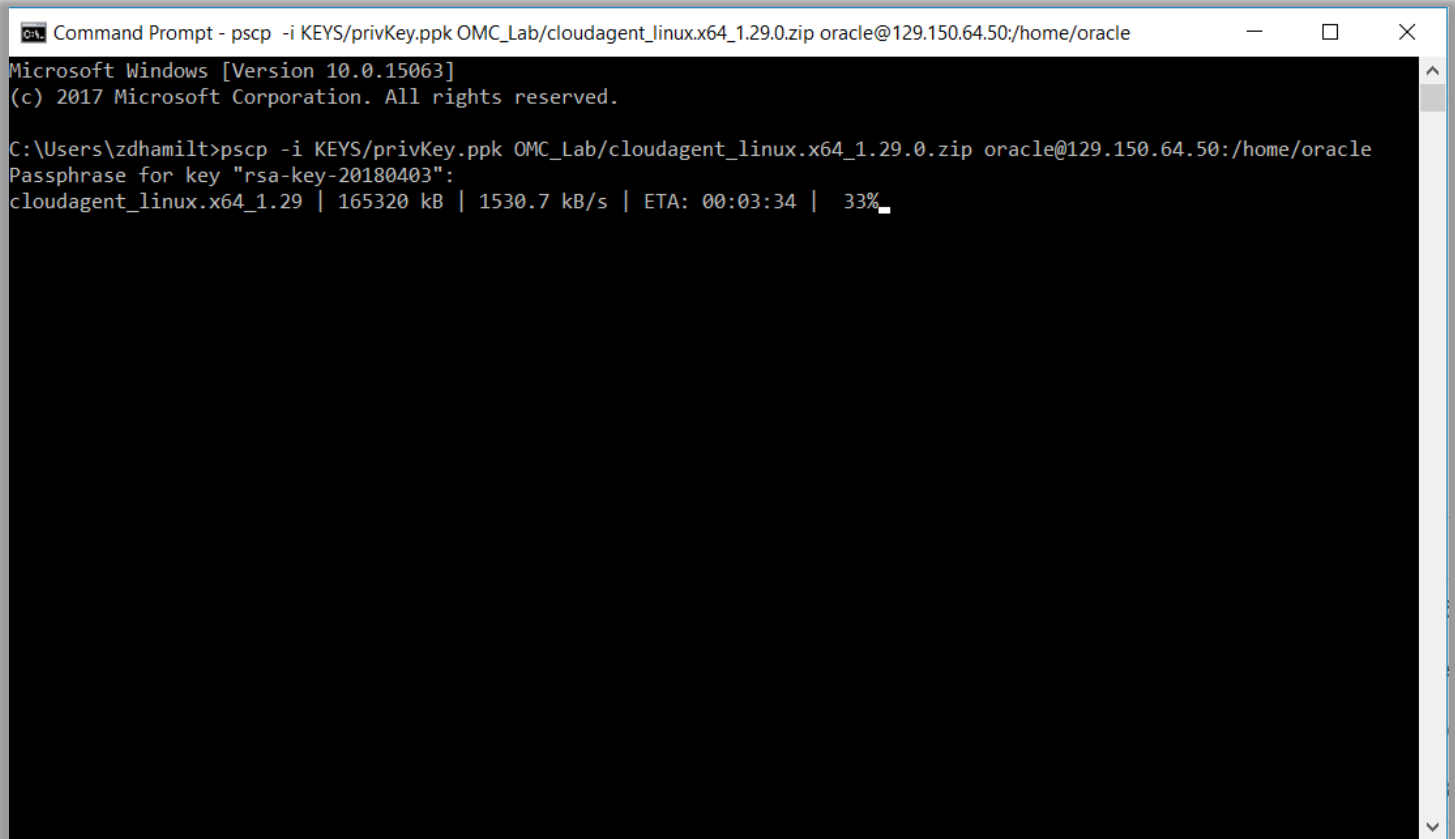
Step 1: If you configured PSCP to the system PATH, the setup of the command will look like the following:

```
> pscp -i <private key> <file to move> oracle@<public ip>:/home/oracle/
> pscp -i path/to/privatekey.ppk path/to/cloudagent_linux.x64_1.29.0.zip
oracle@<public ip>:/home/oracle/
```

If you just moved PSCP.exe to the same directory as cloudagent\_linux.x64\_1.29.0.zip (let's say they're in 'C:/OMC\_Lab/'), then executing the statements from the PWD 'C:/OMC\_Lab/' will look like:

```
> pscp.exe -i <private key> <file to move> oracle@<public ip>:/home/oracle/
> pscp.exe -i path/to/privatekey.ppk cloudagent_linux.x64_1.29.0.zip
oracle@<public ip>:/home/oracle/
```

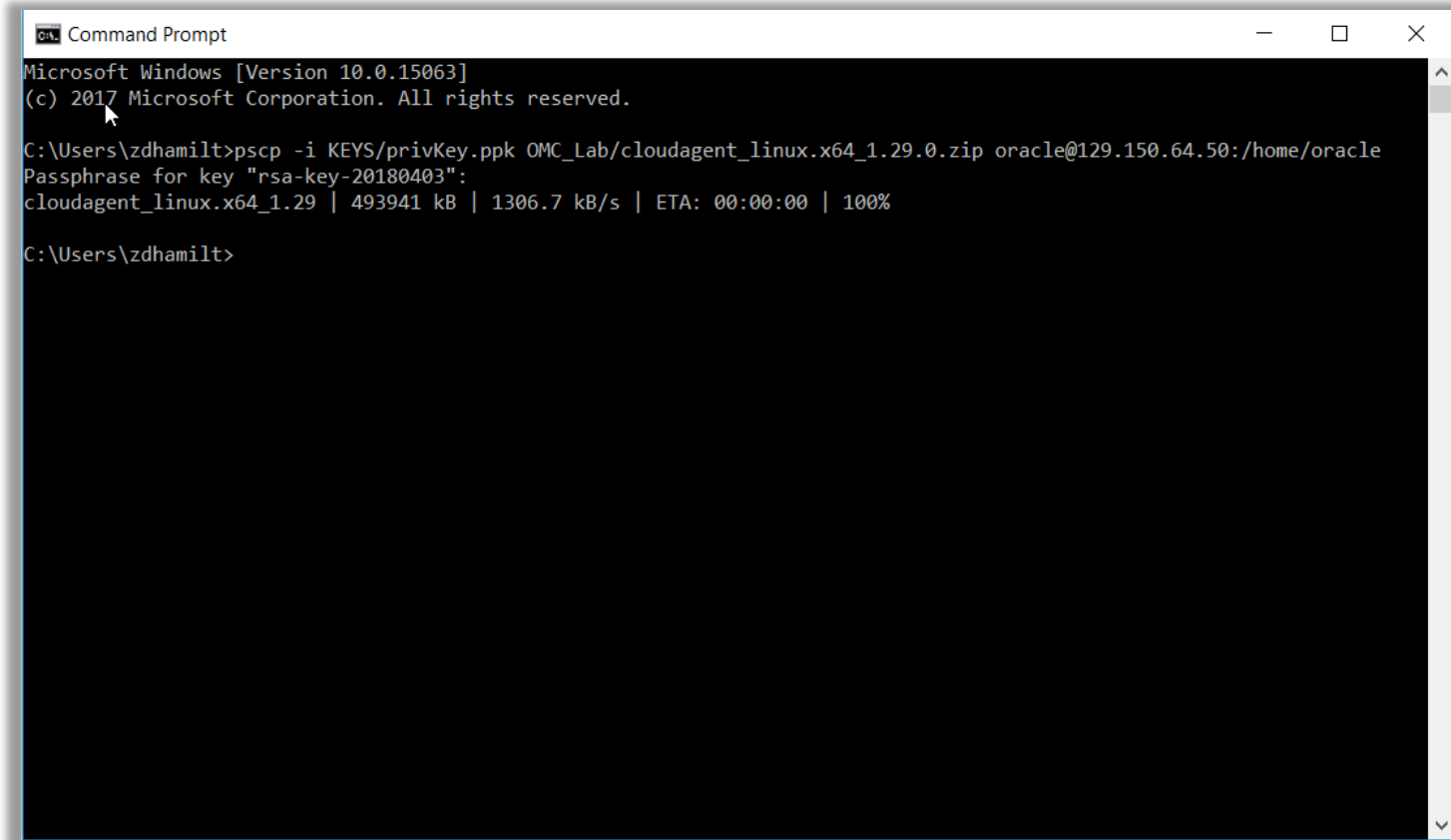
If you've executed those statements correctly, you should see something like this:



```
Command Prompt - pscp -i KEYS/privKey.ppk OMC_Lab/cloudagent_linux.x64_1.29.0.zip oracle@129.150.64.50:/home/oracle
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\zdhmilt>pscp -i KEYS/privKey.ppk OMC_Lab/cloudagent_linux.x64_1.29.0.zip oracle@129.150.64.50:/home/oracle
Passphrase for key "rsa-key-20180403":
cloudagent_linux.x64_1.29 | 165320 kB | 1530.7 kB/s | ETA: 00:03:34 | 33%_
```

Wait for the file to completely transfer to the remote machine that your database is hosted on. Once the transfer is complete, you should see something like this:



```
Command Prompt
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\zdhamil>pscp -i KEYS/privKey.ppk OMC_Lab/cloudagent_linux.x64_1.29.0.zip oracle@129.150.64.50:/home/oracle
Passphrase for key "rsa-key-20180403":
cloudagent_linux.x64_1.29 | 493941 kB | 1306.7 kB/s | ETA: 00:00:00 | 100%

C:\Users\zdhamil>
```

Once you've completed the transfer, we should create another PuTTY session with our remote host to double check that the file is now at `"/home/oracle/"` on our host machine. Open a PuTTY session, select the saved settings by selecting `"cloudDB"` (or whatever you named it), and click `"Load"`. Once the settings are loaded, select `"Open"` to start a new terminal session with the host. Once connected, execute the following statement:

```
$ ls -l
```

If the file transfer was successful, you should see the `.zip` file listed in the terminal session. It should look something like the following:

```
oracle@cloudDB:~  
Using username "oracle".  
Authenticating with public key "rsa-key-20180403"  
Passphrase for key "rsa-key-20180403":  
[oracle@cloudDB ~]$ ls -l  
total 493996  
drwxr-xr-x 3 oracle oinstall      4096 May 16 20:38 bkup  
-rw-r--r-- 1 oracle oinstall 505796202 May 21 20:44 cloudagent_linux.x64_1.29.0.zip  
-rw-r--r-- 1 oracle oinstall      5477 May 16 20:50 dbsetup.out.2816  
-rw-r--r-- 1 oracle oinstall      4218 May 21 19:02 dbsetup.out.2920  
-rwxr-xr-x 1 oracle oinstall    14476 Apr 13 16:37 dbsetup.sh  
-rwxr-xr-x 1 oracle oinstall      3298 Apr 13 16:37 inject-sshkeys.sh  
drwxr-x--- 3 oracle oinstall      4096 May 21 19:27 oracle  
drwxr-xr-x 2 oracle oinstall      4096 May 16 20:26 tmp  
[oracle@cloudDB ~]$
```

The file has been successfully transferred to the remote machine!

#### Installing the Agent software on the remote machine:

Step 1: Now we're going to create a temporary directory to unzip the contents of the .zip file into. From the same PuTTY session, execute the following commands:

```
$ mkdir temp  
$ cd temp  
$ unzip ../cloudagent_linux.x64_1.29.0.zip  
$ ls -l
```

If you did it all right, it should look like the following:

```
oracle@cloudDB:~/temp
Using username "oracle".
Authenticating with public key "rsa-key-20180403"
Passphrase for key "rsa-key-20180403":
[oracle@cloudDB ~]$ ls -l
total 493996
drwxr-xr-x 3 oracle oinstall      4096 May 16 20:38 bkup
-rw-r--r-- 1 oracle oinstall 505796202 May 21 20:44 cloudagent_linux.x64_1.29.0.zip
-rw-r--r-- 1 oracle oinstall      5477 May 16 20:50 dbsetup.out.2816
-rw-r--r-- 1 oracle oinstall      4218 May 21 19:02 dbsetup.out.2920
-rwxr-xr-x 1 oracle oinstall     14476 Apr 13 16:37 dbsetup.sh
-rwxr-xr-x 1 oracle oinstall      3298 Apr 13 16:37 inject-sshkeys.sh
drwxr-x--- 3 oracle oinstall      4096 May 21 19:27 oracle
drwxr-xr-x 2 oracle oinstall      4096 May 16 20:26 tmp
[oracle@cloudDB ~]$ mkdir temp
[oracle@cloudDB ~]$ cd temp
[oracle@cloudDB temp]$ unzip ../cloudagent_linux.x64_1.29.0.zip
Archive:  ../cloudagent_linux.x64_1.29.0.zip
  inflating: unzip
  inflating: AgentDeployment.sh
  inflating: agentimage.properties
  inflating: agent.rsp
  inflating: agent_software_build.xml
 extracting: agentcoreimage.zip
  inflating: AgentInstall.sh
[oracle@cloudDB temp]$ ls -l
total 494076
-rw-rw-r-- 1 oracle oinstall 505707384 Apr 27 04:03 agentcoreimage.zip
-rwxrwxr-x 1 oracle oinstall   32035 Apr 27 04:03 AgentDeployment.sh
-rw-rw-r-- 1 oracle oinstall    226 Apr 27 04:03 agentimage.properties
-rwxr-xr-x 1 oracle oinstall   9421 Apr 27 04:03 AgentInstall.sh
-rw-rw-r-- 1 oracle oinstall   17992 Apr 27 04:03 agent.rsp
-rw-rw-r-- 1 oracle oinstall   1525 Apr 27 04:02 agent_software_build.xml
-rwxr-xr-x 1 oracle oinstall  145976 Apr 27 04:03 unzip
[oracle@cloudDB temp]$
```

Step 2: Now we need to install the agent software from the “/temp” directory that we just created. To do this, we’re going to edit the response file “agent.rsp”. Setting up the response file to be prepared for installation is going to require some information about our OMC instance. The key pieces of information you’re going to need are:

1. TENANT\_ID
2. UPLOAD\_ROOT
3. AGENT\_REGISTRATION\_KEY
4. AGENT\_BASE\_DIRECTORY

Hopefully you’ve saved this information to a Notepad document! If not, you can find the TENANT\_ID and UPLOAD\_ROOT in the OMC console by navigating to: (From left-hand pop-out) Administration > Agents > Download (tab on agent’s page). The information about the UPLAOD\_ROOT and TENANT\_ID should be at the bottom of the page once you select something for the “Agent Type” dropdown. To get the registration key (you should have made this already), you should select the “Registration” tab next to the “Downloads” tab. If you have already created a registration key, it should be shown. If not, create one quickly and then note it.

Once you have those pieces of information we’re going to edit the file on the remote machine. This is one of the more difficult to explain steps in the process, so follow the instructions carefully. If you have experience using Linux solely from the command line, you’re most likely familiar with “vi”. Vi is a text editor that works directly from the command like using only keyboard keystrokes. We’re going to use Vi to add the details about our TENANT\_ID, UPLOAD\_ROOT, etc. to the “agent.rsp” file.

The first thing you’ll want to do is create a new directory that we will define as the “AGENT\_BASE\_DIRECTORY” for the installation in the “agent.rsp” file. I’m going to title mine “cloud\_agent” and I’m going to create it at “/home/oracle/cloud\_agent”. You should have something that looks like mine:

```
oracle@cloudDB:~  
Using username "oracle".  
Authenticating with public key "rsa-key-20180403"  
Passphrase for key "rsa-key-20180403":  
[oracle@cloudDB ~]$ mkdir cloud_agent  
[oracle@cloudDB ~]$ pwd  
/home/oracle  
[oracle@cloudDB ~]$ ls -l  
total 494004  
drwxr-xr-x 3 oracle oinstall      4096 May 16 20:38 bkup  
drwxr-xr-x 2 oracle oinstall      4096 May 23 18:31 cloud_agent  
-rw-r--r-- 1 oracle oinstall 505796202 May 21 20:44 cloudagent_linux.x64_1.29.0.  
zip  
-rw-r--r-- 1 oracle oinstall      5477 May 16 20:50 dbsetup.out.2816  
-rw-r--r-- 1 oracle oinstall      4218 May 21 19:02 dbsetup.out.2920  
-rwxr-xr-x 1 oracle oinstall    14476 Apr 13 16:37 dbsetup.sh  
-rwxr-xr-x 1 oracle oinstall      3298 Apr 13 16:37 inject-sshkeys.sh  
drwxr-x--- 3 oracle oinstall      4096 May 21 19:27 oracle  
drwxr-xr-x 2 oracle oinstall      4096 May 23 18:21 temp  
drwxr-xr-x 2 oracle oinstall      4096 May 16 20:26 tmp  
[oracle@cloudDB ~]$
```

This is going to be the AGENT\_BASE\_DIRECTORY". Also, you may have noticed that the color scheme of my PuTTY session has changed. This can be a very useful thing to do when using Vi because the default color settings make it difficult to read some text within the Vi editor. For instance, compare the next two images.

With PuTTY default color settings:

```
oracle@cloudDB:~/temp
#####
# The Agent Response file lists all the parameters and values required to install the Oracle #
# Management Cloud Agents. The values specified in this file are used by the installer to install #
# the agent. Depending on the type of agent you are deploying, fill the sections below: #
# #
# 1. Gateway: #
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) section. #
# b. If you are using a proxy server, specify values for the parameters listed in the #
# Non-Mandatory Proxy Parameters section. #
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common) section. #
# #
# 2. Cloud Agent: #
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) section. #
# b. If you are using a proxy server, specify values for the parameters listed in the #
# Non-Mandatory Proxy Parameters section. #
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common for #
# Cloud Agent and Data Collector) section #
# #
# 3. Data Collector #
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) and Mandatory #
# Data Collector Parameters sections. #
# b. If you are using a proxy server, specify values for the parameters listed in the #
# Non-Mandatory Proxy Parameters section. #
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common for #
# Cloud Agent and Data Collector) section #
# #
# Note: #
# 1. The parameter values are case-insensitive unless explicitly specified. #
# 2. When running Agentinstall.sh, you can override the value of any valid parameter specified #
# in the .rsp file by passing a new/different value for the same parameter on the commandline. #
# e.g.: When running Agentinstall.sh, if you specify AGENT_BASE_DIRECTORY=/abc/def in the #
# commandline, and if the .rsp file contains AGENT_BASE_DIRECTORY=/xyz/uvw, then the installer #
# will consider AGENT_BASE_DIRECTORY=/abc/def over the value specified in the .rsp file. #
# #
# Refer Agent Installation Guide for further details #
# http://www.oracle.com/pls/topic/lookup?ctx=en/cloud/paas/management-cloud&id=deploy_agent #
#####
##### Mandatory Parameters (Common) #####
1,1 Top
```

With the settings changed:



```
oracle@cloudDB:~/temp
#####
# The Agent Response file lists all the parameters and values required to install the Oracle #
# Management Cloud Agents. The values specified in this file are used by the installer to install #
# the agent. Depending on the type of agent you are deploying, fill the sections below: #
# #
# 1. Gateway: #
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) section. #
# b. If you are using a proxy server, specify values for the parameters listed in the #
# Non-Mandatory Proxy Parameters section. #
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common) section. #
# #
# 2. Cloud Agent: #
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) section. #
# b. If you are using a proxy server, specify values for the parameters listed in the #
# Non-Mandatory Proxy Parameters section. #
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common for #
# Cloud Agent and Data Collector) section #
# #
# 3. Data Collector #
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) and Mandatory #
# Data Collector Parameters sections. #
# b. If you are using a proxy server, specify values for the parameters listed in the #
# Non-Mandatory Proxy Parameters section. #
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common for #
# Cloud Agent and Data Collector) section #
# #
# Note: #
# 1. The parameter values are case-insensitive unless explicitly specified. #
# 2. When running Agentinstall.sh, you can override the value of any valid parameter specified #
# in the .rsp file by passing a new/different value for the same parameter on the commandline. #
# e.g.: When running Agentinstall.sh, if you specify AGENT_BASE_DIRECTORY=/abc/def in the #
# commandline, and if the .rsp file contains AGENT_BASE_DIRECTORY=/xyz/uvw, then the installer #
# will consider AGENT_BASE_DIRECTORY=/abc/def over the value specified in the .rsp file. #
# #
# Refer Agent Installation Guide for further details #
# http://www.oracle.com/pls/topic/lookup?ctx=en/cloud/paas/management-cloud&id=deploy_agent #
#####
1,1 Top
```

The difference is clear. If you are interested in changing the color settings like I have you'll need to go to (in PuTTY) "Window > Colors > Select a color to adjust". Change the "Default Background" RGB values to (255, 255, 255) and then change the "Default Foreground" RGB values to (0, 0, 0).

Step 3: If you've changed the settings or not, it's time to take the information from our Notepad document and add it to "agent.rsp" by leveraging Vi. It's time to follow these steps very closely!

Start by creating a new PuTTY session and connecting to your remote machine. Once you've connected, move to the folder where you unzipped the packaged agent software:

```
$ cd temp
```

Within the "temp" directory, execute the following command:

```
$ vi agent.rsp
```

You should be in the Vi editor now. There's some important things to know. You won't need your mouse for pretty much anything. Use the arrow keys to move the cursor to the "=" after "TENANT\_ID". Now start typing. You should start adding text to the document. Now since the TENTANT\_ID isn't a simple string, we'll ideally copy-paste it into this document. You won't be able to do so with Ctrl + V. Go to your Notepad document and copy the TENANT\_ID string. Now, regain the focus of the PuTTY session. With your cursor still after the "=", simply "Right-click" to paste the string. Don't leave a space between "TENANT\_ID=" and the string. Now, use the arrow keys again to get to the next variable and do the same thing.

Once you are done, press “Escape” once, then type “:x” followed by enter. Once you click “Escape”, anything you type should come up at the bottom, very last, line of the PuTTY session window. The “:x” command will save your changes and close the Vi session. Alternatively, you can also use “:q” to close the session without saving. This is may be confusing if you’ve never used Vi. If you need a guide to it, you can find that [here](#). Everything (right before executing “:x” should look like the following if you’ve done it correctly:

```
oracle@cloudDB:~/temp
#
# 3. Data Collector
# a. Specify values for the parameters listed in the Mandatory Parameters (Common) and Mandatory
# Data Collector Parameters sections.
# b. If you are using a proxy server, specify values for the parameters listed in the
# Non-Mandatory Proxy Parameters section.
# c. Optionally, specify values for the parameters listed in the Non-Mandatory (Common for
# Cloud Agent and Data Collector) section
#
# Note:
# 1. The parameter values are case-insensitive unless explicitly specified.
# 2. When running Agentinstall.sh, you can override the value of any valid parameter specified
# in the .rsp file by passing a new/different value for the same parameter on the commandline.
# e.g.: When running Agentinstall.sh, if you specify AGENT_BASE_DIRECTORY=/abc/def in the
# commandline, and if the .rsp file contains AGENT_BASE_DIRECTORY=/xyz/uvw, then the installer#
# will consider AGENT_BASE_DIRECTORY=/abc/def over the value specified in the .rsp file.
#
# Refer Agent Installation Guide for further details
# http://www.oracle.com/pls/topic/lookup?ctx=en/cloud/paas/management-cloud&id=deploy_agent
#####
##### Mandatory Parameters (Common) #####
# These parameters are mandatory for all agent types.
#####

# Tenant ID where the agent is supposed to upload the data. You can get this information from
# agent administration page "Download" tab in OMC.
# You can also log into the OMC instance and refer the URL first part.
# e.g. URL- https://instance1-oraclepaid.itom.management.us2.oraclecloud.com
# e.g. TENANT_ID=oraclepaid
# e.g. If you have instance name, please use TENANT_ID=<instance name>-oraclepaid
TENANT_ID=

# UPLOAD_ROOT is required to connect to the Oracle Management Cloud for uploading data for the
# specific TENANT_ID. To obtain this value log into the OMC instance and refer the URL. If you
# don't append https:// in the UPLOAD_ROOT value of <TENANT_ID>.<data center> then it will be
# taken care by the installer during installation.
# e.g. UPLOAD_ROOT=<TENANT_ID>.<data center>
# e.g. UPLOAD_ROOT=oraclepaid.itom.us2.oraclecloud.com
# e.g. UPLOAD_ROOT=https://oraclepaid.itom.europe.oraclecloud.com
UPLOAD_ROOT=

# Registration Key - the key to validate identity of tenant and authenticity of the install
# You can obtain registration key from Oracle Management Cloud Dashboard, navigate to
# Administration --> Agents --> Registration Keys.
# e.g.: AGENT_REGISTRATION_KEY=RD0o9uF7dUA3aSzk89u6djYtXU
AGENT_REGISTRATION_KEY=

# Agent Base Directory where the agent needs to be installed on the host machine.
# The directory must be empty and has write privileges for agent install user.
# e.g.: AGENT_BASE_DIRECTORY=/agent/install/gateway
# e.g.: AGENT_BASE_DIRECTORY=D:\agent\install\gateway
AGENT_BASE_DIRECTORY=

:x
```

Now that everything in the “agent.rsp” file is configured properly, we can install the agent! To do this, change to the “/home/oracle/temp” directory where the unzipped software package is found. Once there, you’ll just need to run the install script. This can be done by executing the following commands:

```
$ cd /home/oracle/temp
$ ./AgentInstall.sh
```

This script will install all the agent software into the “/home/oracle/cloud\_agent” directory we defined in the “agent.rsp” file. Once you’ve done that, it should look like this from the command line:

A terminal window titled 'oracle@cloudDB:~/temp' showing the execution of the 'AgentInstall.sh' script. The script performs various steps including authentication, unzipping, installation, and registration of the Cloud Agent. It concludes with instructions on how to start the agent upon system restart and a reference to Oracle's documentation.

```
oracle@cloudDB:~/temp
Using username "oracle".
Authenticating with public key "rsa-key-20180403"
Passphrase for key "rsa-key-20180403":
[oracle@cloudDB ~]$ cd temp
[oracle@cloudDB temp]$ ./AgentInstall.sh
Unzipping agent software, this may take some time...
Installing Cloud Agent...
Cloud Agent parameter validation started...
Cloud Agent pre-requisite checks started...
Cloud Agent base directory creation started...
Security artifacts download started...
Cloud Agent setup started...
Registering Cloud Agent...
Starting Cloud Agent...
Cloud Agent started.
Cloud Agent installation completed.
Cloud Agent post installation checks started.
Cloud Agent is up and running.
Cloud Agent is communicating to Oracle Management Cloud.
Cloud Agent post installation checks completed.
To start Cloud Agent upon Operating System restart include '/home/oracle/cloud_agent/agent_inst/bin/omcli
start agent' in the start-up scripts.
For further details please refer http://www.oracle.com/pls/topic/lookup?ctx=en/cloud/paas/management-cloud&id=agent\_service.
[oracle@cloudDB temp]$
```

The agent is now monitoring the remote machine! It should start sending information to OMC about the host. To configure the agent to monitor the database, you’ll need to run some other scripts in the installation directory (“/cloud\_agent”). Move to the directory with all the scripts for the agent by doing the following:

```
$ cd cloud_agent/plugins/oracle.em.sgfm.zip/1.29.0/scripts/
```

Then try to run the script by doing:

```
$ ./grantPrivilegesMonSvc.sh
```

If you get a permission error like I did, give yourself execute permission:

```
$ chmod u+x grantPrivilegesMonSvc.sh
```

You can now execute the script. It’s going to ask you the Database home, to get this information, execute:

```
$ echo $ORACLE_HOME
```

Now you should have what you need to run the script. Run the script with the same command as above. The prompt will ask you for the “Database Home” (this is the result of \$ORACLE\_HOME), the “SID” (this should be ORCL by default unless you specified otherwise when creating the instance), a SYSDBA user (this should just be SYS), the SYSDBA user password (this is the password you create when you created the instance earlier, and should have written down), the new user to create (this can be anything, its creating a new user in the database to monitor it, so I call my “monitor”), and a password for the new user (you can use anything you want). You should see something like the following if you’ve done that all correctly:

```
oracle@cloudDB:~/cloud_agent/plugins/oracle.em.sgfm.zip/1.29.0/scripts
Using username "oracle".
Authenticating with public key "rsa-key-20180403"
Passphrase for key "rsa-key-20180403":
[oracle@cloudDB ~]$ cd cloud_agent/plugins/oracle.em.sgfm.zip/1.29.0/scripts
[oracle@cloudDB scripts]$ chmod u+x grantPrivilegesMonSvc1.sh
[oracle@cloudDB scripts]$ echo $ORACLE_HOME
/u01/app/oracle/product/12.1.0/dbhome_1
[oracle@cloudDB scripts]$ ./grantPrivilegesMonSvc1.sh
Enter Database Home: /u01/app/oracle/product/12.1.0/dbhome_1
Enter Database SID: ORCL
Enter SYSDBA User: SYS
Enter SYSDBA User password:
Enter Monitoring User to be created : monitor
Enter Monitoring User password :
Enter Output File Name: outputfile
Version: 12.1.0.2.0
WARNING: Provided database details were CDB database, so user will be created with c## prefix.
       70 Grants given to user c##monitor
[oracle@cloudDB scripts]$
```

Everything *should* be set up on the remote machine to now configure a database entity in OMC. Based on desired monitoring, the same process can be taken to other privilege granting scripts in the same directory. Let's go to the OMC console and add an entity.

### Adding an Oracle Database Entity:

To configure the agent to the Oracle Database entity we're going to need a few pieces of information first. The "Add Entity" process is going to ask us some information about the database instance we have running on our remote machine. First thing to do—PuTTY back into your remote machine. Once you have a new session, enter the following command and execute it:

```
$ !snrctl status
```

This should give us the information we need about the instance to add the entity. Hopefully you have something like what I have here:

```

oracle@cloudDB:~
Using username "oracle".
Authenticating with public key "rsa-key-20180403"
Passphrase for key "rsa-key-20180403":
[oracle@cloudDB ~]$ lsnrctl status

LSNRCTL for Linux: Version 12.1.0.2.0 - Production on 29-MAY-2018 12:47:37

Copyright (c) 1991, 2014, Oracle. All rights reserved.

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP) (HOST=cloudDB.compute-601701664.oraclecloud.internal) (PORT=1521)))
STATUS of the LISTENER
-----
Alias                     LISTENER
Version                   TNSLSNR for Linux: Version 12.1.0.2.0 - Production
Start Date                21-MAY-2018 19:01:04
Uptime                    7 days 17 hr. 46 min. 33 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   /u01/app/oracle/product/12.1.0/dbhome_1/network/admin/
listener.ora
Listener Log File         /u01/app/oracle/diag/tnslsnr/cloudDB/listener/alert/lo
g.xml
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=cloudDB.compute-601701664.oraclecloud.internal) (PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc) (KEY=EXTPROC1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcps) (HOST=cloudDB.compute-601701664.oraclecloud.internal) (PORT=5500)) (Security=(my_wallet_directory=/u01/app/oracle/product/12.1.0/dbhome_1/admin/ORCL/xdw_wallet)) (Presentation=HTTP) (Session=RAW))
Services Summary...
Service "ORCL.601701664.oraclecloud.internal" has 1 instance(s).
  Instance "ORCL", status READY, has 1 handler(s) for this service...
Service "ORCL.601701664.oraclecloud.internalXDB" has 1 instance(s).
  Instance "ORCL", status READY, has 1 handler(s) for this service...
Service "pdb1.601701664.oraclecloud.internal" has 1 instance(s).
  Instance "ORCL", status READY, has 1 handler(s) for this service...
The command completed successfully
[oracle@cloudDB ~]$

```

We're going to take the value of "HOST" in the "Listening Endpoints Summary..." and the value in quotation marks in the "Services Summary...". Take a close look at the values I have above and write them down somewhere. Hopefully you remember how to get back to the Management Cloud console. Once you're there, navigate to "Administration > Add Entity". If you haven't added anything yet the page should be blank. Select "Add Entity". Once you are on the page for adding entities, configure the settings the way I have below. You should use the information that you wrote down in the step above. If everything looks the way I have it below, click "Add Entity".

Add Entity

Add Entity

Cancel

Add entities and monitor them using a cloud agent in Oracle Management Cloud. This process might take several minutes. Once started, it cannot be cancelled. Before adding an entity, make sure that all the [pre-requisites](#) are met. If your entity type is not on the list, refer to the [documentation](#) for adding entities using omcli.

\* Entity Type

Oracle Database

\* Mode

Service Name

\* Entity Name

dbcs

Entity Display Name

dbcs94

\* Server Name

cloudDB.compute-601701664.oraclecloud.internal

\* Port

1521

\* Service Name

ORCL.601701664.oraclecloud.internal

\* Cloud Agent

cloudDB.compute-601701664.oraclecloud.internal:4459

\* Job Name

Oracle\_Database\_1528117671726

**Monitoring Credentials**

These are the credentials that will be used by the cloud agent for monitoring. The credentials will not be saved in Oracle Management Cloud.

Database Credentials

\* Username

c##monitor

\* Password

\*\*\*\*\*

\* Confirm Password

\*\*\*\*\*

\* Database Role

NORMAL

Now you have configured an Oracle Management Cloud agent to communicate with a remote Linux machine as well as an instance of Oracle Database! You can now do to the dashboards page and start seeing the ways that information about the machine and the database are being pulled. I strongly suggest restoring a database backup to the database instance in order to generate some log files within the database that you can then use for discovery.

That's all for now! More will come later.